5-2019

Emerging Security Threats in Modern Digital Computing Systems: A Power Management Perspective

Rajesh Jayashankara Shridevi
Utah State University

EMERGING SECURITY THREATS IN MODERN DIGITAL COMPUTING SYSTEMS: A POWER MANAGEMENT PERSPECTIVE

by

Rajesh Jayashankara Shridevi

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

Approved:

Koushik Chakraborty, Ph.D. Major Professor	Sanghamitra Roy, Ph.D. Committee Member
Todd Moon, Ph.D. Committee Member	Zhen Zhang, Ph.D. Committee Member
Paul Barr, Ph.D. Committee Member	Richard Inouye, Ph.D. Vice Provost for Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2019

ABSTRACT

Emerging Security Threats in Modern Digital Computing Systems: A Power Management Perspective

by

Rajesh Jayashankara Shridevi, Doctor of Philosophy

Utah State University, 2019

Major Professor: Koushik Chakraborty, Ph.D.
Department: Electrical and Computer Engineering

Power management units (PMU) have come into the spotlight with energy efficiency becoming a first order constraint in modern digital computing systems. Dynamic power management solutions are spread across multiple layers of the hardware and software design spectrum, often requiring synergistic operation of power control knobs. To cater to the exponential rise in power events, and to meet the demands of tight power and energy budgets, power management solutions are evolving into feature rich, complex and intelligent designs. The evolution of this industry has created a dynamic, volatile, and competitive landscape, resulting in several third-party solutions entering the niche market, raising concerns of trust and security assurance.

This research embarks on cross-layer analysis to identify, classify, detect and prevent emerging threats in power management solutions. To uncover a broad spectrum of possible security violations, two incarnations of security problems stemming from the rampant use of untrustworthy third-party components are explored. One targeting specialized hardware PMU, and the other, complex software power management modules in the data-center hypervisor. Security vulnerability in these complex sub-systems arise either

from security oblivious design artifacts, or from intentional insertion of trojans. Comprehending the challenges of security assurance by attack realization, this research proposes novel non-invasive detection and defense mechanisms against compromised power management units.

Today, power management is synonymous with digital computing. Given the proliferation of digital devices in all walks of life, be it in battery operated mobile devices or, high-performance data-centers, this research potentially has far reaching impacts on our society. Altogether, this work pioneers research in a critical, yet vastly unexplored research problem: *providing secure and reliable computing in the presence of an untrustworthy third-party power management components*.

(104 pages)

PUBLIC ABSTRACT

Emerging Security Threats in Modern Digital Computing Systems: A Power Management Perspective

Rajesh Jayashankara Shridevi

Design of computing systems — from pocket-sized smart phones to massive cloud based data-centers — have one common daunting challenge : minimizing the power consumption. In this effort, power management sector is undergoing a rapid and profound transformation to promote clean and energy proportional computing. At the hardware end of system design, there is proliferation of specialized, feature rich and complex power management hardware components. Similarly, in the software design layer complex power management suites are growing rapidly. Concurrent to this development, there has been an upsurge in the integration of third-party components to counter the pressures of shorter time-to-market. These trends collectively raise serious concerns about trust and security of power management solutions.

In recent times, problems such as overheating, performance degradation and poor battery life, have dogged the mobile devices market, including the infamous recall of Samsung Note 7. Power outage in the data-center of a major airline left innumerable passengers stranded, with thousands of canceled flights costing over 100 million dollars. This research examines whether such events of unintentional reliability failure, can be replicated using targeted attacks by exploiting the security loopholes in the complex power management infrastructure of a computing system.

At its core, this research answers an imminent research question: *How can system designers ensure secure and reliable operation of third-party power management units?* Specifically,

this work investigates possible attack vectors, and novel non-invasive detection and defense mechanisms to safeguard system against malicious power attacks. By a joint exploration of the threat model and techniques to seamlessly detect and protect against power attacks, this project can have a lasting impact, by enabling the design of secure and cost-effective next generation hardware platforms.

To my family who encouraged me to pursue my dreams.

ACKNOWLEDGMENTS

This dissertation represents the culmination of a profound and truly rewarding journey made possible by numerous thoughtful individuals. Filled with nostalgia and deep gratitude, I ink these few words conveying my sincere appreciation for their fruitful impact on my life.

Foremost, I would like to express my sincere gratitude to Dr. Chakraborty and Dr. Roy for their continuous professional support of my Ph.D study and research, for their patience and motivation during paper rejections, for their endless echoes of encouragement to strive for perfection, for their belief in my abilities, for inspiring and advocating an indomitable attitude. I have greatly benefited by working closely with them for the past six years. Their work ethic and contrasting mentoring styles facilitated a balanced and holistic development.

Besides my advisors, I would like to thank the rest of my dissertation committee: Dr. Moon, Dr. Zhang and Dr. Barr for their encouragement, responsiveness, insightful comments, and hard questions. I would also like to acknowledge my graduate coordinator Tricia Brandenburg for her positive energy, welcoming attitude, timely and assuring counsel. Her quick and clear guidance on various administrative matters has been vital for fulfilling my potential. I would also like to express my great appreciation to the ECE department, for this opportunity to pursue my doctoral education here, as well as, the financial assistance towards my tuition.

I would like to thank various members of Bridge Lab for their constant support and for making Bridge Lab such a pleasant and rewarding place to work. Yiding, Dean and Hu were always approachable for technical guidance, as well as, words of encouragement, and played a key role during my formative years in the lab. Harshitha and Manzi, with whom I shared many meaningful discussions, moments of anxiety, pressure of paper submission, and the first joy of publishing. I would like to thank Chidam and Prabal, with whom I have worked closely on different projects, and shared interesting moments as fellow teaching

and grading assistants. Aatreyi, Asmita, Sourav and Tahmoures have all been very encouraging, made life in the lab fun and rewarding. I appreciate having the opportunity to work with these talented and colorful individuals, for many stimulating discussion and for fostering healthy competition in Bridge Lab.

A very special gratitude goes out to my friends Abhishek, Niranjan and Abhijit, who have tolerated me, stood by me through my highs and lows, kept me sane, motivated and happy. Nostalgia strengthens my gratitude to them, for they have been my family away from home. A special mention to my eternal cheerleader, Nivedita, for reinforcing my self-confidence and energizing me to work harder.

Finally, my deepest and heartfelt gratitude to my family. My parents nurtured my dream with relentless love, patience, support and encouragement, and I am indebted to their sacrifice. My brother, Swamy, has been my one-stop solution to all problems, and a beacon helping me navigate the hurdles of life away from home.

Rajesh Jayashankara Shridevi

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

3PIP	third party intellectual property
ACPI	advanced configuration and power interface
API	application programming interface
ASIC	application specific integrated circuit
CAGR	compound annual growth rate
DEB	delay estimation block
DES	data encryption standard
DPM	dynamic power management
DVFS	dynamic voltage-frequency scaling
FDR	false detection rate
FN	false negative
FP	false positive
FSM	finite state machine
GEOPM	global extensible open power manager
ICT	information and communication systems
IMVP	Intel mobile voltage positioning
IoT	internet of things
IP	intellectual property
IPRAM	intellectual property risk assessment module
IQR	inter quartile range
LFSR	linear feedback shift register
MIPS	million instructions per second
MLF	machine learning framework
MMT	minimum migration time
MPSoC	multi processor system on chip
NoC	network-on-chip
NTC	near threshold computing

OS	operating system
P-State	performance state
PCM	preempt control module
PM	P-Virus monitor
PMIP	power management intellectual property
PMU	power management unit
QoS	quality of service
RAB	response audit block
SC-SVM	single class support vector machine
SCALE	single class assisted learning environment
SLA	service level agreement
SLATAH	SLA time per active host
SVID	serial voltage identification
SVM	support vector machine
TDP	thermal design power
TU	tactical unit
UPF	unified power format
VID	voltage identification
VM	virtual machine
VMM	virtual machine monitor
WA	wake-up alarm

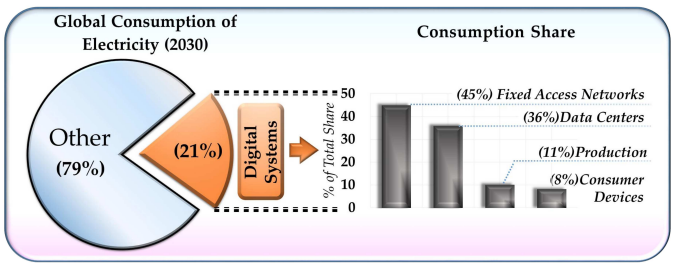
CHAPTER 1

INTRODUCTION

Design of modern information and communication technology (ICT) systems—from miniature internet of thing (IoT) devices to massive cloud based data centers—have one common, daunting challenge : ***minimizing the power consumption***. Recent studies have shown that ICT systems currently consume nearly 10% ($\sim 1500TWh$) of the world's electrical energy, and its share is expected to rise 21% by the year 2030 (Figure 1.1) [1]. A major percentage (81%) of this energy consumption is dominated by high performance digital computing systems (data-centers) and fixed access networks due to an unprecedented rise in data/content generation and consumption. Accordingly, the CO_2 emissions related to digital systems will contribute to approximately 4% of global emissions in four years. Realizing the environmental and social impact, the digital computing sector has triggered a rapid and profound transformation to promote clean and energy efficient computing.

1.1 Trends in Power Management

The power management in modern computing systems encompasses several layers of the design spectrum including the application, operating system (OS), hypervisor, low-level software such as firmware and device drivers, hardware architecture, circuit and device layers (Figure 1.2a). Figure 1.2b shows the landscape of power management and the



(a)

Fig. 1.1: Components of ICT and its contribution to the global electricity (2030).

increasingly complex interactions between the hardware and software power management features. The vehement emphasis on energy efficiency has put ICT power management front and center, and fueled extensive technological innovations at all abstraction levels of the ICT design, rendering a very complex power management architecture [2].

At one end of the design spectrum, third-party computing power management software suites have swelled to a net size of $314 million, with a growth of 35.48% during the years 2012-2016 [3]. These software suites promise to deliver $18.6 billion/year in cost reduction [4], complementing the rapid evolution of intelligent and complex power management algorithms in OS kernels, and device drivers.

At the other end, extensive proliferation of specialized hardware components for chip level power management—*power management units* (PMU)—are dominating more than 50% of the analog integrated circuit (IC) market today [5]. PMUs offer a host of services ranging from system start-up and shut-down to dynamic control of power rails, voltage scaling and management of power states. Traditional simple PMUs for low cost power conversion are evolving into feature rich, hardware controlled and intelligent power management IPs (PMIPs) to increase energy efficiency and provide flexible power management in high-end multi-processor system on chips (MPSoC) [6]. This trend in hardware PMUs is part of a broader trend of upsurge in the integration of third-party intellectual property (3PIP) components within a single hardware platform, so as to achieve a rapid time to

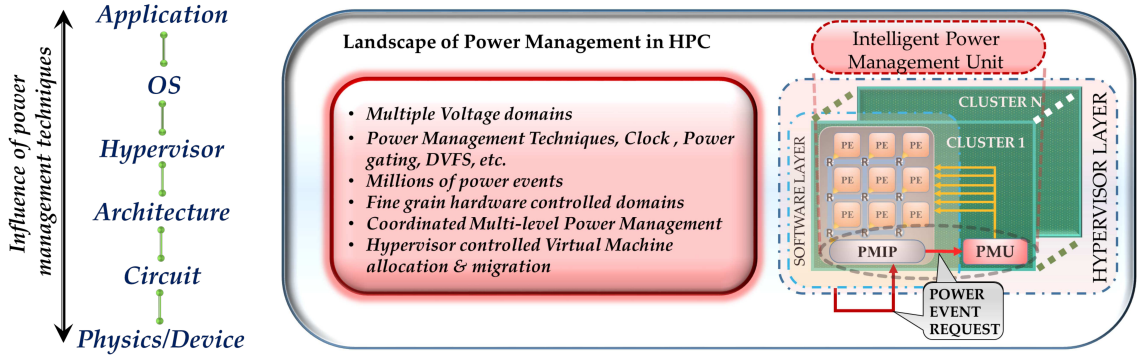


Fig. 1.2: Landscape and scope of power management features.

market [7, 8].

1.2 Problem : Security Vulnerabilities in Power Management

The influx of copious technological innovations for efficient power and thermal management, has brought forth a new potent attack surface—***the complex power management architecture***. The intricate interplay of the control signals across the design layers, combined with the phenomenal growth and integration of niche third-party components poses intriguing questions regarding the trustworthiness of the upcoming power management architecture. This research tackles a critical, yet vastly unexplored research question: **How can we assure reliable and trustworthy computing in the presence of untrustworthy third-party components in the power management architecture?**

A malicious trojan embedded in the third-party power management unit can cause sporadic errors, reliability issues, or even catastrophic failures. These attacks can result in data corruption, denial of service, or degradation of the system performance and energy efficiency. For example, a malicious increase in the supply voltage can cause a surge in the peak power and chip temperature, leading to thermal throttling, functional errors and even a systematic chip failure. This research embarks on a cross-layer and comprehensive analysis to identify, classify, detect and prevent emerging threats from the power management systems. With the evolution of power management systems, both in software and hardware, a study of their influence on system functionality, performance and energy efficiency is critical to ensure the *confidentiality*, *integrity* and *availability* of future computing systems. The project can spawn a radical shift from the current practices, where the power management system is assumed to be trustworthy, and thus free of malicious intent.

1.3 Impact of Security Vulnerabilities in Power Management

Numerous events in the recent past highlight the danger emanating from vulnerabilities in power management systems. In August 2016, a power outage in the data center of Delta airlines left innumerable passengers stranded, with thousands of canceled flights,

costing over a 100 million dollars [9]. Similarly, an outage in the Verizon data-center resulted in JetBlue flight delays, along with shutdown of the airline's website and booking systems [10]. A failure of the Oregon government services data-center delayed unemployment payments and loss of email access to government employees [11].

In the last few years, a plethora of problems such as overheating, performance degradation and poor battery life, have dogged the mobile devices market, including the recall of Samsung Note 7 [12]. In 2012, complaints about the Apple iPad's wireless reliability were traced back to a power-saving feature that ran amok and starved the Wi-Fi chip of the required power [13]. The automobile industry, with its emerging massive digital system integration, is also at the cusp of this problem. For instance, Chrysler had to recall more than $300,000$ vehicles due to a faulty *integrated power module* [14]. Similarly, Toyota announced a safety recall of $807,000$ Prius cars due to a malfunction in the power management electronics control unit that prevented the vehicle from entering fail-safe driving mode as intended [15]. In the safety critical military Reaper drone, mysterious electrical failures have caused a surge in major drone crashes [16].

Deliberate attacks on the complex power management infrastructure can damage ICT systems, often resulting in severe loss of productivity, economy and reputation. Loss of access to services in the health-care, finance and transportation sectors can prove catastrophic. Irrespective of whether these events are results of unintentional reliability failures or targeted attacks, they have a profound significance on trustworthy computing in the modern world. *A key research question then is how can we identify threats and defense mechanisms to protect this emerging attack surface on our power management systems?*

1.4 Contributions of this Research

This systematic investigation lays a solid foundation to security assessment in power management by uncovering a new class of security vulnerabilities and the imminent threat they pose to the emerging computing paradigm. In this pursuit, this research explores cross-layer methodologies for the design of secure power management platforms and tech-

niques. Figure 1.3 summarizes the collective threat model investigated in this work. At one end of the design spectrum, this work uncovers the impact of hardware trojan embedded in the power management unit of a MPSoC. At the other end, this work identifies and analyzes the potency of malicious software power management module in the virtualization suite of the data-center. Although seemingly disjoint, at their core, these two security problems stem from the rampant use of untrustworthy third-party components in the complex power management sub-system.

Publications in direct relation to this dissertation are listed below:

- **Securing Data Center Against Power Attacks**, JS, Rajesh, Rajamanikkam, Chidhambaranathan, Chakraborty, Koushik, Roy, Sanghamitra. *Journal of Hardware and Systems Security.* (2019)

- **Catching the Flu: Emerging Threats from a Third Party Power Management Unit**, JS, Rajesh, Chidhambaranathan Rajamanikkam, Koushik Chakraborty, and Sanghamitra Roy. *Proceedings of the 53rd Annual Design Automation Conference (DAC)*, (2016).

- **Runtime Detection of a Bandwidth Denial Attack from a Rogue Network-on-Chip**,

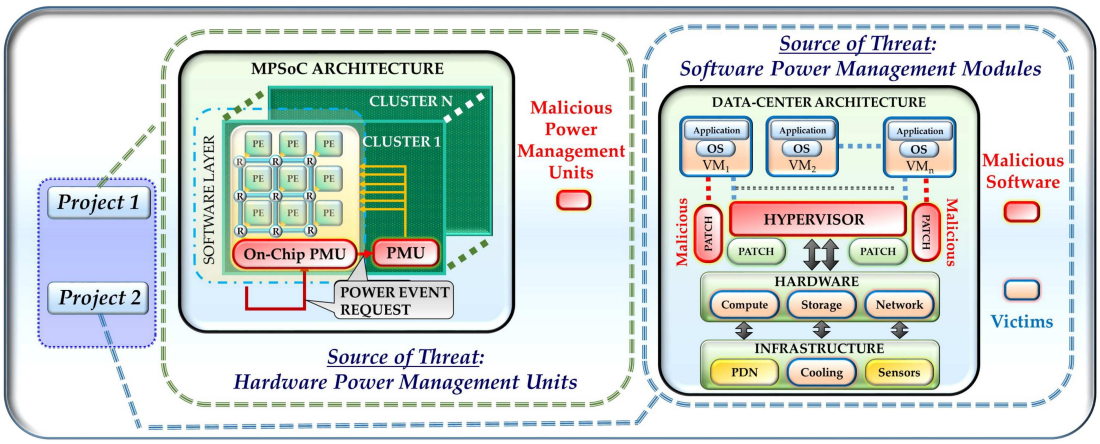


Fig. 1.3: Research outline: cross-layer security issues stemming from third-party the power management components.

JS, Rajesh, Dean Michael Ancajas, Koushik Chakraborty, and Sanghamitra Roy, *Proceedings of the 9th International Symposium on Networks-on-Chip (NOCS)*, (2015).

This research spawns a radical shift from the current practices, where the power management features are assumed to be from a trusted source, and thus free of malicious intent. This research can prove instrumental in reshaping the design practices of future digital computing systems. The specific contributions are as follows:

- Uncovers two covert threats to the MPSoC, stemming from a hardware trojan embedded in an untrustworthy third-party hardware power management unit. Performs detailed analysis on the impact of attacks on performance and energy efficiency of the computing system.

- Proposes a novel *intellectual property risk assessment module* (IPRAM) that can be implemented in-house by an MPSoC integrator to evaluate power management request-response combinations and detect malicious activities in the PMU.

- Designs and evaluates two attack specific non-invasive security solutions, that are bundled into the IPRAM to aid runtime monitoring of power management decisions. These solutions effectively detect the targeted malicious activity, while incurring low design overheads (less than 1%) in area and power.

- Investigates an incarnation of software-centric threat in the power management subsystem of a high performance data-center. The devised threat exploits security loopholes to carry out a broad range of potent power and energy attacks compromising the system behavior.

- Introduces a secure sub-system architecture by employing a machine learning algorithm to monitor and learn acceptable data-center power characteristics under dynamic workloads. The proposed architecture safeguards the data-center by identifying malicious power consumption behavior. Central to the secure architecture are concepts of isolation and redundancy.

- Evaluates the efficacy of the proposed learning framework and tunes the solution to achieve a high accuracy in distinguishing between legitimate and anomalous power consumption behavior for the examined data-center characteristics.

1.5 Dissertation Outline

The dissertation is organized as follows:

- Chapter 2 equips the reader with a thorough review of contemporary research on the growing complexities of the power management sub-system, security vulnerabilities in third-party components, cloud security and power management verification to highlight the need for research on *exposing the security vulnerabilities in power management*.

- Chapter 3 outlines the generic threat model to realize compromised third party software and hardware components, and discusses the relevance of such vulnerabilities in modern computing ecosystem.

- Chapter 4 presents a detailed analysis of two covert security threats, *P-Virus* and *Drowsy* stemming from a hardware trojan embedded in a malicious third-party PMU. Consequently, the chapter presents the design specifics of the proposed non-invasive IP risk assessment module to monitor the trust and security of deployed PMU by matching response to requests at critical interfaces.

- Chapter 5 uncovers a potent threat from a compromised software power management module in the hypervisor to motivate the need to safeguard data-centers from power attacks. Next, it details the proposed machine learning based secure architecture to detect anomalous power consumption activity and protect against catastrophic power outages.

- Chapter 6 summarizes the contribution of this research, and paves the way for future research in this domain by highlighting the key challenges for security in power management solutions.

1.6 Breadth of Doctoral Work

This section presents the synopsis of research publications in the field of security and reliability of system-on-chip components published during the course of this doctoral research.

- *Runtime detection of a bandwidth denial attack from a rogue network-on-chip* [17, 18] - This research investigates the threat of a hardware trojan in a third-party network-on-chip (NoC). A malicious trojan in the NoC can disrupt the availability of on-chip communication resources, thereby creating a huge performance bottleneck. The work proposes a runtime latency auditor that enables the multi-processor system-on-chip (MPSOC) integrator to monitor the trustworthiness of the deployed NoC throughout the chip lifetime.

- *Tackling voltage emergencies in NoC through timing error resilience* [19] - This research tackles the reliability concern due to voltage emergencies in forthcoming NoC designs. The work proposes a droop resilient NoC architecture with two techniques to mitigate and recover from voltage emergency induced timing errors. The proposed techniques use concepts of frequency down-scaling and pipeline error recovery to reclaim corrupt flits in the router, and maintain a reliable and energy efficient operation.

- *Tackling voltage noise in the NoC power supply through flow-control and routing algorithms* [20, 21] - This research proposes a collection of novel flow-control protocols and an adaptive routing algorithm to mitigate the power supply noise in NoC and preserve power supply integrity.

- *BoostNoC: Power efficient network-on-chip architecture for near threshold computing* [22] - This research investigates a promising and emerging design approach for energy efficient systems—near threshold computing (NTC). The work demonstrates that NoC can prove to be a performance bottleneck in many-core NTC systems. To reclaim the performance lost due to sub-optimal design in the NTC regime, the work proposes

and evaluates *BoostNoC*—a power efficient, multi-layered NoC architecture. *BoostNoC* exploits the temporal and spatial variation of on-chip communication demand in a MPSoC under dynamic workloads, and seamlessly shifts between two layers optimized for contrasting design parameters: performance and power-efficient data transmission.

- *Catching the flu: Emerging threats from a third-party power management unit* [23] - This work investigates the security vulnerabilities stemming from complex power management hardware sub-systems. The research demonstrates two covert attacks designed using a hardware trojan embedded in the power management unit. To detect malicious power management behavior and monitor trustworthiness the work proposes a non-invasive IP risk assessment module.

- *Securing data-center against power attacks* [24] - This work uncovers a potent threat emerging from vulnerabilities in the power management module of a data-center hypervisor. The compromised power management module inflicts a power attack and increases the data-center power consumption significantly. The work investigates a secure learning framework based on a support vector machine learning algorithm to detect anomalous power consumption behavior in the data-center.

- *Understanding Security Threats in Emerging Neuromorphic Computing* - This yet to be published research, investigated the security loopholes in a neuromorphic system design environment. The work examined multiple exploitable attack vectors stemming from hardware trojan insertion, as well as, from inherent device, circuit and implementation specific vulnerabilities. The dissection of these vulnerabilities will help in systematic development of collective security primitives and design-for-trust solutions in emerging neuromorphic systems.

CHAPTER 2

LITERATURE SURVEY

Demand for energy efficiency, intelligent battery management, energy harvesting and wireless charging features continue to re-sculpt ICT industry's power management landscape. System evaluation metrics are evolving from pure performance based measures (operating frequency, throughput, etc.,) to energy centric parameters (performance per watt, energy delay product, etc.,) to reflect the growing demand for energy proportional computing. Consequently, an alarming trend seen in the rapidly evolving power management solution space is the proliferation of digital complexity for reliable and efficient power delivery. Designers of ICT systems such as smartphones, notebooks, workstations and, even the cloud infrastructure are becoming increasingly reliant on third-party vendors to fulfill the complex and niche power requirements [25]. This Chapter introduces the advances in power management solutions across all design layers, and emphasizes the lack of research in security and trust assurance of power management solutions.

Section 2.1 and Section 2.2 together, delineate the landscape of modern power management sub-systems to trace the progression of the domain. Section 2.3 summarizes previous literature on power management security to assess the state-of-the art, and clarify the significance of our research. Finally, Section 2.4 presents other security related research most relevant to this project to justify the choice of research topic.

2.1 Landscape of Power Management Hardware

Modern hardware PMUs include a broad portfolio of digitally enabled, configurable and fully programmable complex power controllers. Table 2.1 presents the taxonomy of evolving hardware PMU.

Off-chip hardware PMUs integrate power converters required to deliver power to processors, interfaces, memory, and various sensors, low drop-out linear regulators (LDO)

for peripheral devices, battery charging, and control intelligence [26–28]. Features such as active and dynamic voltage positioning based on dynamic load are becoming increasingly important for both thermal management, as well as, longer battery life. Technology standards such as Intel's mobile voltage positioning (IMVP) define strict specifications regarding input pins, control signals, output signals, softstart, restart and voltage-slew rate control. Many modern PMUs have additional capabilities that include system clock generation, analog to digital conversion and digital input/outputs.

Advanced hand-held low power systems and MPSoCs require controlled and choreographed sequencing of various power domains as on-chip IP blocks have critical timing dependencies. Additionally, differences in battery chemistry necessitate unique charging techniques, algorithms and voltages to optimize battery performance, capacity, reliability, safety, thermal behavior and longevity [29–31]. PMU targeted for smartphones and tablets are highly programmable and reconfigurable units equipped with essential real-time battery monitors to protect against harmful charging and discharging conditions [31].

Finally, to address the growing demand for power fidelity and integrity in shrinking MPSoCs, novel on-chip power management IPs are on the rise [6, 32, 33]. These IPs allow for fine grain power management with features supporting hundreds of power domains,

Hardware PMU	Description and Sample Systems
Programmable PMUs	Functionality includes sequencing voltage rails, programmable regulators, intelligent battery charging, among others. Popular PMUs have adopted Intel's smart regulation technique called *Intel mobile voltage positioning* (IMVP) specifications for their design [26–28].
Feature rich PMUs for portable devices	Functionality includes efficient battery management, while simultaneously operating a host of sensors, camera, radio and wireless connectivity, all powered from a single lightweight battery. [29–31].
On-chip fine grain PMU	New generation PMU designed as an on-chip IP to support faster and simultaneous power states transitions of multiple circuits and voltage islands. Sonics ICE-Grain power architecture is the industry's first such solution [6, 32, 33].

Table 2.1: Classification of hardware PMUs employed by modern MPSoCs.

power gating of voltage islands and operation at near threshold voltage. These fine grain IPs cater to very fast switching activity while tackling issues such as jitter, clock skews, electromigration, coupling noise and power distribution droops.

2.2 Landscape of Power Management Software

While hardware design dictates the upper bounds on system performance and energy efficiency, the software stack is essential to leverage the features to deliver an optimal solution. Power management implementation in the software can be at the firmware, virtual machine (VM), OS, and application layer based on access to relevant data, response time and control granularity of resources. Rather than trying to survey this huge design space, Table 2.2 examines a few representative examples to illustrate the diversity of existing solutions, with increasing coarseness of control granularity.

Many open source frameworks for power monitoring and control such as global extensible open power manager (GEOPM), PowerAPI and advanced configuration and power interface (ACPI) provide high-level power management interfaces for accessing power knobs [34–36]. ACPI platform exposes platform-independent interfaces for hardware configuration, power monitoring and control, to the OS policies. ACPI specifications define multiple power states, known as global states, device states, processor states and performance states that can be leveraged by the OS to implement power policies.

Software layer	Description and Sample Techniques
Firmware/device drivers	Responsible for exposing hardware performance counters, software-defined events to upper layers of software stack. Software library with APIs to access power control knobs [34–36].
Operating system and hypervisor	Policies and algorithms that leverage firmware capability for dynamic control of power consumption and system resource utilization [37–39]
Application	Power-aware optimization of applications, maximizing performance within set power bounds by adaptive and dynamic reconfiguration [40, 41].

Table 2.2: Classification of software power management solutions.

Hypervisors and OS take specific actions to optimize device power consumption by making dynamic and runtime decisions to selectively power down logic blocks and peripherals based on resource utilization. Several algorithms have been proposed previously to set static and dynamic bounds on application power consumption for energy efficient operation. Based on the system under consideration, two broad principles are employed for energy efficient dynamic control of resource utilization, *(a)* optimize energy consumed by resources under active workload, and *(b)* minimize power consumption under no/low load [37–39].

Static power analysis tools, compilers, programming models and power aware optimization techniques have been developed to minimize power consumption in applications. Compilers optimize instruction scheduling, register assignments, cache access and loop transformations, among other application dependent parameters to gain energy savings. Application power analysis tools and programming models are being developed to deliver energy transparency to the designer, along with established low power coding standards. Additionally, learning algorithms are being adopted to perform usage pattern based power optimization [40, 41].

In a nutshell, abundant number of independent solutions have been proposed and evaluated at various abstraction levels employing the same knobs for power control. In this convoluted software ecosystem, mismanagement and manipulations of power management solutions are likely to cause unpredictable, and potentially catastrophic behavior of a system.

2.3 Power Management Security

Security analysis of power management sub-systems in computing systems is largely unexplored, and is principally limited to traditional test and verification techniques. Existing literature reveals research on power management robustness analysis [42, 43], power grid verification [44, 45], and thermal monitoring [46, 47], to ensure the functionality and reliability of the chip. Although these solutions play an important role in rooting out de-

sign problems, they cannot be easily adopted for runtime security assurance of power management solutions in the MPSoC post-deployment.

- Bembaron et al. explore power aware simulation emphasizing on the use of unified power format (UPF) standard for verification of dynamic power management techniques in low power systems [43]. The research focuses on functional verification of power control hierarchy, power sequence protocols, memory retention behavior and power control signal corruption using static checks and power aware simulation. Dormant trojans inserted in a compromised design space can side-step such pre-fabrication functional tests.

- Kouroussis et al. propose a static technique for power grid verification to ensure that the voltage on the power grid does not drop below a critical threshold. They evaluate the use of pattern independent linear programs to verify a set of designer supplied current constraints [44]. A compromised PMU with control over supply voltage can thwart such pre-deployment verification techniques.

- Francisco et al. investigate the use of an infrared measurement setup in combination with genetic algorithm for non-invasive runtime analysis of power and thermal characteristics of integrated chips [46]. Subsequently, they followed up by characterizing and classifying popular workloads used for architectural design exploration from a thermal standpoint. This work provides valuable insights into the dynamic thermal and power characteristics of real hardware at runtime [47]. However, a compromised PMU can manipulate the characterization process.

- Najeeb et al. are among the first to introduce the concept of power and thermal virus [48], where a set of input vectors are found to inflict maximum dynamic power dissipation in the system under test. Several following works use this concept constructively in peak power estimation, integration testing, thermal testing and product benchmarking [49, 50].

Security analysis and exploits specifically related to the power consumption in low power ICT systems is limited to power signature based side channel analysis and thermal sensor monitoring [51–54]. These works illustrate innovative attack vectors to exploit power signature based security loopholes in low power embedded systems. A common theme in all these works is that the hardware power management infrastructure is considered to be trustworthy.

- Messerges et al. examine the use of simple and differential power analysis to steal sensitive information from smart-cards. They implement power analysis attacks on the data encryption standard (DES) algorithm and demonstrate a multiple-bit attack to extract the DES encryption key in a very short amount of time. Ambrose et al. study power analysis attacks in processor architecture and propose security mechanisms to scramble the power waveform with random instructions and register accesses to mask the power signatures [53].

- Guri et al. demonstrate a technique to bridge the air-gap between two compromised systems in close proximity and render bi-directional communication. Their research reveals that using heat emissions and built-in thermal sensors a covert communication channel can be created to communicate very small amount of data (1-8 bits per hour). Although this attack is limited to systems in close proximity, and allow very slow communication between the compromised systems, they demonstrate innovative and covert attack vectors to exploit security vulnerabilities [52].

- Hutter et al. present a set of temperature related side-channel attacks on a low power micro-controller. Characterizing the relationship between heat dissipation and circuit activity, the research demonstrates data leakage relative to the hamming weight. They inflict a fault attack on the micro-controller by operating the device beyond the specified temperature, and present a technique to exploit the physical property of data remanence attack by applying extensive heat to burn in constant data stored in the microcontroller [54].

Similarly, cloud infrastructure security is a hotbed of research activity. The synthesis of research on power management and data-center security gives rise to alarming new threats posing formidable challenges.

- In high performance data-centers, Wu et al. demonstrates energy attacks targeted solely at abusing server power consumption [55]. The research first profiles the power consumption of victim server, and then leverages knowledge of the client service usage pattern to generate specific requests to increase power consumption by inflicting cache misses. The energy attack increases server power consumption significantly with small increase in service response latency keeping the attack hidden.

- Palmieri et al. unveil a series of denial of service (DoS) attacks that increase the power consumption by exploiting the resources in the data-center [56]. This work details a new perspective to DoS attacks exploiting network bandwidth exhaustion, processing power exhaustion and disk solicitation purely to increase the energy demand of the victim machine.

- Xu et al. investigate power attacks to cause undesired power outages [57]. The research illustrates the generation of power spikes by adjusting workloads, exploiting virtual machine migration and developing specific service usage patterns to trip the electrical circuit breakers in data-center infrastructure. Consequently, the publication briefly discusses a few existing mechanisms that can be adopted to mitigate power attacks.

- Gao et al. explore thermal attacks to increase the server temperatures and degrade performance and reliability of the systems [58]. Their research employs thermal-intensive workloads to create local hotspots in the data-center hosts to inflict cooling failures.

This research spawns a shift from these works by considering a compromised power management unit, further complicating the existing problem. The central focus of this

research is to *uncover the implications of untrustworthy power management in modern computing systems*—an uncharted territory.

2.4 Relevant Hardware Security Research

Security assurance of on-chip hardware components is an obstacle of growing magnitude and importance. The pervasive use of third-party components and IPs has created opportunities for unforeseen security vulnerabilities. Contemporary works have revealed that hardware trojans can be embedded in the third party components used in MPSoCs such as processor cores, on-chip communication platforms and memory modules [59, 60].

Recent works in hardware security include protection against compromised foundries by detecting malicious circuit modifications including hardware trojan detection (e.g., [7, 61–63], among others), techniques to enable secure execution at the microprocessor (e.g., [64–66]), securing off-chip memory access ([67, 68], among others), and protect against counterfeiting through physically unclonable functions (PUF) (e.g., [69,70], among others).

Hu et al. investigate the use of multi-modal thermal and power characterization techniques to detect hardware trojan inserted during the compromised fabrication process [61]. The proposed technique entails the creation of thermal maps using infrared imaging for real chips to extract the spatial power consumption. However, to detect a Trojan, the technique assumes the presence of a golden model without the Trojan for comparison, detection and localization. The lack of golden model for third-party components, and digression in attack manifestation of a trojan in PMU makes power, thermal and other side-channel characterization techniques ineffective.

Ravi et al. argue that in practice, the use of functional security measures alone is far from ideal for security assurance of embedded systems. They echo the concern presented in this work that design complexity and implementation weakness can result in unforeseen security flaws [65]. Their publication surveys a host of attacks, various counter measures for tamper-resistant design assurance. Based on their survey they infer that design of efficient security assurance techniques require a clear understanding of the attacks, as well

as, trade-offs associated with security assurance.

Sifting through existing literature revealed that no previous work investigates threats and attack manifestations from a compromised power management hardware. The unique role of the PMU, combined with the existence of a singular instance in an MPSoC renders a majority of existing security assurance solutions ineffective. Chen et al. propose task duplication to verify the validity of one execution [71]. While their work is an important step forward, it is unable to protect against a malicious PMU, as MPSoCs contain a single instance of PMU rendering their comparison based strategy ineffective.

Besides, contemporary research on providing application security in an unsecured OS setting are comparable, albeit at a higher abstraction level [72–81]). Although a few of these works can be adapted to limit the exposed vulnerabilities, they cannot detect or prevent against malicious modifications in the PMU hardware.

CHAPTER 3

SOURCE OF SECURITY THREAT

Globalization, rapidly changing consumer demands, shrinking product life cycles and fierce competition has changed the dynamics and organization of the computing sector's supply chain. Modern ICT systems have a globally distributed supply chain with both, hardware and software constituting of numerous components that are sourced from a multitude of third-party vendors. In this globalized and collaborative innovation design model, trust and security are fundamental for the seamless integration and interaction between specialized hardware components and extensive software suites of the ICT system. Yet, third-party vendors are averse to expose their internal design to the system integrator to protect their competitive advantage in their niche domains, thereby creating a stiff barrier to many existing trust assurance techniques. As a result, the notion of trust in ICT systems has become a moving target.

This chapter discusses two orthogonal sources of security vulnerabilities that can expose the power management infrastructure to a potentially catastrophic attack. Section 3.1 presents the relevance and typical life-cycle of malicious trojans embedded in the hardware or software modules of the ICT system. Section 3.2 dissects the eventuality of unsuspecting design choices leading to uncharted security flaws. Section 3.3, finally summarizes the trusted and compromised components in threat models considered for this research across two projects discussed in Chapter 4 and Chapter 5.

3.1 Malicious Trojans

In digital computing, trojan is an umbrella term used for covert malware that can be concealed within the legitimate digital design. Simply put, it is a delivery mechanism that an attacker can use to gain access to the system and carry out a myriad of attacks compromising the *confidentiality*, *integrity* and *availability* of the underlying ICT system.

Researchers, academicians and industry experts have explored an overabundance of trojan related security vulnerabilities, metrics and remedies in the software domain [82,83]. Contemporary research clearly demonstrate the presence of trojans in operating system (OS) kernels [84], device driver customization [85], software libraries [86], virtualization techniques [87], and applications [88]. Similarly, over the past decade, a plethora of previous works have established that an adversary can introduce a malicious circuits (hardware trojans) during the design, fabrication and manufacturing of application specific integrated circuits (ASIC) [89–91]. In line with previous studies on trojans, without loss of generality, the following section outlines the sequence of phases to realize a potent trojan in the third-party power management sub-system.

3.1.1 Life Cycle of a Trojan

A typical trojan life-cycle comprises of three distinct phases of activities detailed below.

- ***Trojan Insertion***: Complex designs, lack of sufficient budget, time and technology for rigorous verification, and inadequate supervision of engineers has resulted in an untrustworthy ICT supply chain. A few rogue engineers can insert malicious circuits or backdoors during the design and verification of a system and its applications [92]. Trojans are designed to have low design and activity footprints, so as to camouflage their presence during pre-deployment verification and validation tests. There exist cases of disgruntled employees involved in unprofessional practices, as well as, cases of corporate sabotage resulting in exploitable backdoors [93]. Recently, a key player in the processor sector was accused of using its position to ward-off competition by sabotaging its compiler to use sub-optimal code paths in their competitor's hardware [94]. Additionally, internal attacks are one of the biggest concerns in the IT industry [95]. In software, even if the initial release is malware-free, it can subsequently be tampered by exploiting flaws in the patch management process [96]. The patch payload can contain a trojan along with the legitimate code addressing a real

issue. Since, many updates are now processed online, the source request for patch updates can be redirected to a mirror web address (hijack/spoof attack). Similar models of trojan insertion can be adopted to embed malicious trojans in the power management sub-system.

- ***Trojan Activation***: The attacker can employ a variety of subtle activation mechanisms that can either be triggered by the *internal* or the *external* environment by creating a rare-event trigger [89–91]. Externally activated trojan triggers rely on the interaction with the outside world and hence require a medium to communicate or access data to/from an external environment. Use of antenna, sensors, input/output peripherals, etc. are the common modes of external activation. Similarly, a variety of internal triggers can also be envisaged (e.g., logic/condition based, time based, temperature/voltage based, supply noise based and internal counter based triggers). Another class of trojan trigger involves the coalition of software-hardware, where an application running on the hardware can send a sequence of cryptic messages/requests to activate a dormant trojan in the hardware [97]. Since, third-party components allow limited opportunity for introspection, a smart designer can carefully create an activation module that can elude existing trojan detection techniques. Specifically, in software, existing bugs in the legacy code can also be used to trigger the malware by creating a rare event [98].

- ***Trojan Operation***: Once activated, the trojan can inflict a plethora of attacks on both, hardware components and client applications, adversely affecting the overall system behavior. The type, flavor and potency of an attack on the power management module is only limited by the attackers imagination due to its overreaching influence on the entire system. Broadly, malicious behavior in the power management can lead to data corruption, denial of service, degradation of system performance and energy efficiency, and even result in permanent chip failures due to burnout. Even an uncomplicated attack such as an undesired increase in the supply voltage can cause a surge in the peak power and chip temperature, leading to thermal throttling,

functional errors and systematic chip failure. The specific attacks envisaged in this research, along with their implementation details and impact on system behavior are discussed in Chapter 4 and Chapter 5.

3.2 Security Oblivious Design Artifacts

Burgeoning power management software stack complexity, compounded with multiple specialized hardware-software interfaces, and lack of proper verification methodologies, will together create a *security eventuality*, even in an absence of malicious intent.

Recent discovery of security flaws, *meltdown* and *spectre*, in the processor domain, has underscored the prevalence of design choices that eventually lead to unforeseen and dire security vulnerabilities [99, 100]. These vulnerabilities are an unforeseen by-product of a constitutive performance feature of modern processors—*out-of-order speculative execution*, and do not rely on any other software vulnerabilities. Although both vulnerabilities were first discovered in the year 2018, a vast majority of the processors being used in personal computers, mobile devices, and in the cloud infrastructure from the year 2010 are believed to be affected.

Likewise, in the software domain, even though the security industry has been laser-focused on stamping out bugs and loopholes, major security breaches have been a direct result of software design flaws [95]. Vulnerability and security loopholes in the power management sub-systems are not unheard of. For example, the HP power manager (HPPM) [101] that can monitor, manage and control power environments was previously reported to contain security vulnerabilities that could potentially allow remote attackers to gain access and attack the system [102, 103]. Similarly, a vulnerability was discovered in Intel's power management controller firmware (PMC) that could allow privilege escalation and information disclosure [104].

In alignment to these threats, this research envisages emergence of security flaws in dynamic voltage frequency scaling (DVFS), workload management and consolidation techniques, and power state transitioning, as the feature set and algorithm complexity for

Threat model of a hardware PMU trojan (Figure 1.3)		
Trojan Insertion *Malicious circuit embedded during the hardware design phase as a part of the RTL code, gatelist or a netlist [92].*	**Trojan Activation** *Hardware-software coalition based sequential trigger. Long sequence of power state requests to the PMU are sent by a colluding software.*	**Trojan Operation** *MPSoC system behavior degraded with targeted voltage and power state manipulation attacks on the on-chip components.*

Table 3.1: Life cycle of a typical hardware-centric malicious PMU Trojan.

decision making grows vehemently [105]. Many DVFS mechanisms extend across memory controllers, memory devices, network devices, embedded clusters, among other physical resources [106]. Complex statistical and learning techniques are being quickly adopted for workload management based on energy consumption and prediction [105, 107]. Design flaws in these sub-systems can expose the critical internal power management framework to power attacks, far more potent than any external attacks.

3.3 Summary of Threat Model

Section 3.1 and Section 3.2 together classify the source of threats relevant to our research. Deriving the research problem statement in the backdrop of untrustworthy third-party components in the ICT power management supply chain, this research formulates two cross-layer projects to develop a transformative approach to promote trustworthy computing.

Table 3.1, along with Figure 1.3 summarizes the threat model assumed in the first project discussed in Chapter 4. The project implements and investigates the impact of a malicious trojan circuit embedded in the hardware PMU of an MPSoC during it design and verification phase. The embedded trojan employs a rare-event trigger based on a specific sequence of voltage requests. Once active, the hardware trojan targets critical operations of the on-chip components, by covertly manipulating the voltage assignments and tampering with the power states. Chapter 4 elaborates on the project specifics.

Threat model of a compromised hypervisor (Figure 1.3)		
Trojan Insertion *Malicious code inserted during a software update by exploiting the flaws in the patch management process [96].*	**Trojan Activation** *Trigger distributed between the legacy code and an incoming patch, such that their interaction can produce a rare-event trigger.*	**Trojan Operation** *Data-center characteristics significantly impaired with coordinated attacks on the data-center hardware, and software.*

Table 3.2: Life-cycle of a typical software-centric malicious PMU trojan.

Table 3.2, along with Figure 1.3 summarizes the threat model assumed for the second project detailed in Chapter 5. In the context of a data-center, the project uncovers and evaluates the impact of a software-centric attack, where the power management module of the virtualization suite is assumed to be compromised. A bug in the legacy code is used to trigger the embedded trojan to inflict an internal power attack by manipulating resource allocation and utilization. Chapter 5 delves into the project technicalities.

CHAPTER 4

HARDWARE TROJAN IN PMU

Growing demand for high performance, portable and battery operated devices in automobiles, consumer electronics, industrial and military has resulted in extensive proliferation of specialized hardware components for chip level power management. In fact, the global hardware PMU market accounted for more than 50% of the total analog integrated circuit (IC) revenue in 2012 [5]. Figure 4.1 dissects this growing prominence of PMUs in the analog semiconductor market. Further, owing to the trends discussed in Section 2.1, PMUs are projected to expand at a compound annual growth rate (CAGR) of 4.6% and reach US $56 billion by the year 2026 [108]. In this rapidly evolving sector, *threat evaluation and security assurance in PMU*—a novel contribution of this work—is the need of the hour.

This chapter, based on the work published in design automation conference (DAC) in 2016, presents an end-to-end investigation of security assurance of third-party hardware PMU employed in a MPSoC [23]. Specifically, the chapter answers two critical questions:
(i) How are existing trojan detection techniques limited in detecting emerging threats from a malicious third-party PMU?
(ii) How can we assure security and trustworthiness of third-party PMUs?

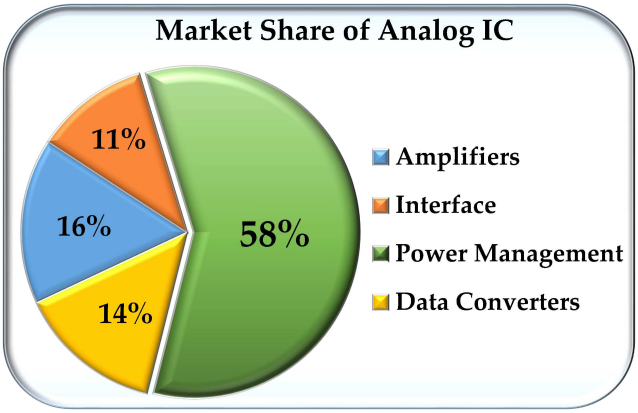


Fig. 4.1: Market share of analog semiconductors ICs in 2012.

Section 4.1 encapsulates shortcomings of existing hardware trojan detection techniques. Section 4.2 details the background of the vulnerable ecosystem. Section 4.3 presents the conceived threat model to embed a hardware trojan in the third-party PMU. Section 4.4 and Section 4.5, elaborate on the design, implementation and operation of two potent attacks evaluated in this project—*P-Virus* and *Drowsy*. Section 4.6 describes the novel non-invasive security assurance technique developed to detect malicious behavior of third-party PMU.

4.1 Limitations of Existing Solutions for PMU Security Assurance

Contemporary research on hardware trojan detection techniques predominantly rely on supplementary verification, replicated execution, testing against a golden model and component redundancy, as seen in Chapter 2 [71, 109–111]. These techniques, however, are impractical in the detection of a malicious PMU broadly due to: *(i)* the singular instance of a PMU in the MPSoC, *(ii)* the nature (analog) of PMU response, and *(iii)* the criticality (response delay constraints) of PMU signals under attack.

The inadequacy of existing solutions for hardware trojan detection in the PMU is discussed next.

- *Limitation of post-silicon tests*: Many previous works examine test pattern generation based on excitation of rare logic conditions [61, 112–118], although none specifically target PMUs. Results in Section 4.3.2 demonstrate that innovative trojan activation techniques in PMUs can easily evade realistic post-silicon tests. The magnitude of trojan activation complexity, time required for an exhaustive state-space exploration and the skyrocketing cost for re-verification of procured third-party components limit the effectiveness of such post-silicon tests.

- *Lack of design transparency*: Side channel analysis and comparison of side channel signatures against a golden reference chip have been widely explored for trojan detection [63, 119–125]. However, third-party vendors are reluctant to reveal their internal RTL design to foster a competitive edge. The lack of a trusted golden model disrupts the effectiveness of side channel techniques. Further, side channel signatures are

susceptible to process and environment variables making them inefficient for trojan detection [126].

- *Limited trust boundary*: Another commonly used design strategy is employment of third-party sensors in a MPSoC circuit to monitor its on-chip environment [127–131]. For example, contemporary voltage droop detectors can, in theory, be modified to detect voltage manipulation attacks from a compromised PMU [132, 133]. However, relying on another third-party vendor design to protect against a third-party PMU trojan is counter-intuitive, as third-party components are assumed to be outside the trusted boundary.

Re-purposing existing solutions from an untrustworthy third-party is futile in the context of considered threat model.

4.2 Threat Environment

Figure 4.2 illustrates a simplified MPSoC consisting of a processor core and other IP blocks representing on-chip memory, communication and accelerators. On-chip IP blocks often have independent voltage domains and are capable of operating at various dynamically changing voltage-frequency levels. Processor cores have dedicated pins to request the PMU to supply a specific voltage level at runtime, as determined by the dynamic

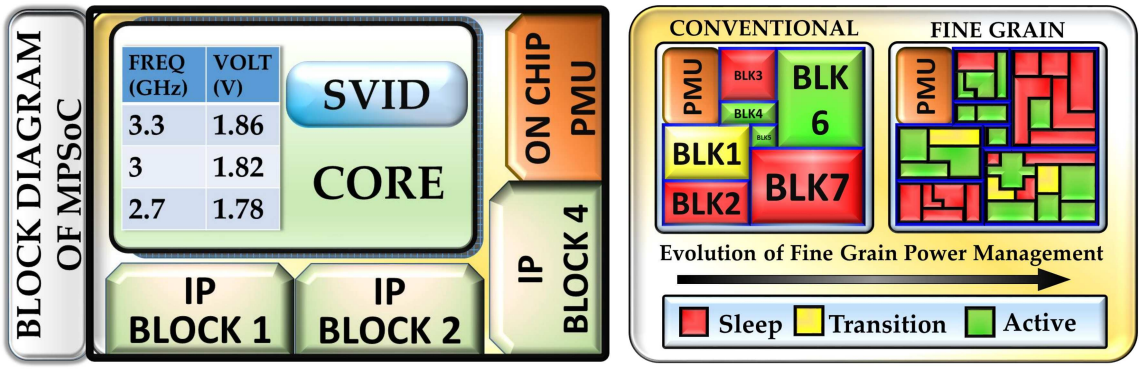


Fig. 4.2: Hardware power management environment of a typical MPSoC.

voltage-frequency scaling (DVFS) control algorithms. Intel's SpeedStep and AMD's PowerNow are two such commercial implementations of DVFS technology. Intel microprocessors have serial voltage identification (SVID) interface to communicate with on-chip and off-chip power management units. Processors send digital voltage identification (VID) signals to the PMU through the SVID interface. PMU decodes a VID signal and supplies the corresponding voltage to the requesting processor through its power rails. For example, in Intel's i7-4650 processor line, the 8-bit ($00h - FFh$) VID signal can request for voltages ranging between 0 to $3.04V$, and the specified operating voltage range of the processor is between $1.64V$ to $1.85V$ [134].

Further architectural dissection of on-chip components in Figure 4.2 reveal an emerging trend in the control granularity of dynamic power management (DPM) events. Compared to conventional independent power domains, emerging architectures have control grains that are an order of magnitude smaller [6]. In densely integrated modern MPSoCs, the number of independently controllable blocks rise to hundreds, thereby increasing the complexity and latency involved for run-time monitoring of individual blocks. DPM techniques involve selectively turning off unused components, based on speculated idle time and history of execution patterns. In conventional architectures, the operating system (OS) is responsible for idle state management of blocks. To achieve a fine grain control in emerging architectures, a layer of hardware control in the form of power management IP (shown as PMU in Figure 4.2) is introduced, in addition to the OS control.

Niche and complex hardware PMU designed for dynamic control of independent voltage domains, along with optimal management of thousands of fine grain power events in a MPSoC serves as the vulnerable environment in this work.

4.3 Threat Model

In line with the life-cycle of a trojan discussed in Section 3.1.1, this Section elaborates on the particulars of threat in modern MPSoCs.

4.3.1 *Trojan Insertion* (Threat Origin)

A few key engineers on the third-party PMU team insert a malicious circuit during the design or test phase [92]. The trojan is designed to remain dormant to elude detection during verification, and to have a low design and operational foot print.

4.3.2 *Trojan Activation* (Threat Concealment)

The activation phase is particularly important to a trojan design. Careful choice of a rare event trigger can conceal the trojan during pre-deployment testing of the chip. This work uses a software-hardware coalition based sequential trigger that takes advantage of voltage change requests made by on-chip components. Analogous to the idea presented by Waksman et al. (*sequence cheat code*) [135], an attacker uses a sequence of power event requests such as specific VID patterns to trigger the underlying trojan. An unsuspecting software is designed to send a pre-defined sequence of voltage change requests in the form of VID signals to the malicious PMU. The embedded trojan in the third-party PMU monitors these staggered incoming requests for the complex sequence. In modern processors, VID signals are typically 6 to 8 bits and a sequence of these requests can potentially have an enormous state space resulting in astronomic test times.

Table 4.1 reveals the magnitude of complexity involved in *guaranteed* trojan activation during testing using the technique employed by Chakraborty et al. [112]. PMUs are tested by applying test patterns (VID), and observing its resulting voltage levels. However, each test pattern can be applied after a short delay associated with the voltage stabilization latency. Taking into account the voltage level stabilization time (between 0.1 to a few milliseconds [136]), Table 4.1 evaluates time required in years for guaranteed trojan activation caused by a specific sequence of VID signals (sequence length of 4 to 64 signals). Table 4.1 also illustrates area and power overhead of the trojan activation design, along with its trigger probability. Observed data shows extremely low probability of activating a trojan in the malicious PMU during post-silicon tests due to the enormous state space exploration time associated with exhaustive tests.

SL	Area	Power	TP	Time (yrs)
4	0.16%	0.07%	2.3×10^{-10}	6.8×10^{-2}
8	0.38%	0.17%	5.4×10^{-20}	2.9×10^{8}
16	0.87%	0.40%	2.9×10^{-39}	5.4×10^{27}
32	1.80%	0.99%	8.6×10^{-78}	1.8×10^{66}
64	3.40%	2.10%	7.4×10^{-155}	2.1×10^{143}

Table 4.1: Limitation of post-silicon testing

4.3.3 *Trojan Operation* (Threat execution)

Once activated, a trojan can potentially inflict a plethora of attacks on on-chip components, adversely affecting the overall system behavior. This work demonstrates two specific attack vectors that cause substantial degradation in energy efficiency and performance. Inimical attacks are carried out in a covert manner making it difficult to pinpoint the source of an attack.

P-Virus (Section 4.4): The embedded trojan in the PMU tampers with operating voltage requests from a processor core. Consequently, the PMU delivers a higher voltage than what a processor requests, leading to improper voltage-frequency assignments. A surge in peak power leads to thermal throttling, causing reduction in operating frequency. *P-Virus* primarily affects the energy efficiency with a potential degradation in performance due to thermal throttling. The attack mimics symptoms of a virus in humans such as increase in operational temperature and reduction in performance.

Drowsy (Section 4.5): The malicious PMU furtively delays wake-up requests of on-chip components, and at times, abruptly forces blocks to sleep. The attack mimics effects of drowsiness, affecting the availability of resources. Latency sensitive applications suffer from severe performance degradation due to an extraneous delay in resumption from sleep state, and undesired power state transitions. If a component forces an access to a block under transition, the MPSoC may encounter functional errors.

4.4 P-Virus

Three variants of *P-VIRUS* are conceived based on manipulation of voltage requests

and its resulting behavior. In jest, these three variants are named in relation to effects a virus has on humans.

- *Spasm* ($V_{supply} < V_{request}$): Voltage supplied by the compromised PMU is less than the requested voltage. Processor circuits cannot meet timing constraints for the chosen frequency due to the higher delay associated with lower supply voltage, leading to functional errors that manifest as glitches in the system.

- *Fever* ($V_{supply} > V_{request}$): Supply voltage is higher than the requested voltage. A higher supply voltage will result in loss of energy efficiency, and increase the chip temperature, similar to an onset of fever in humans. Subtle manipulation of supply voltage can also help keep an attack stealthy. To compensate for elevated temperature, the system may lower its operating frequency, thereby degrading the processor performance.

- *Seizure* ($V_{supply} >> V_{request}$): Supply voltage is substantially higher than the requested voltage. Excess heat dissipation due to a sudden increase in the power consumption can lead to the breach of a processor's TDP resulting in catastrophic failure and chip burnout, mimicking a seizure.

This work models and evaluates the *Fever* variant of *P-Virus* to show that even modest voltage manipulations affect the system characteristics significantly.

4.4.1 P-VIRUS Implementation

Figure 4.3 shows the conceptual block diagram of *P-Virus* attack. Processor core requests the PMU for a voltage determined by the DVFS algorithm by sending a 8-bit digital signal through the dedicated SVID interface. Hardware trojan embedded in the PMU manipulates this request and misguides the voltage regulator to deliver a voltage higher than the requested value. Other on-chip IP blocks are oblivious of this attack on the processor, and function within their set margins.

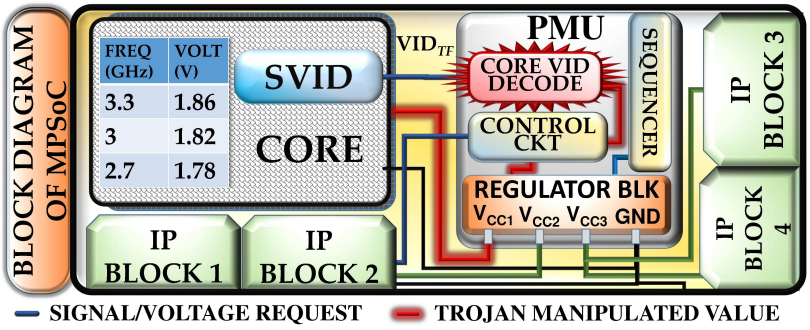


Fig. 4.3: Conceptual block diagram of a *P-VIRUS* attack. The PMU corrupts the voltage request sending a higher operating voltage to the processor.

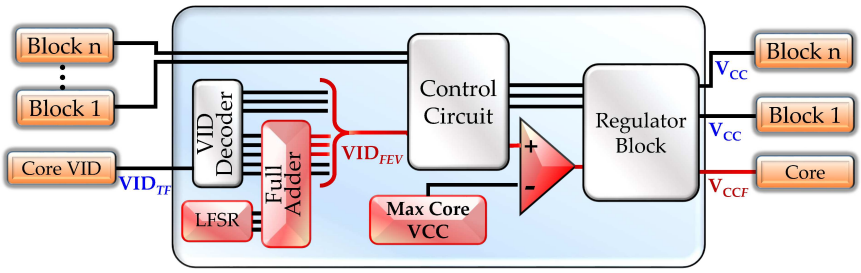


Fig. 4.4: Implementation of *P-VIRUS* attack. Blocks in red illustrate the trojan behavior.

Figure 4.4 shows the implementation of *P-Virus*. The VID decoder block deciphers an incoming VID signal and the trojan manipulates this VID signal. To keep an attack stealthy, the degree of attack is constrained by altering only 3 bits (4, 5 & 6) of VID in a random fashion using a linear feedback shift register (LFSR). This also helps reduce the trojan footprint, and hence, keep the attack concealed. The infected VID (VID_{FEV}) request is forwarded to the control circuit that manages requests from many on-chip components and instructs the regulator to scale the voltage. The regulator outputs the scaled voltage to power rails corresponding to the block that requested the voltage change. *P-Virus* infected supply voltage value (V_{CC_F}) corresponding to the VID_{FEV} is constrained within the *Max. Core* V_{CC} [1]. The set constraint, along with randomness of voltage manipulation obscures *P-Virus* attacks, while ensuring that manipulations are significant enough to cause considerable degradation in system behavior.

Table 4.2 gives an example of VID manipulation for an incoming request of $1.65V$.

[1] Max. Core V_{CC} is the voltage beyond which permanent damage to the processor core is likely.

Req. V_{CC}	**VID_{TF}**	**VID_{FEV}**	**V_{CC_F}**
1.65V	*74*	*7C - AC*	1.73V - **2.02V**
1.8V	*87*	*8B - BB*	1.88V - **2.02V**

Table 4.2: VID manipulation example. Req. V_{CC} is the voltage requested by the core by sending VID_{TF} (Hex). LFSR generates random values between *001 to 111* and manipulates the VID_{TF} to a value in the range of VID_{FEV} (Hex). V_{CC_F} is the range of infected voltage supplied to the core.

Req. V_{CC} is the voltage requested by the core by sending VID_{TF} (trojan free hex value). A 3-bit LFSR generates a random number between *001 to 111*, to inflict an inconsistent attack and circumvent the generation of identifiable trojan attack patterns in system behavior. On adding it to VID_{TF}, a range of possible VID_{FEV} (*Fever* attack inflicted hex value) is determined. Note that the V_{CC_F} is limited to a maximum of 2.02V (*Max.Core* V_{CC}) though the VID_{FEV} corresponds to a voltage range of 1.73V to 2.21V. V_{CC_F} (trojan infected supply voltage) is the range of infected voltage supplied to the core.

4.4.2 P-Virus Evaluation

The impact of *P-Virus* is evaluated using Sniper6.0 multi-core simulation infrastructure [137]. An Intel i7-4650U processor is modeled with specifications detailed in Table 4.3 and the trojan behavior is designed in it. To simulate the effect of *P-Virus*, operating voltages of different processor performance-states (p-states) are manipulated, while running Splash2 benchmark applications. A widely-used *on-demand* DVFS algorithm is used for p-state assignment [138]. Ondemand DVFS algorithm dynamically controls the processor to achieve maximum clock frequency when system load is high, and also minimum clock

Parameters	**Processor Configuration**
Intel	i7-4650U
Technology node	22nm
Cores	1 - 2
CPU Type	Out-of-Order Engine (OoO)
Frequencies	(1.7, 2.1, 2.4, 2.7, 3.0, 3.3) GHz
Voltages	(1.6,1.65, 1.70,1.75, 1.80,1.84) V

Table 4.3: Configuration Parameters used to model *P-Virus* and *Drowsy*.

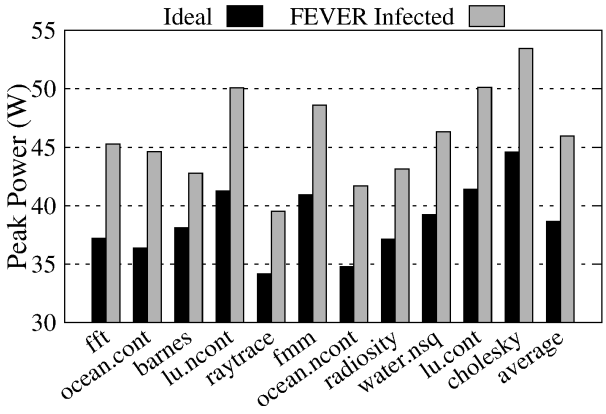


Fig. 4.5: Effect of P-VIRUS on peak power. Peak power increases as a result of increased operating voltage. Higher value represents a more potent attack.

frequency when the system is idle. A peak thermal design power (TDP) is set, which represents the maximum heat that is allowed to be generated by a processor during operation. Frequency is throttled down when the dynamic power (thermal load) rises beyond the set TDP, to shun assignments that can lead to chip burn-out or system failure.

The potency of *P-VIRUS* is evaluated using 3 integral metrics: peak power, energy efficiency and performance. Figure 4.5 shows the comparison of peak power between *P-Virus* free and *P-Virus* infected executions. The x-axis represents chosen real-world Splash2 workloads, and the y-axis denotes the absolute value of peak power obtained from simulations. On one particular execution, the average peak power increase across chosen workloads was 19%, with *ocean.cont* incurring the highest increase in peak power (23%). Since the attack is designed with capabilities to inflict random increases in voltage, subsequent runs had varying impact on workloads. Rise in peak power is limited by the thermal design power (TDP), beyond which operating frequency of the processor is throttled down to prevent a thermal failure.

Figure 4.6 shows the impact of *P-VIRUS* on energy consumed. The y-axis denotes the increase in energy consumption over the course of workload execution. On an average, energy increases by 18%, with *fft* incurring the highest increase in energy consumption (26%). Degradation in system performance is a resultant of thermal throttling in the pro-

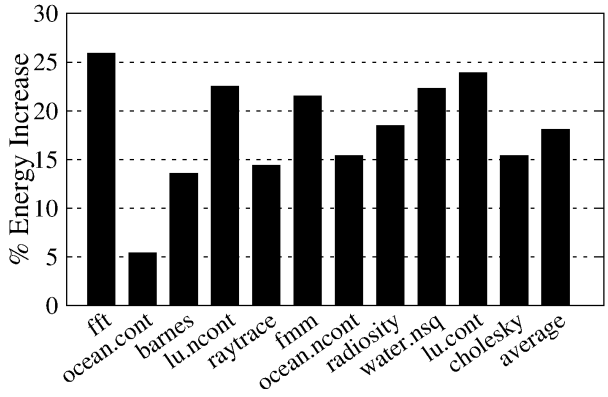


Fig. 4.6: Effect of *P-VIRUS* on energy. Energy increases due to the increased power and degradation in application performance. Higher value represents a more potent attack.

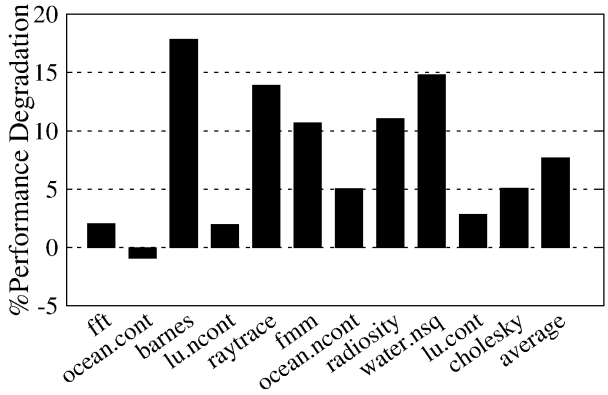


Fig. 4.7: Effect of *P-VIRUS* on application performance. Performance degrades due to the thermal throttling to ensure chip safety. Higher value represents a more potent attack.

cessor. In Figure 4.6, *ocean.cont* workload appears to be an anomaly incurring a meager 5% increase in energy, even when the peak power increases by 23%. This can be explained by analyzing energy increase in conjunction with performance results in Figure 4.7.

Figure 4.7 reveals that *ocean.cont* did not incur any performance degradation. For this benchmark, the processor is able to operate at a higher voltage without breaching the set TDP, implying that attack manifestation also depends on workload characteristics. Loss in energy efficiency is purely due to an increase in power consumption. For the other benchmarks, Figure 4.7 shows the degradation in processor performance due to thermal

throttling (up to 26%). Overall, the trojan degrades system behavior in all three measured metrics.

To evaluate the footprint of *P-Virus*, PMU RTL of the OpenSPARC T2 processor is modified [139]. Logic discussed in Section 4.4.1 is implemented, and synthesized with the TSMC 45*nm* library using Synopsys Design Compiler. *P-Virus* incurs area and power overheads of 1.39% and 1.06%, respectively.

4.5 Drowsy

Drowsy attack tampers with sleep and wake up requests of various on-chip components. The attack mimics drowsiness, affecting the availability of on-chip resources. Three incarnations of the *DROWSY* attack are conceived:

- *Delayed Sleep:* Compromised PMU delays sleep signals, allowing components to be active even when unused, thereby degrading energy efficiency of the system.
- *Delayed Wake up:* Delay in wake up can result in resource unavailability. Latency sensitive applications will suffer from performance degradation.
- *Abrupt Sleep:* Abrupt transition of blocks to sleep states results in loss of data and performance degradation.

This work models and demonstrates the *delayed wake up* scenario of *Drowsy* attack.

4.5.1 Drowsy Implementation

Figure 4.2 shows an evolving trend in the power management granularity, to enable precise control of independent functional logic, much smaller than conventional clock and power domains. When on-chip resources are idle, they are put to sleep to save power. Breaking down on-chip components into smaller sections for independent control allows for greater reduction in power.

Figure 4.8 shows the operation of *Drowsy* attack. Control circuit executes decisions of the PMU and puts the corresponding block to sleep state. When the resource is scheduled

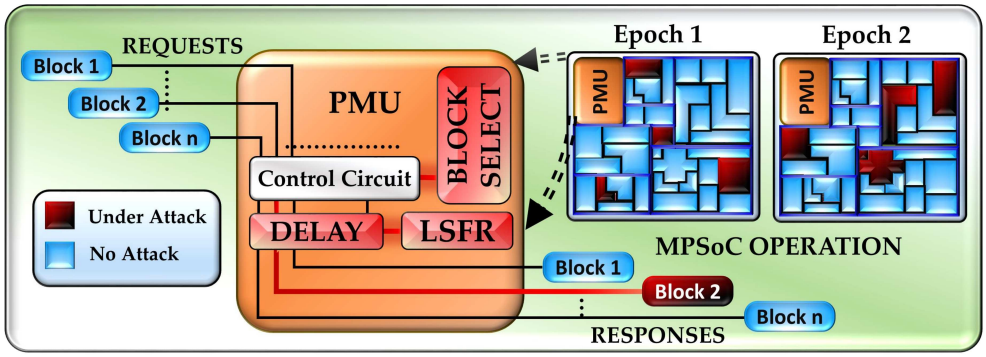


Fig. 4.8: Implementation details of *Drowsy* attack. DROWSY attack is random and focused, where the blocks under attack vary at different epochs to keep the attack stealthy.

for operation, trojan in the PMU delays the response to this request, adversely affecting application performance. During an epoch, one or many blocks may send wake-up requests to the control circuit in the PMU. A *block select* module randomly chooses a subset of blocks to inflict a sluggish response. Response to these blocks are delayed by a random, but bounded time, using the delay block fed by a LFSR. Randomness in attack, constraint in delay threshold, and incoherent focus on a subset of blocks, obscures *Drowsy* attack. In summary, *Drowsy* is a random but focused attack, where blocks under attack vary at different epochs, thereby keeping the attack stealthy.

4.5.2 Drowsy Evaluation

Drowsy attack is modeled by re-purposing the cost associated with sleep periods and manipulating the latency whenever a block resumes from a sleep state in Sniper6.0 multi-core simulator using Splash2 workloads.

Figure 4.9 shows the performance degradation due to *Drowsy*. Sluggish responses to wake-up requests increase the number of idle cycles, resulting in a maximum performance degradation of 59% (average across benchmarks is 34%). Blocks in sleep or those under transition cannot be accessed, thereby preventing any functional errors.Energy overhead is negligible, as these blocks consume minimal power during sleep.

Drowsy incurs an overhead of 1.84% in area and 1.92% in power.

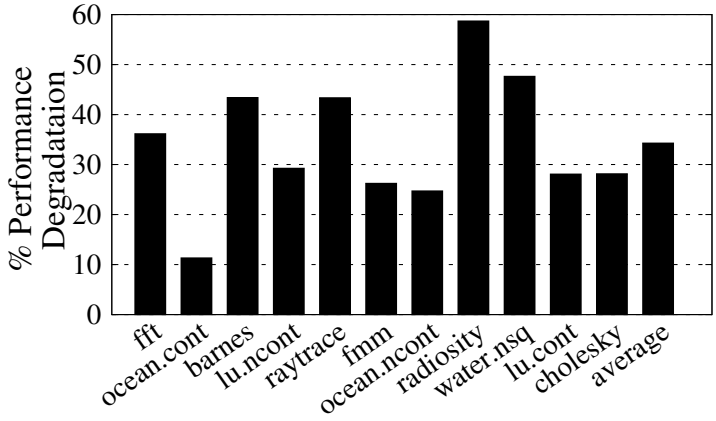


Fig. 4.9: Performance degradation due to *Drowsy*.

4.6 Malicious PMU Detection

Security assurance in these complex PMUs is the need of the hour due to their impact on the entire chip behavior. However, design of techniques to guarantee trust and security are plagued by following challenges:

- *Overcome 3PIP design constraints*: Lack of design transparency and the inability to verify 3PIP design internals introduce security bottlenecks.
- *Understand attack semantics*: Morphological differences in manifestation and potency of a trojan varies significantly with each type of attack.
- *Component agnostic solution*: Diversity in design characteristics among various on-chip components presents a challenge in creating a generic solution.
- *Manage overheads*: Constraint on chip size and power consumption requires the minimization of circuit-architectural overheads.

To surmount these design challenges, this work proposes a novel **I***ntellectual* **P***roperty* **R***isk* **A***ssessment* **M***odule* (IPRAM), ***designed by the MPSoC integrator and placed at interface of critical 3PIP blocks***.

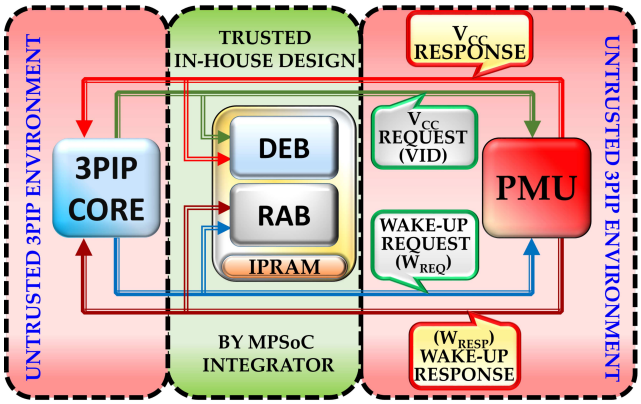


Fig. 4.10: Conceptual overview of the IPRAM. The IPRAM is designed in-house by the MPSoC integrator and placed at the interface of a 3PIP block to be monitored.

4.6.1 IPRAM

Figure 4.10 shows the block diagram of the proposed IPRAM. IPRAM assumes no support from a third-party vendor and effectively detects malicious activities in PMUs, across various vendors. All outgoing power management requests from third-party IPs are in parallel sent to the IPRAM, and corresponding responses from the PMU are forwarded to the IPRAM too. Since IPRAM is placed in parallel to the communication between a third-party IP and the PMU, it does not add to the latency of power management state changes. At its foundation, IPRAM sits at the interface (boundary) between trusted and untrusted zone, and matches all requests to responses to identify the presence of malicious activity.

In this project, IPRAM consists of two low complexity innovative blocks specifically designed to identify all malicious power management behavior.: (a) *P-Virus* monitor (Section 4.6.2) and (b) Wake up alarm (Section 4.6.3).

4.6.2 P-Virus Monitor

P-Virus Monitor (PM) is designed based on the insight that *supply voltage (V_{cc}) of a digital circuit profoundly influences its delay*. By characterizing the delay of a known circuit, supply voltage imposed on that block can be estimated.

Figure 4.10 presents the interface of *PM*, modeled as a **d**elay **e**stimation **b**lock (*DEB*)

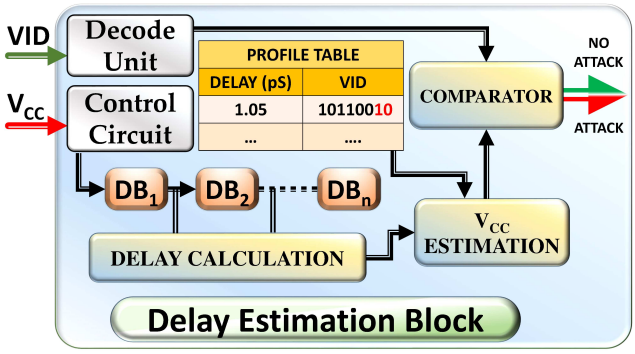


Fig. 4.11: Design for the detection of *P-Virus*.

fed by the same supply line as that of the requesting IP block. Figure 4.11 shows a detailed operation of *DEB*. The control unit power gates the *DEB* when not in use, to curtail the power consumption overhead. A series of cascaded delay buffers (DB) in the form of a tunable replica circuit are used to estimate the parasitic delay of the circuit [140]. These cascaded buffers are sampled at equal epochs to capture state transitions at different stages and thereby estimate the delay for an imposed V_{cc}. The estimated delay is correlated to values stored in the look up table to obtain **VID** corresponding to the delay. The look up table is filled, during post-silicon characterization tests. The identified **VID** is then matched to the **VID** request in comparator unit, by ignoring 2 least significant bits to grant a margin for error. If values do not match, the *DEB* flags as anomalous behavior.

4.6.3 Wake Up Alarm

Wake Up Alarm (WA) is a finite state machine (FSM) to observe arrival of requests and responses of power state transitions. Delayed and unprompted state changes triggered by the malicious PMU can be detected.

WA is modeled as a **r**esponse **a**udit **b**lock (*RAB*) in Figure 4.10, where a request sent to the PMU is also sent in parallel to the *RAB*. Figure 4.12 shows a detailed block diagram of *RAB*. MPSoC blocks usually have multiple sleep states, varying in state transition times and power saving capabilities. The control circuit in the *RAB* feeds a multiplexer to select the appropriate delay associated with each sleep-to-wake transition. This delay is imposed

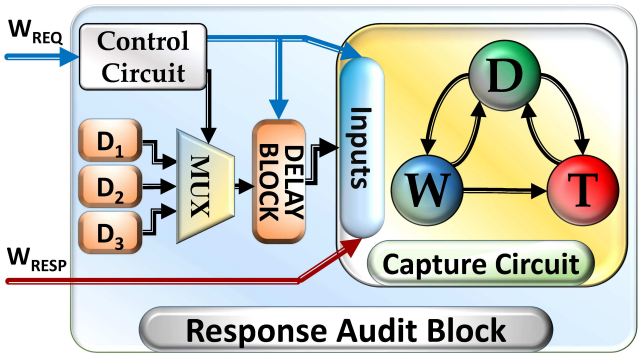


Fig. 4.12: Design for the detection of *DROWSY*.

on the request and forwarded to the *capture* circuit, that consists of a FSM with 3 states, *default* (*D*), *wait* (*W*) and *anomaly detected* (*T*). The FSM is in *D* state until it detects a signal from the 3PIP block under test. State transitions in *RAB* are discussed below.

- $\boldsymbol{D \rightarrow W}$: On receiving a wake-up request from a third-party IP block, the state changes from *D* to *W*.
- $\boldsymbol{W \rightarrow D}$: If the PMU response is observed within the vendor specified response time, it triggers a transition from *W* to *D*. Third party IP continues its operation.
- $\boldsymbol{W \rightarrow T}$: If the PMU response is delayed, the delay block triggers a transition from *W* to *T*, flagging the anomalous behavior— a *delayed wake up* attack.
- $\boldsymbol{T \rightarrow D}$: Once an attack is flagged, all states of monitor blocks are reset, and the request-response combination for the block under test is quarantined for analysis. The FSM state transitions to the default (*D*) to wait for the next arriving request.
- $\boldsymbol{D \rightarrow T}$: An unsolicited shutdown/wake-up signal from the PMU, is captured and flagged— an *abrupt* attack.

4.6.4 IPRAM Analysis Methodology

This section summarizes the methodology used to evaluate the efficacy and overhead of IPRAM.

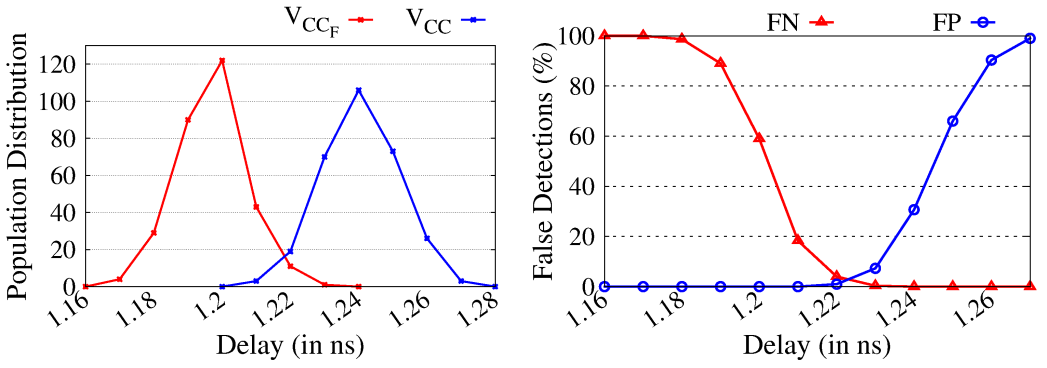
PM: To evaluate the efficacy of *PM*, cascaded buffers present in the *DEB* are modeled using HSPICE and delay characteristics are obtained for different V_{CC} values. To account for process variation, a Monte Carlo analysis is conducted by creating a Gaussian distribution of three parameters: threshold voltage, effective channel length and transistor width [141].

WA: *WA* identifies *Drowsy* attack via a FSM, based on incoming signals. *WA* is evaluated by implementing the proposed *RAB* design as a Verilog RTL model using Xilinx ISIM. Exhaustive tests are performed to observe its behavior for different scenarios.

To find the design overheads of the proposed security assurance solutions, blocks within the IPRAM are implemented in Verilog RTL and synthesized with the TSMC 45*nm* library using Synopsys Design Compiler to find their design overheads.

4.6.5 IPRAM Evaluation

PM: Figure 4.13 and Figure 4.14 together illustrates two representative cases of *PM*. In Figure 4.13a, the worst case scenario for trojan detection is demonstrated, where the difference between V_{CC_F} (1.73V) and V_{CC} (1.65V) is a mere 0.08V. V_{CC_F} is the *P-Virus* inflicted voltage. This is the lowest increment in voltage that the *P-Virus* can impose (Section 4.4.1). The overlap in distribution implies that these delay values can occur both in the presence and absence of a *P-Virus* attack, leading to an inaccuracy in trojan detection (false detec-

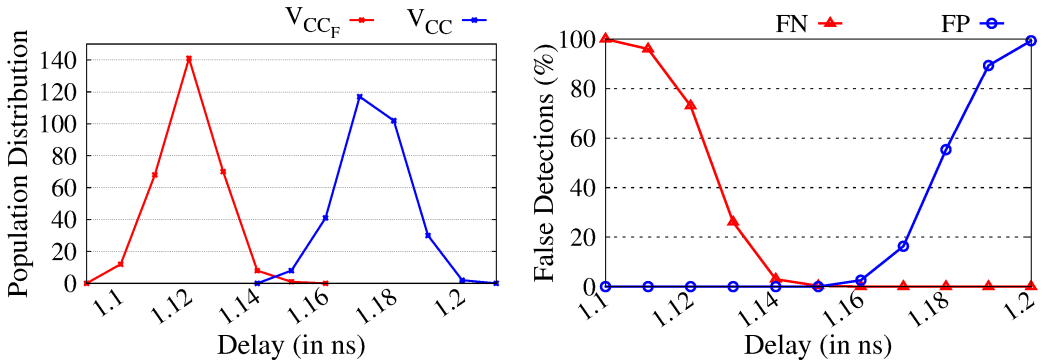


(a) Worst case

(b) FP and FN analysis

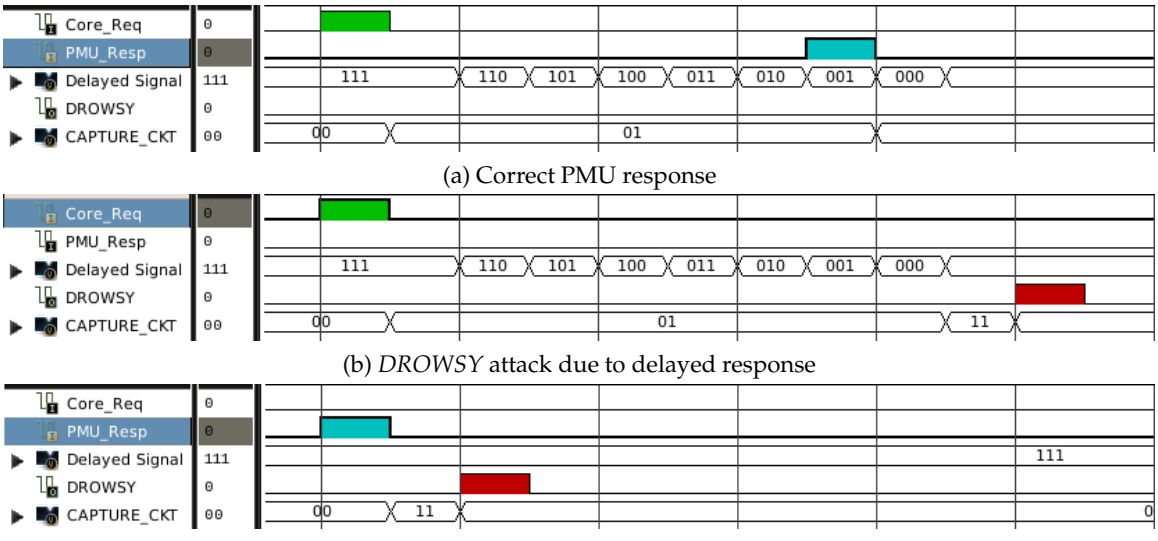
Fig. 4.13: Results from *PM*. Worst case representative of the delay characteristics for *P-VIRUS* detection.

tion). Figure 4.14a represents a typical case, where *P-Virus* increases the voltage by $0.1V$. For attacks with a voltage increment greater than $0.1V$, distribution curves are distinct (no overlap), implying that there exists an optimum delay threshold with zero false detection. **Threshold Selection**: Delay threshold indicates the minimum bound on the delay that can occur for a chosen voltage. If a voltage higher than the requested value is supplied, the obtained delay value will ideally be below the threshold. Smart threshold selection helps reduce the number of false positives (FP) and false negatives (FN). Figure 4.13b, shows the FP and FN analysis of the worst case. With an optimum threshold of $1.22ns$, FP rate is a



(a) Typical case (b) FP and FN analysis

Fig. 4.14: Results from *PM*. Typical case of the delay characteristics for *P-VIRUS* detection.



(a) Correct PMU response

(b) *DROWSY* attack due to delayed response

(c) *DROWSY attack by an unsolicited signal*

Fig. 4.15: Important scenarios of *WA*. Figure 4.15a illustrates the correct behavior. Figure 4.15b shows a *delayed response* attack detection. Figure 4.15c detects unsolicited response.

mere 1% and FN rate is 4%, giving a false detection rate (FDR) of 5%. As delay moves away from the optimum threshold, the FDR increases rapidly. For a typical case, the tolerance for error in optimum threshold selection increases (Figure 4.14b) as the FDR remains less than 3% for a range of delay thresholds (1.14*ns* to 1.16*ns*). Meticulous profiling during the testing stage helps set accurate thresholds to reduce the FDR.

WA: Figure 4.15 shows three important cases of *Drowsy* detection. States *00*, *01* and *11* indicate states *D*, *W* and *T*, respectively. Figure 4.15b illustrates an attack scenario. A 3PIP core sends a wake-up request, but the response is delayed by the malicious PMU. *WA* flags the trojan, once the vendor specified time expires without a PMU response.

4.6.6 Implementation Overhead

PM and *WA* incurs area overheads of 0.97% and 0.14%, respectively. Power overheads are negligible, as blocks in the IPRAM are power gated when not in use.

CHAPTER 5

HYPERVISOR POWER ATTACKS

Modern data-centers employ complex and specialized power management strategies in pursuit of energy and thermal efficiency. Computing power management software suites have swelled to a net size of $314 million, with a growth of 35.48% during 2012-2016 [3]. These software suites promise to deliver $18.6 billion/year in cost reduction as a direct consequence of improved energy efficiency [4]. Interestingly, innovation in power and energy management strategy is intertwined with increasing complexity, and consequently, exposes a new attack surface in an already vulnerable cloud ecosystem. Deliberate attacks on complex power management can leave data-centers crippled, often resulting in severe loss of productivity, economy and reputation. With more than 58% of the enterprises spending at least 10% of their annual budget on cloud services, the need for power management security assurance is imminent [142].

This chapter, based on the work published in the journal of hardware and systems security in 2019, presents a novel secure framework created to combat the dangers posed by a compromised power management module in a data-center. The proposed secure framework leverages a learning algorithm, along with the long-standing concepts of system isolation and resource redundancy to safeguard against harmful power attacks in the cloud ecosystem. Specifically, this work investigates two critical questions.

(i) What are the repercussions of a security loophole exposed by a compromised power management module in the data-center?

(ii) How can data-centers be safeguarded from malicious power attacks?

Section 5.1 briefly introduces the design of a typical data-center, and Section 5.2 emphasizes on the source and life-cycle of a vulnerability in the system. Section 5.3 delves into attack specifics, exploiting the exposed security vulnerability, and presents an evaluation of the attack. Section 5.4 explains the novel secure framework proposed in this work,

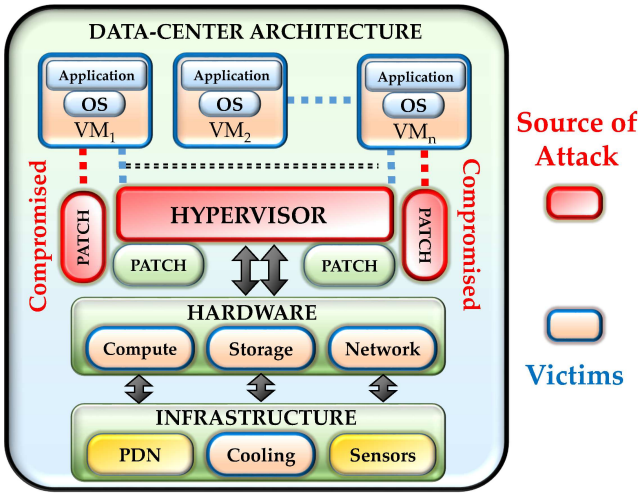


Fig. 5.1: Data-center illustration of the threat model.

to safeguard data-center from power attacks, and presents a structured assessment of the framework's efficacy.

5.1 Threat Environment

Figure 5.1 presents a simplified illustration of a typical data-center architecture. Data-centers consist of a distributed and multi-layered software stack including a hypervisor, multiple virtual machines (VM) and guest OSes, device driver modules, and other service infrastructure. At the heart of data-center operation is the hypervisor, also known as, virtual machine monitor (VMM). It manages the underlying compute, network and storage hardware resources and allocates required resources to subscribing applications. This process of seamless and efficient sharing of hardware among multiple applications/operating systems is key to energy proportional computing. Consequently, the hypervisor handles a lions share of power and energy management responsibilities, as it has direct access to arriving applications, as well as, the underlying hardware. By inspecting internal data structures, hypervisor obtains coarse-grained per-VM information on how energy is spent on the hardware, and efficiently manages resources by throttling frequency, migrating VMs and consolidating applications based on desired system performance.

Power consumption in a data-center can be attributed to various system components

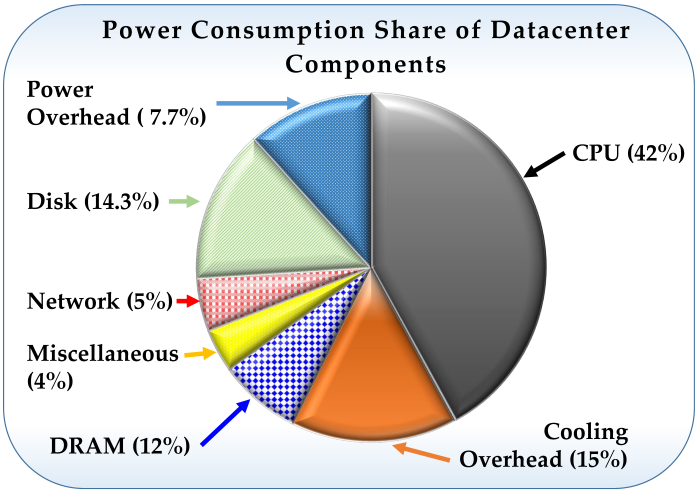


Fig. 5.2: Power consumption breakdown of a typical high performance data-center system.

such as, processors, storage disks, network elements, memory and a cooling system. Figure 5.2 illustrates a typical breakdown of power consumption among various system components. Section 2.2 revealed the existence of abundant research in both, power management schemes for individual components, as well as, synergistic cross-layer strategies. This work explores the implications of security vulnerabilities in this complex eco-system.

5.2 Threat Relevance

Inferring from the generic life-cycle of a trojan (Section 3.1.1), this section formulates the minutiae that lead to the exposure of security vulnerability.

5.2.1 Origin of an Attack

In line with the discussion in Chapter 3, two orthogonal avenues lead to security vulnerabilities in the power management module of the hypervisor:

(a) Security oblivious design artifacts, and

(b) Insider attacks.

In this premise, the power management software stack complexity, compounded with multiple specialized hardware-software interfaces, and lack of proper verification methodologies, will together create a *security eventuality*. While the security industry has been

laser-focused on stamping out bugs and loopholes, major software security breaches have been a direct result of software design flaws [95].

A malicious employee inserts bugs or backdoors in the legacy code of a third-party hypervisor that an attacker later capitalizes on. Here, the attack surface is initially either undiscovered or non-existent. At a later stage, the backdoor is exploited to inject a trojan as a part of legitimate software patch/update, and interact with the legacy bug. Several design and implementation flaws plague the patch management process [96]. Exploiting these flaws, malicious code is inserted into power management module of the hypervisor. The patch payload contains a trojan along with the legitimate code addressing a real issue.

On the contrary, since all updates are now processed online, the source request for patch updates can be redirected to a mirror web address (hijack/spoof attack) by the data-center administrators too. Internal attacks are one of the biggest concerns in the IT industry [143].

5.2.2 Threat Activation

Once a patch is successfully applied, several health checking tasks, as well as, regression tests are performed to ensure that an update does not adversely affect the existing system. To elude detection, a bug in the legacy code is exploited to create a rare event to trigger the trojan after the testing phase.

5.2.3 Operation

Figure 5.1 shows that a compromised power management module in the hypervisor can, in theory inflict a threefold attack on the data-center architecture.

- First, it can adversely affect the demand response to application requests by manipulating the virtual machine (VM) management algorithms (*attack on the software*).
- Second, it can impact the application performance by manipulating the power states of the underlying compute, network and storage hardware (*attack on the hardware*).

- Third, it can cripple the data-center by mismanaging the cooling resources. (*attack on the infrastructure*).

This work implements and evaluates the first two attack vectors. The attack details are discussed in Section 5.3.

5.3 HyperAttack - Power Attacks by Compromised Hypervisor

Hypervisor forms the core of data-center power management and hence, has direct access to power management techniques employed at different abstraction levels. In this work, *HyperAttack* epitomizes a new class of potent power attacks realized by compromised power management modules in a hypervisor. A malicious trojan inflicts a coordinated attack across both the hardware and software power management modules. The operational details are discussed below.

Hardware: Hypervisor governs hardware power and sleep states based on the resource utilization. To inflict an attack in processor cores, a compromised module manipulates the process of frequency and voltage assignment to increase the power consumption, even at low utilization. For example, to retain stealth in the attack, the compromised module increases computation core frequency/voltage in epochs surrounding a virtual machine (VM) migration and obscures the increase in power consumption. In addition, if the utilization increases due to application demand, the hypervisor restricts access to available computation power within each core and distributes workloads across multiple physical nodes. This results in an increased power consumption, with minimal impact on application performance.

Software: Hypervisor is responsible for creation and allocation of VMs based on application subscription. In power aware VM allocation algorithms, the hypervisor aggressively tries to consolidate requests on to a minimum number of physical servers so as to power down all unused resources and reduce the overall power consumption. The compromised module manipulates the utilization threshold that determines the allocation of new VMs on each physical server. Wasteful and unsystematic distribution of VMs leads to a large

Parameters	Data-center Configuration	
Physical Nodes	800×HP Proliant	
	400×G4	400×G5
CPU	Intel Xeon 3040	Intel Xeon 3075
$\#\ of cores$	2	2
$Frequency$	$1860MHz$	$2660MHz$
RAM	$4GB$	$4GB$
Hypervisor	*Xen*	

Table 5.1: Data-center configuration parameters.

number of physical servers being under utilized. To further increase the potency of an attack, the compromised module infects VM migration algorithms. While choosing the VM to migrate, the module calculates the migration time required based on the random access memory (RAM) utilized and the spare network bandwidth. The compromised module then migrates the VM that utilizes the maximum RAM and requires a long migration time. Hence, by manipulating the utilization threshold and migration cost function, the trojan significantly increases the data-center power consumption.

5.3.1 Evaluation Methodology

The proposed attack model is evaluated using the CloudSim simulator [144, 145] as it provides support for both system and behavioral modeling of individual components such as physical hosts, VMs, and resource provisioning policies. Further, it also provides support for modeling and simulation of energy-aware computation, with adaptive heuristics for dynamic consolidation of VMs based on resource utilization analysis [144, 145]. CloudSim simulator also includes power-aware VM allocation and migration schemes to perform more energy-efficient data center operations [145].

To implement the proposed attacks, VM allocation heuristics that inspects the host resource utilization are manipulated. In this work, two VM allocation policies, inter quartile range (IQR) and static utilization threshold (THR) are chosen to evaluate the impact of *HyperAttack*. Along with two VM allocation policies, minimum migration time (MMT) is chosen as the VM migration policy. VM allocation is provisioning of a VM on a physical

host based on the VM's request for resources such as, predefined processor cores, memory, storage, network bandwidth etc. IQR policy determines the upper utilization threshold based on a statistical dispersion, equal to the difference between the third and first quartiles. IQR algorithm is manipulated to covertly reduce the utilization threshold and induce more VM migrations. The THR technique, however uses a static threshold to detect over utilization based on host usage. For a more comprehensive evaluation of the attack, four variants of the attack are simulated. *HyperAttack* variants differ based on the sanctioned deviation of utilization threshold from the optimal value. For four attack variants, the trojan is curtailed from manipulating the utilization threshold within 10%, 20%, 30% and 40% of the optimal value.

This evaluation employs Planetlab workload, that comprises of real life traces of thousand VMs from servers located around the world [146]. Table 5.1 illustrates the simulation model based on a data-center with HP Proliant ML110 servers.

5.3.2 Potency of HyperAttack

Figure 5.3 shows the average increase in energy consumption as a result of attacks on the data-center, for two VM allocation policies: IQR and THR. As expected, the results reveal an increasing trend in energy consumption with increasing degree of attack (deviation

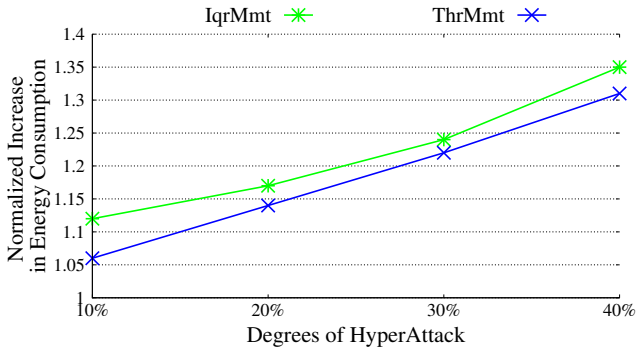
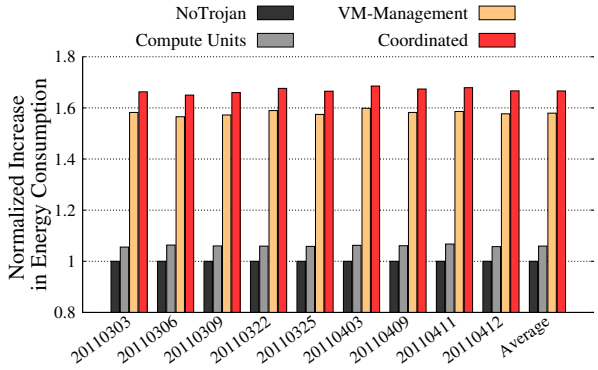


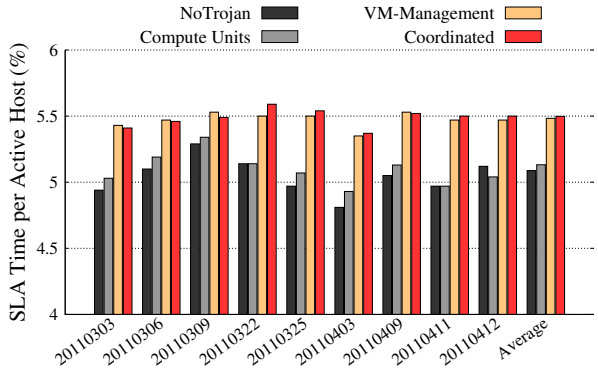
Fig. 5.3: Potency of attack variations.
The x-axis denotes the limits enforced by an attacker to the deviation of utilization threshold from the optimal value, and the y-axis denotes the increase in energy consumption normalized to the optimal energy consumption. Trend shows that as an attacker increases the sanctioned manipulation of threshold, the attack potency increases.

from optimal threshold). When an attacker limits the deviation of manipulated threshold to 30% of the optimal value, energy consumption increases on an average by 22% for IQR, and 24% for THR VM allocation policy. Figures 5.4, 5.5 and 5.6 consider one variant of the *HyperAttack* and examines its potency in detail. For these results, a maximum deviation of 50% from the optimal threshold is enforced along with other components of the attack. The trojan is designed to inflict a significant increase in power consumption with minimal impact on application performance measured as the SLA violations suffered. Cloudsim models SLA in terms of quality of service (QoS) parameters such as availability, reliability and throughput. Four scenarios are evaluated: **NoTrojan** signifies the legitimate behavior. **Compute Units** and **VM-Management** imply that only the respective resource is attacked and a **Coordinated** attack illustrates the interplay of both the attack vectors.

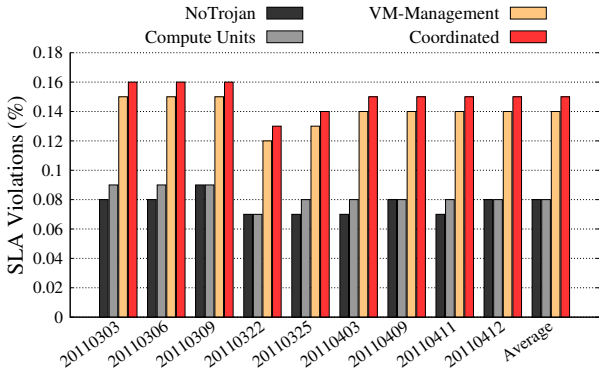
Figure 5.4a demonstrates that an attack on the compute resource of the data-center alone, increases the energy consumption on an average by 6%. Attack on compute resources shows a limited influence on the power consumed predominantly due to constraints of maintaining an obscure attack profile by preserving the application performance. However, an attack on the VM management policy increases the energy consumption by nearly 60%, on an average. Interestingly, a *coordinated attack* inflicts higher damage than the sum of its parts and the overall increase in energy consumption is 72%, on an average. Figure 5.5a shows a similar trend in energy consumption for the THR VM allocation policy, with the coordinated attack inflicting 47% increase in energy consumption on an average. On closer examination, an overall increase in the number of VM migrations is observed, forced by the attack during data-center runtime. VM migrations are a resultant of two orthogonal components: (a) shutdown of active hosts to prevent resource damage from aggressive consolidation, and (b) workload dispersion to caused by reduction in allowed utilization . Figure 5.4b and 5.5b present the percentage increase in SLA time per active host (SLATAH). SLATAH is the percentage of time during which active hosts experience a CPU utilization of 100%. On an average SLATAH increases by 5.5% for IQR and 5.77% for THR allocation policies. The increase in SLATAH indicates an increase in the



(a) Increase in Energy Consumption.

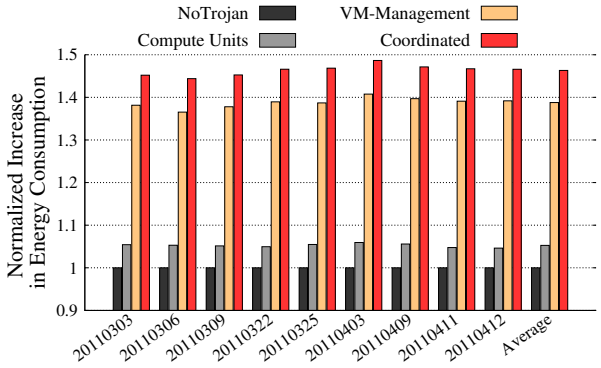


(b) Increase in SLA Time per active host

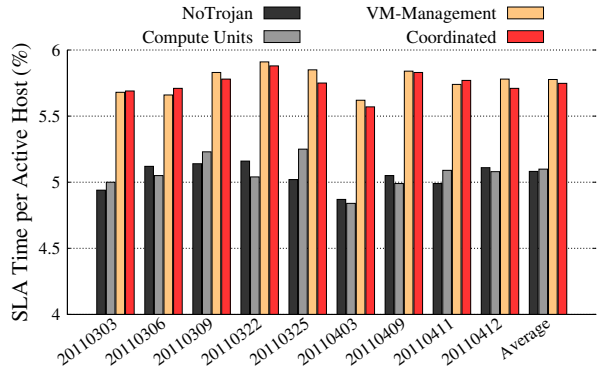


(c) Increase in IQR SLA violations.

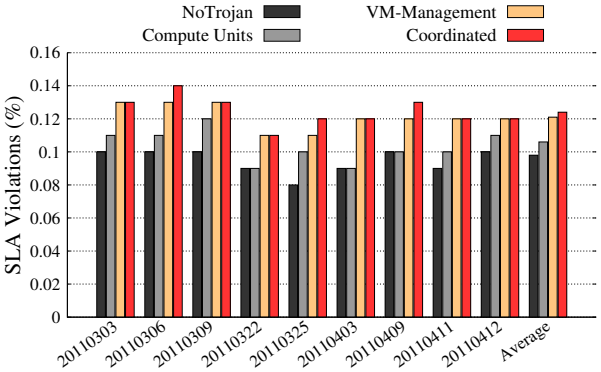
Fig. 5.4: Impact of a *HyperAttack* using inter quartile range VM migration policy. Four scenarios are evaluated: **NoTrojan** signifies the legitimate behavior. **Compute Units** and **VM-Management** imply that only the respective resource is attacked and a **Coordinated** attack illustrates the interplay of both the attack vectors. Figure 5.4a shows a substantial increase in the energy consumption. Figure 5.4b shows the substantial increase in SLA time per active host. Figure 5.5c show attractive stealth characteristics as the overall SLA violations are low. *X*-axis represents the planetlab workload trace IDs collected from real servers.



(a) Increase in Energy Consumption.



(b) Increase in SLA Time per active host



(c) Increase in THR SLA violations

Fig. 5.5: Impact of a *HyperAttack* using static utilization threshold VM migration policy. Four scenarios are evaluated: **NoTrojan** signifies the legitimate behavior. **Compute Units** and **VM-Management** imply that only the respective resource is attacked and a **Coordinated** attack illustrates the interplay of both the attack vectors. Figure 5.5a shows a substantial increase in the energy consumption. Figure 5.5b shows the substantial increase in SLA time per active host. Figure 5.5c show attractive stealth characteristics as the overall SLA violations are low. *X*-axis represents the planetlab workload trace IDs collected from real servers.

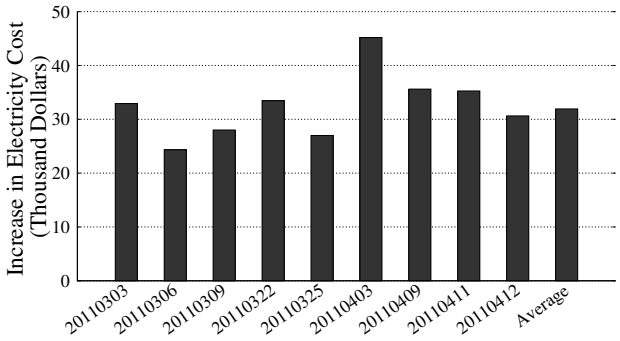


Fig. 5.6: Economic impact of a *HyperAttack* on energy cost.

VM migrations to prevent performance degradation.

Figure 5.4c and 5.5c show that even under an attack, the increase in overall SLA violations, remain within 0.15%, even while the corresponding impact on energy consumption is greater than 60%. In several workload traces, attack on the compute units alone does not reflect in the SLA violations at all. In addition, Figure 5.5c shows a marginal increase in SLA violations across all variants of the attack. This small percentage of SLA violations is distributed across different VMs, and multiple users. The noticeable impact on end user experience is minimal, thereby keeping the attack stealthy.

Figure 5.6 illustrates the economic impact on the data-center. An upsurge in the power consumed directly transforms to inflated electricity consumption costs, both to keep the physical servers in operation, as well as, to cool the operational servers. On an average, the required operational costs increase by more than $\$30,000$ per month.

5.3.3 HyperAttack Escalation

HyperAttack can be further escalated to catastrophic proportions to cause a power outage and impair the data-center services by exploiting the current trends of data-center power over-subscription. To alleviate the high cost of infrastructure upgrades, and to maximize returns from the existing servers, data-centers are adopting power over-subscription [147]. Since workloads in the data-center rarely peak simultaneously, power over-subscription supports a larger number of physical servers than the rated capacity. To safeguard from

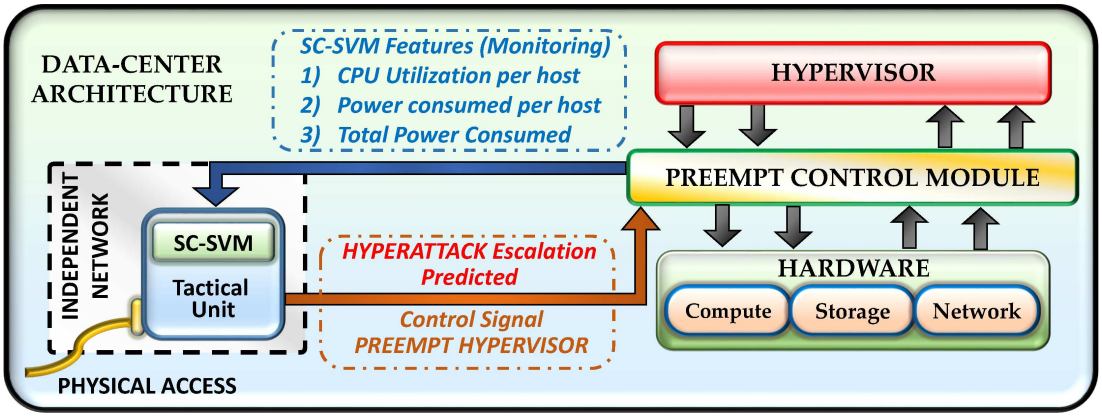


Fig. 5.7: Framework of *SCALE*. Tactical unit, an isolated core executes the SC-SVM to classify the data from preempt control module (*PCM*). PCM monitors power management decisions and responses, along with system parameters. Under an escalation attack, *PCM* seizes control of the data-center.

power violations, a variety of power capping techniques are employed at the server, racks and the system level. However, these techniques will prove ineffective if the hypervisor itself is compromised and causes intentional power violations. For instance, over-aggressive consolidation of VMs onto fewer physical servers can overload the circuits and trip the circuit breakers and at times even damage the infrastructure.

Power outage—impact of an escalated HyperAttack—can result in service downtime and monumental revenue losses. A major airline recently lost about $100 million in revenue due to a power outage in its data-center [9]. In 2013, data-center downtime across industries cost a whopping $7900 per minute with the cost per incident as high as $690,000. A recent power outage resulted in cancellation of 2300 flights globally and cost the airline about $100 million in revenue. Realizing the monumental impact of power management security in data-centers, this work introduces a novel secure architecture to detect anomalous power consumption behavior and safeguard against *HyperAttack* escalations.

5.4 HyperAttack Mitigation

Figure 5.7 illustrates the framework of the proposed **S**ingle **C**lass **A**ssisted **L**earning **E**nvironment (*SCALE*) to detect hypervisor inflicted power attacks and protect against catastrophic *HyperAttack* escalations. Section 5.4.1 presents a comprehensive glimpse at

the functioning of the secure architecture. *SCALE*'s two co-dependent components, *machine learning framework* (Section 5.4.2) and *preemptive control module* (Section 5.4.3) are discussed next.

5.4.1 SCALE Overview

SCALE employs a two pronged approach for security assurance:

1. Monitors and learns the acceptable behavior of data-center vitals to identify anomalous behaviors.

2. Preempts control from the hypervisor to prevent *HyperAttack* escalations.

To fulfill these objectives coherently, *SCALE* requires two co-dependent modules:

- ***Machine learning framework (MLF)*** (Section 5.4.2) incorporates an independent computation core known as tactical unit (*TU*), optimized to execute the single class support vector machine (SC-SVM) [148]. SC-SVM identifies whether a test-point (here, recurring data-center vitals) lies within the trained data (here, accepted data-center vitals) to detect anomalous behavior and *HyperAttack* escalations.

- ***Preemptive Control Module (PCM)*** (Section 5.4.3) serves as an interface between the hypervisor and the hardware resources. Primarily, it monitors the hypervisor power management decisions along with the hardware responses and furnishes the *TU* with the required data. Additionally, it is equipped with the functionality essential to preempt hypervisor control and manage the data-center when the *TU* predicts a *HyperAttack* escalation.

5.4.2 Machine Learning Framework

An independent high-performance core (tactical unit) optimized to execute machine learning algorithms is allocated to the data-center during system design. Depending on the data-center owner's security requirement, *TU* can be either placed on an independent secure network or totally devoid of remote access to protect against cyber-attacks. *TU*'s

only point of communication to the data-center is through the *PCM*. It accesses features collected by the *PCM* and feeds it to the learning algorithm for the detection of anomalies in data-center vitals. *TU* logs these anomalies for further analysis. However, on detecting a *HyperAttack* escalation, it first commands the *PCM* to seize control from the hypervisor to prevent a power outage and then notifies the administrator.

One key challenge of *SCALE* is to accurately classify legitimate and anomalous behaviors while using a limited set of data. *Traditional multi-class learning algorithms require two samples of data (here, malicious data-center vitals, as well as, accepted vitals) to aid the process of classification, while single class algorithms need only one.* In the complex data-center environment, acquisition of collective training data of all malicious behaviors is impractical, due to limitations in attack modeling and simulation. Hence, *SCALE* will use a single class SVM algorithm to capture and classify data-center vitals. Data classification and anomaly detection using SVM involves two phases: training and analysis, discussed below.

- *Training*: In this phase, *TU* is provided with n data points denoted by $\{x_1...x_n\}$, that correspond to legitimate data-center vitals. SC-SVM creates a model of the data-set, maps that data into a feature space and establishes a hyperplane that will divide the feature space into two disconnected regions. Region below the hyperplane is populated with a cluster of data-points that represent the legitimate data-center vitals.

- *Analysis*: In this phase, for any new incoming sample (current data-center vital), the SC-SVM classification function will determine which side of the hyperplane the sample lies, to identify an anomalous behavior. In addition, a subset of data-center vitals like compute utilization and power consumption, for example, will be frequently monitored. *An aggressive increase in these data-center vitals, among the physical nodes sharing a power distribution network, will be flagged as a* HyperAttack escalation.

SC-SVM Feature Extraction

Selection and optimization of relevant features is essential to ensure the accuracy and computational efficiency of the SC-SVM. To mitigate the threat presented by *HyperAttack*,

three evident features are identified and collected from the *PCM*: *(1) compute utilization per host*, *(2) power consumed per host* and *(3) total data-center power*. However, with data-centers having hundreds of physical hosts, considering each individual compute utilization and power consumed per host will lead to large computational overheads. Hence, to optimize the feature set, the physical hosts are clustered together and the mean utilization of each cluster is considered. To compensate for the loss in accuracy, a fourth feature is added: *(4) number of idle hosts in each cluster*.

The process of learning the characteristics of legitimate execution and anomaly detection relies severely on feature extraction. *SCALE* can be extended to detect a broader set of attacks by a careful selection of the feature set. For example, to detect a power management attack on the data-center network, the following set of features will prove beneficial: *network bandwidth utilization*, *network power consumption*, *network power state*, *VM migration* and *workload characteristics*.

The influence of each feature will depend significantly on the objective of a given attack vector. By carefully selecting the feature set (for example: *VM migrations*, *storage utilization*, *network bandwidth utilization*), *SCALE* can be extended to detect a broader set of attacks.

HyperAttack Escalation Prediction

Based on parameters collected from the *PCM*, the *TU* can monitor *compute utilization* and *power consumption* of physical hosts sharing a power distribution unit. An aggressive increase in these data-center vitals, among the physical nodes sharing a power distribution network, will be flagged as a *HyperAttack escalation*, and prompt the *TU* to signal the *PCM* to seize control of the data-center.

5.4.3 Preemptive Control Module

The PCM is a featherweight software layer beneath the hypervisor, **designed in-house by the data-center team.** It will serve as an interface between the hypervisor and the hard-

ware resources. Similar to TinyChecker [149], the *PCM* will be a special-purpose hypervisor that can be designed in-house, capitalizing on the concept of nested virtualization [150]. Primarily, the PCM's role will be to monitor the hypervisor power management decisions, along with the hardware responses, and furnish the *TU* with the required data. Furthermore, it will be able to preempt the hypervisor control when the *TU* predicts a *HyperAttack* escalation.

In the labyrinthine data-center setting, monitoring and controlling the critical interface between commodity hardware and third-party software renders a vital leverage to validate the presence of malicious entities. To assure reliability through formal verification, the *PCM* must have a small code footprint while fulfilling the following objectives:

- *Interface Monitoring*: The *PCM* snoops on data communicated between hypervisor and the hardware and compiles a set of inputs to the *SC-SVM*.
- *Preempt Data-center Control*: When the *TU* predicts a *HyperAttack* escalation, *PCM* preempts all hypervisor decisions and takes over the data-center. *PCM* evenly spreads subscribing VMs among all physical hosts to maintain reliable operation until the hypervisor security is investigated.

To maintain control flow integrity, the *PCM* has no privileges to send unsolicited signals to the *TU* and no runtime modification to the *PCM* function is allowed.

The addition of PCM between hardware and hypervisor, albeit a featherweight bare bone module, will add a small performance overhead to the system. The performance overhead due to PCM will predominantly depend on three factors: *(a)* architectural overhead, *(b)* I/O overhead and *(c)* control flow overhead. In addition, the penalty inflicted due to PCM will vary based on the type of workload, as an I/O intensive or memory intensive workload will have frequent data exchanges across the layers, and the control flow will add a small overhead for VM entry and exits. The multitude of factors, along with the dependence on workload characteristics make it difficult to accurately estimate the performance overhead of PCM.

5.4.4 Challenges of SC-SVM Adoption

Following challenges attribute to the use of SC-SVM:

- *ν parameter*: In SC-SVM, parameter ν determines the upper bound on the fraction of outliers (here, malicious events), and the lower bound for the number of samples that are support vectors (here, legitimate samples). Broadly, ν parameter plays a decisive role in trade-off between overfitting and generalization. Hence, ν significantly affects the accuracy of SC-SVM for anomaly detection but ν's value cannot be designated in advance. ν is selected by meticulous testing of the SC-SVM with a data-set having only the legitimate data-center vitals and ensuring that the false detection is minimal. The influence of ν on *SCALE* accuracy is discussed in Section 5.4.6.

- *Feature Extraction*: Computation complexity and efficacy of *SCALE* largely depends on the choice and size of feature set selected. In a complex and large data-center, accurately determining the features that are affected by an attack vector and optimizing these features is arduous.

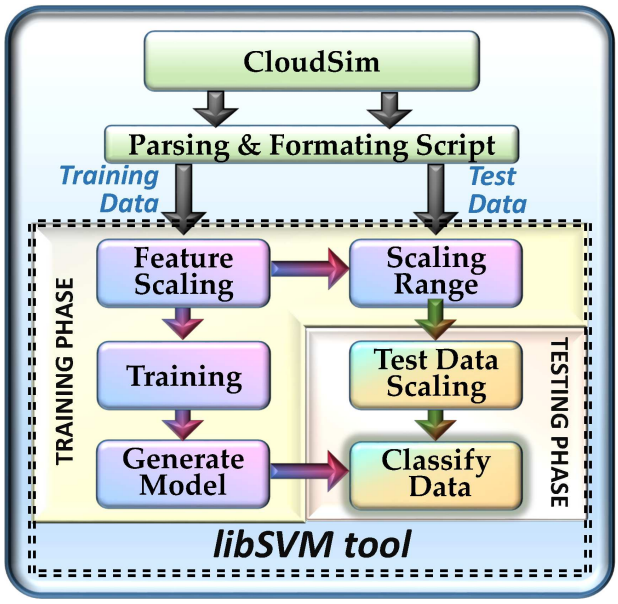


Fig. 5.8: Implementation of *SCALE* using libSVM tool.

5.4.5 Methodology

Figure 5.8 shows the process of SC-SVM training and analysis along with feature set collection from Cloudsim [144, 145]. The chosen attributes are monitored as features (discussed in Section 5.4.4), and data-points are optimized by finding the mean of each cluster of physical hosts. This processed data is used to generate the training set for the SC-SVM. ***Single-Class Support Vector Machine***: An open source tool, *libSVM* [151] is used to build the learning environment and classify data in sequence of phases shown in Figure 5.8. *libSVM* models a SC-SVM based on the work presented in [148]. The sequences of phases for SC-SVM classification are:

- *Scaling Phase*: The data collected from Cloudsim is processed, formatted and fed to the *libSVM* tool where the data is scaled to a value between -1 to 1.
- *Training Phase*: SC-SVM cross-validates the scaled data and generates a training model by identifying the required tuning parameters (C, γ and $CVrate$).
- *Analysis Phase*: Test data from Cloudsim is properly formatted and supplied to the SC-SVM tool for classification.

5.4.6 Analysis of SCALE

This section presents the sensitivity and specificity analysis of *SCALE* for classifying power utilization characteristics. The influence of parameter ν on *SCALE*'s accuracy is analyzed first. The following metrics are used to analyze the efficacy of *SCALE* in hypervisor power attack detection: *(i)* classification accuracy, *(ii)* false positive (FP) and false negative (FN) and *(iii)* precision and recall.

ν Parameter Analysis

Figure 5.9 shows the influence of ν on the *SCALE* accuracy. The x-axis represents the value of ν used for classification, and y-axis represents the average classification accuracy for PlanetLab traces. In this work, SC-SVM employs a radial basis function (RBF) kernel. ν is upper bound by the fraction of outliers while lower bound by the fraction of support

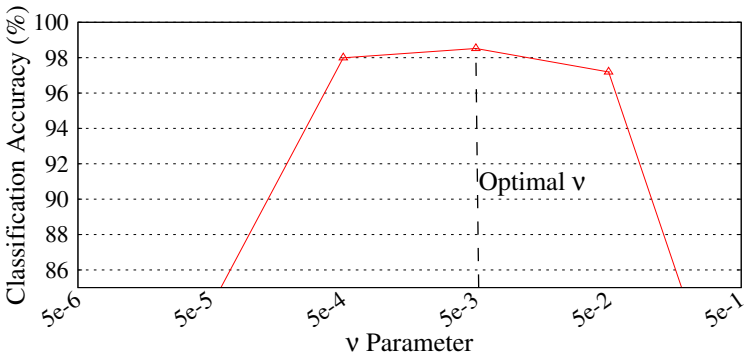


Fig. 5.9: ν parameter's influence on classification accuracy.

vectors. Figure 5.9 shows that *SCALE* achieves the best accuracy at an optimal ν of 0.005 and the accuracy drops off by increasing or decreasing ν.

***SCALE* Efficacy**

Figure 5.10 presents a comprehensive analysis of the efficacy of SC-SVM learning algorithm in the proposed novel secure architecture, *SCALE*, considering a 40% deviation from optimal selection as the attack variant using three metrics:

- ***Classification Accuracy*** : Figure 5.10a shows the detailed classification accuracy of *SCALE* for PlanetLab workload traces for the two VM allocation policies. Accuracy of *SCALE* is defined as the ratio of correctly classified samples to the total samples. On an average, *SCALE* delivers an accuracy of 99% for IQR and 98% for THR with ν set at 0.005. Traces collected on *20110309* (March 09 2011) suffered in classification accuracy for IQR policy compared to traces collected on other days. For this trace, it was observed that 71 samples were incorrectly flagged by *SCALE*.

- ***FP and FN***: Figure 5.10b illustrates the percentage of false positive detection. Trojan-free samples incorrectly classified as malicious samples are represented as FP. For most traces, the FP lies around 1% for IQR, and 2.5% for THR. However, in IQR policy two traces *20110322* and *20110411*, suffer from higher FP rates of 3.8% and 2.2% respectively. Detailed analysis showed that during these samples the overall

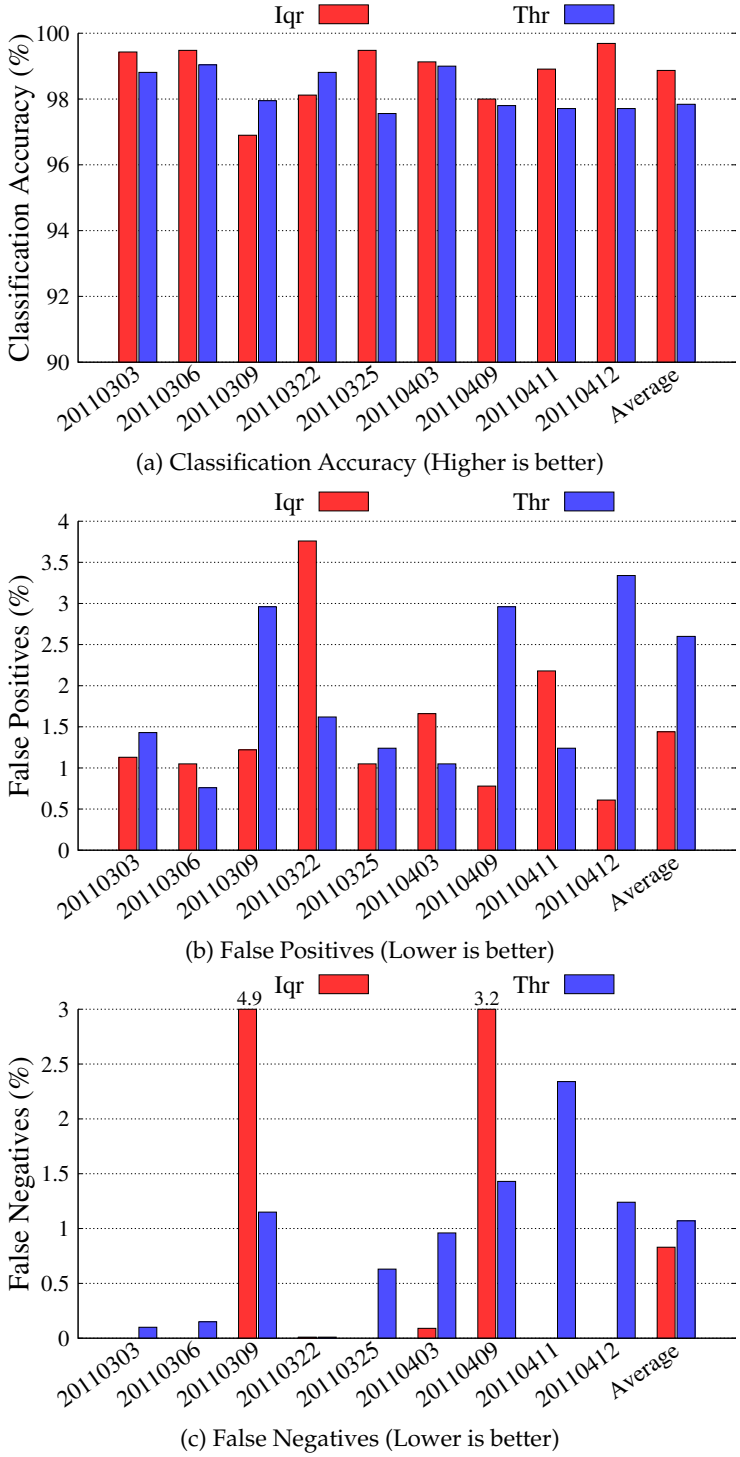


Fig. 5.10: Efficacy of *SCALE* for the detection of anomalies in data-center power consumption.

load on the data-center was low, and aggressive consolidation of VMs was falsely identified as malicious activity.

FN analysis (Figure 5.10c) showed that only three workload traces incurred FN for IQR, whereas FN was slightly more prevalent in THR policy. FN describes *SCALE*'s inability to detect true security event samples under certain circumstances. Ideally all malicious activity should be detected and quarantined. However, the minimal FN rate of *SCALE* for dynamic IQR VM allocation policy is promising. *20110309* incurred nearly 5% FN, i.e, *SCALE* failed to identify 5 out of every 100 malicious samples. Overall, the FN rate was 0.8% for IQR and 1% for THR.

- ***Precision and Recall***: In binary classification such as SC-SVM, precision and recall present a measure of relevance. Precision also known as positive predictive value is given by the Equation 5.1. A high precision indicates that only a handful of legitimate samples are wrongly classified. Figure **??** shows that *SCALE* has a high precision (98.58% on average for IQR, and 97.3% for THR).

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{5.1}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{5.2}$$

Recall (Equation 5.2), on the other hand indicates the sensitivity of *SCALE* to accurately identify malicious behavior. Analysis revealed that *SCALE* has a 99.2% sensitivity for IQR.

Figure 5.11 summarizes the accuracy of SCALE in detection of power anomalies across power attack variants discussed in Section 5.3.1. The 10%, 20%, 30% and 40% in the x-axis indicate the maximum deviation of utilization threshold from optimal value that can be inflicted by an attacker. As the degree of attack increases from 10% to 40%, the detection accuracy of SCALE also increases for nearly all data-center traces.

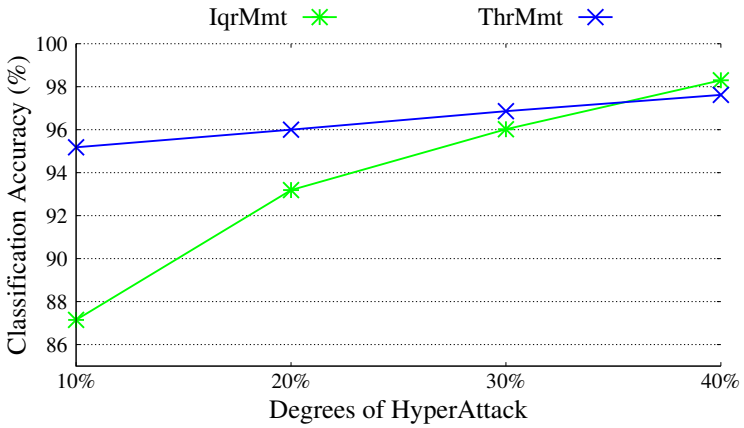


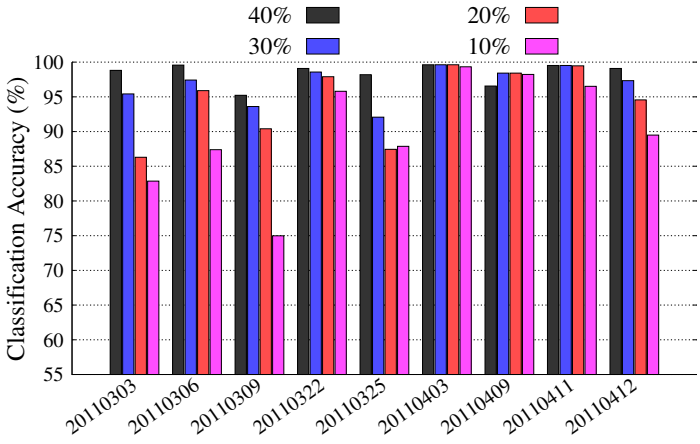
Fig. 5.11: Average detection accuracy of *SCALE* across benchmarks.

For 10% deviation of utilization threshold in Figure 5.11, the average detection accuracy falls to 87% for IQR allocation policy. On correlating these results to the attack potency in Figure 5.3, it can be observed that the average impact of this variant of the attack on the data-center is minimal. Further analysis reveals that the delta in energy consumption was spread out across different epochs of the trace runtime, and the number of host shutdowns and VM migrations resulting from the attack variant was minimal, and accurately identified as an anomaly.

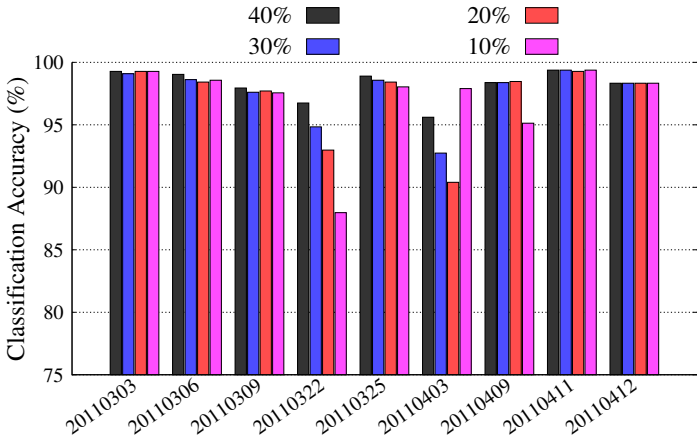
The correlation of Figure 5.3 to Figure 5.11 uncovers another interesting characteristic that acts as a proponent to SCALE's credibility. Although the attack inflicts a bigger impact for IQR allocation policy, than when THR policy is enforced (in Figure 5.3), the detection accuracy of SCALE is higher for THR. For example, for the attack variant of 10%, the increase in energy consumption is 6% for THR, and 12% for IQR (THR $<$ IQR), however the corresponding SCALE detection accuracy is 95.2% (THR) and 87.2% (IQR) respectively (i.e., THR $>$ IQR). The contrasting observation arises as THR uses a static threshold for VM allocation making it easier for the learning algorithm to identify anomalies, whereas IQR uses a dynamic threshold. However, in attack variants beyond 40%, the results become fairly consistent.

Figure 5.12 expands on results shown in Figure 5.11 illustrating the classification accuracy for different PlanetLab workloads that have been considered. Figure 5.12 clearly

demonstrates that dynamic runtime characteristics of workloads play a significant role in classification accuracy of *SCALE*.



(a) Classification Accuracy of IqrMmt (Higher is better)



(b) Classification Accuracy of ThrMmt (Higher is better)

Fig. 5.12: Detection accuracy of *SCALE* under different attack variants for both IQR and THR allocation policies. 10%, 20%, 30% and 40% indicate the maximum deviation of utilization threshold the attacker sanctions. Higher deviation suggests a stronger attack.

CHAPTER 6
CONCLUSION

The research pursued in this dissertation navigates through vulnerability identification, threat assessment and security assurance using design-for-trust solutions against compromised third party power management solutions—*a first of its kind*. This research proposes innovative designs with low footprint to defend against security vulnerabilities in modern complex power management subsystems. The techniques designed during this work tackle security vulnerabilities across different abstraction levels. By a joint exploration of attack vectors, as well as, techniques to seamlessly detect and protect against vulnerabilities, this work presents a holistic approach to lay the foundation for trustworthy and reliable computing in the presence of malicious modifications in power management sub-system.

Deriving the threat model from semiconductor supply chain plagued with untrustworthy third party components, this work exposed security concerns in specialized hardware power management units. This research proved that covert hardware trojans with negligible footprints can exist in the PMU, operate in stealth and inflict potent attacks on the system. Uncovering this novel and imminent challenge to secure hardware design, this research led to the design of novel IP risk assessment module placed at the boundary between trusted and untrustworthy components, to detect and defend against attacks from compromised power management hardware. IPRAM has minimal design and operational footprint, and can easily be developed in-house by MPSoC integrator without assistance from third party vendors.

In the realm of high performance data-centers, the repercussions of malicious modifications in power management sub-systems are catastrophic. To safeguard the data-centers from a broad range of internal power and energy attacks, a secure framework called *SCALE* was developed. *SCALE* employs fundamental concepts of security, i.e., isola-

tion and redundancy, in combination with a learning framework to detect power consumption anomalies. The secure framework delivers high classification accuracy in detecting malicious power activities across multiple power management strategies and dynamically varying real-time workload traces.

This research underscores the influence of power management sub-systems on system functionality, performance and energy efficiency. It uncovers an imminent threat to the emerging energy proportional computing paradigm, posed by a new class of exploitable security vulnerabilities in complex third party power management solutions. As a result, this dissertation lays the necessary foundation for design of secure, reliable and energy efficient computing systems.

REFERENCES

[1] A. S. Andrae and T. Edler, "On global electricity usage of communication technology: trends to 2030," *Challenges*, vol. 6, no. 1, pp. 117–157, 2015.

[2] R. Wilson, "What your soc designer might not tell you about power management," http://www.design-reuse.com/articles/30018/power-management.html.

[3] TechNavio, "Global pc and server power management software market," Tech. Rep., June 2013.

[4] P. Reserach, "Pc and server power management software," Tech. Rep., October 2011.

[5] Gartner Inc., *Competitive Landscape: Power Management IC and Power Semiconductor Vendors*. Gartner Research, 2012.

[6] Sonics Inc. Ice-grain power architecture for mainstream soc designs-the semiconductor ip industrys first power management solution, 2015. [Online]. Available: http://sonicsinc.com/products/ice-grain-power-architecture/

[7] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Design and Test for Computers*, vol. 27, no. 1, pp. 10–25, 2010.

[8] F. Nekoogar, *From ASICs to SOCs: a practical approach*. Prentice Hall Professional, 2003.

[9] M. Sasso, "Delta pegs loss from computer breakdown at $100 million," 02 Sep 2016, http://www.dallasnews.com/business/airline-industry/20160902-delta-pegs-loss-from-computer-breakdown-at-100-million.

[10] Y. Sverdlik, "Verizon data center outage delays jetblue flights," http://www.datacenterknowledge.com/archives/2016/01/14/verizon-data-center-outage-delays-jetblue-flights/.

[11] S. Rich, "Battery failure, human error still cause most data center outages," http://www.govtech.com/data/224102581.html.

[12] Y. Lee and S. Kim, "Samsung blames small battery flaw for prompting note 7 recall," https://www.bloomberg.com/news/articles/2016-09-13/samsung-blames-small-battery-flaw-for-igniting-note-7-recall.

[13] G. Keizer, "Expert: ipad wi-fi issues may be linked to power management," https://www.computerworld.com/article/2502757/apple-mac/expert--ipad-wi-fi-issues-may-be-linked-to-power-management.html.

[14] C. Jensen, "Chrysler owners sound off on a power defect," http://www.nytimes.com/2014/08/23/automobiles/chrysler-owners-sound-off-on-defective-power-module.html.

[15] "Toyota australia to recall prius and prius v vehicles due to the power management control software," http://www.govtech.com/data/224102581.html.

[16] C. Whitlock, "More air force drones are crashing than ever as mysterious new problems emerge," https://www.washingtonpost.com/news/checkpoint/wp/2016/01/19/more-u-s-military-drones-are-crashing-than-ever-as-new-problems-emerge/.

[17] R. J. Shridevi, D. M. Ancajas, K. Chakraborty, and S. Roy, "Runtime detection of a bandwidth denial attack from a rogue network-on-chip," in *ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, 2015, pp. 8:1–8:8.

[18] R. JayashankaraShridevi, D. M. Ancajas, K. Chakraborty, and S. Roy, "Security measures against a rogue network-on-chip," *Journal of Hardware and Systems Security*, pp. 1–15, 2017.

[19] R. J. Shridevi, D. M. Ancajas, K. Chakraborty, and S. Roy, "Tackling voltage emergencies in noc through timing error resilience," in *ACM International Symposium on Low Power Electronic Devices (ISLPED)*, 2015, pp. 104–109.

[20] P. Basu, R. J. Shridevi, K. Chakraborty, and S. Roy, "Prada: Combating voltage noise in the noc power supply through flow-control and routing algorithms," in *IEEE/ACM Design Automation & Test in Europe (DATE)*, 2016.

[21] ——, "Iconoclast: Tackling voltage noise in the noc power supply through flow-control and routing algorithms," *IEEE Trans. VLSI Syst.*, vol. 25, no. 7, pp. 2035–2044, 2017.

[22] C. Rajamanikkam, R. J. Shridevi, S. Roy, and K. Chakraborty, "Boostnoc: Power efficient network-on-chip architecture for near threshold computing," in *IEEE International Conference on Computer-Aided Design (ICCAD)*, 2016.

[23] R. J. Shridevi, C. Rajamanikkam, S. Roy, and K. Chakraborty, "Catching the flu: Emerging threats from a third party power management unit," in *IEEE/ACM Design Automation Conference (DAC)*, 2016, pp. 86:1–86:6.

[24] J. Rajesh, C. Rajamanikkam, K. Chakraborty, and S. Roy, "Securing data center against power attacks," *Journal of Hardware and Systems Security*, pp. 1–12, 2019.

[25] D. Hillman, V. Silicon, and J. W. Tensilica, "Implementing power management ip for dynamic and static power reduction in configurable microprocessors using the galaxy design platform at 130nm."

[26] *Integrated Power Management IC For IMVP8 Platforms*, http://www.intersil.com/content/dam/Intersil/documents/isl9/isl95852.pdf.

[27] *Dual-Channel (3-Phase CPU/2-Phase GPU) SVID, D-CAP+ Step-Down Controller for IMVP-7 VCORE with Two Integrated Drivers*, http://www.ti.com/lit/ds/symlink/tps59650.pdf.

[28] *High Efficiency Power Supply for Intel IMVP-6/IMVP-6+/IMVP-6.5 CPUs*, http://cds.linear.com/docs/en/lt-journal/LTJournal-V21N3-04-df-LTC3816-Jian_Li.pdf.

[29] *Companion PMIC for Smartphone and Tablet*, https://datasheets.maximintegrated.com/en/ds/MAX77829.pdf.

[30] J. Bagherli, "Powering the next generation portable devices through programmable pmics," http://www.gsaglobal.org/events/2012/0508/docs/dialogkeynote_webversion.pdf.

[31] *High Efficiency Display Power and LED Driver for Smart Phones*, http://www.intersil.com/content/dam/Intersil/documents/isl9/isl98611.pdf.

[32] *ICE-G1 Energy Processing Unit*, http://sonicsinc.com/wp-content/uploads/2016/10/ICE-G1_Product-Brief_160929.pdf.

[33] E. Esteve, "New power management ip solution can dramatically increase soc energy efficiency," https://www.dolphin-integration.com/index.php/pdf/newsroom/in_the_press/201809_ipnest-Power-IP-for-SoC.

[34] A. Bourdon, A. Noureddine, R. Rouvoy, and L. Seinturier, "Powerapi: A software library to monitor the energy consumed at the process-level," *ERCIM News*, vol. 2013, no. 92, 2013. [Online]. Available: http://ercim-news.ercim.eu/en92/special/powerapi-a-software-library-to-monitor-the-energy-consumed-at-the-process-level

[35] J. Eastep, S. Sylvester, C. Cantalupo, B. Geltz, F. Ardanaz, A. Al-Rawi, K. Livingston, F. Keceli, M. Maiterth, and S. Jana, "Global extensible open power manager: A vehicle for HPC community collaboration on co-designed energy management solutions," in *High Performance Computing - 32nd International Conference, ISC High Performance 2017, Frankfurt, Germany, June 18-22, 2017, Proceedings*, 2017, pp. 394–412.

[36] I. Hewlett-Packard, "Microsoft, phoenix, and toshiba. advanced configuration and power interface specification," 2004.

[37] J. Stoess, C. Lang, and F. Bellosa, "Energy management for hypervisor-based virtual machines." in *USENIX annual technical conference*, 2007, pp. 1–14.

[38] Y. Liu and H. Zhu, "A survey of the research on power management techniques for high-performance systems," *Software: Practice and Experience*, vol. 40, no. 11, pp. 943–964, 2010.

[39] C. Isci, S. McIntosh, J. Kephart, R. Das, J. Hanson, S. Piper, R. Wolford, T. Brey, R. Kantner, A. Ng *et al.*, "Agile, efficient virtualization power management with low-latency server power states," in *ACM SIGARCH Computer Architecture News*, vol. 41, no. 3. ACM, 2013, pp. 96–107.

[40] M. Kandemir, N. Vijaykrishnan, and M. J. Irwin, "Compiler optimizations for low power systems," in *Power aware computing*. Springer, 2002, pp. 191–210.

[41] R. Kravets and P. Krishnan, "Application-driven power management for mobile communication," *Wireless Networks*, vol. 6, no. 4, pp. 263–277, 2000.

[42] A. Lungu, P. Bose, D. J. Sorin, S. German, and G. Janssen, "Multicore power management: Ensuring robustness via early-stage formal verification," in *Proceedings of IEEE/ACM International Conference on Formal Methods and Models for Co-Design (MEMOCODE)*, 2009, pp. 78–87.

[43] F. Bembaron, S. Kakkar, R. Mukherjee, and A. Srivastava, "Low power verification methodology using upf," *Proceedings of DVCon*, pp. 228–233, 2009.

[44] D. Kouroussis and F. N. Najm, "A static pattern-independent technique for power grid voltage integrity verification," in *IEEE/ACM Design Automation Conference (DAC)*, 2003, pp. 99–104.

[45] M. Nizam, F. N. Najm, and A. Devgan, "Power grid voltage integrity verification," in *ACM International Symposium on Low Power Electronic Devices (ISLPED)*, 2005, pp. 239–244.

[46] F. J. Mesa-Martinez, J. Nayfach-Battilana, and J. Renau, "Power model validation through thermal measurements," in *International Symposium on Computer Architecture (ISCA)*, 2007, pp. 302–311.

[47] F. J. Mesa-Martinez, E. K. Ardestani, and J. Renau, "Characterizing processor thermal behavior," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2010, pp. 193–204.

[48] K. Najeeb, V. V. R. Konda, S. K. S. Hari, V. Kamakoti, and V. M. Vedula, "Power virus generation using behavioral models of circuits," in *IEEE VLSI Test Symposium (VTS)*, 2007, pp. 35–42.

[49] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "Peak power estimation of VLSI circuits: new peak power measures," *IEEE Transactions on VLSI Systems (TVLSI)*, vol. 8, no. 4, pp. 435–439, 2000.

[50] H. Kriplani, F. N. Najm, and I. N. Hajj, "Pattern independent maximum current estimation in power and ground buses of CMOS VLSI circuits: Algorithms, signal correlations, and their resolution," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 14, no. 8, pp. 998–1012, 1995.

[51] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE transactions on computers*, vol. 51, no. 5, pp. 541–552, 2002.

[52] M. Guri, M. Monitz, Y. Mirski, and Y. Elovici, "Bitwhisper: Covert signaling channel between air-gapped computers using thermal manipulations," *Journal on Computing Research Repository*, vol. abs/1503.07919, 2015.

[53] J. A. Ambrose, R. G. Ragel, and S. Parameswaran, "A smart random code injection to mask power analysis based side channel attacks," in *2007 5th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ ISSS)*. IEEE, 2007, pp. 51–56.

[54] M. Hutter and J.-M. Schmidt, "The temperature side channel and heating fault attacks," in *International Conference on Smart Card Research and Advanced Applications*. Springer, 2013, pp. 219–235.

[55] Z. Wu, M. Xie, and H. Wang, "Energy attack on server systems," in *USENIX Technical Conference*, 2011, pp. 62–70.

[56] F. Palmieri, S. Ricciardi, U. Fiore, M. Ficco, and A. Castiglione, "Energy-oriented denial of service attacks: an emerging menace for large cloud infrastructures," *The Journal of Supercomputing*, vol. 71, no. 5, pp. 1620–1641, 2015.

[57] Z. Xu, H. Wang, Z. Xu, and X. Wang, "Power attack: An increasing threat to data centers," in *21st Annual NDSS Symposium, USA, February 23-26*, 2014.

[58] X. Gao, Z. Xu, H. Wang, L. Li, and X. Wang, "Why some like it hot too: Thermal attack on data centers," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 45, no. 1. ACM, 2017, pp. 23–24.

[59] M.-K. Yoon, S. Mohan, J. Choi, J.-E. Kim, and L. Sha, "Securecore: A multicore-based intrusion detection architecture for real-time embedded systems," 2013, pp. 21–32.

[60] L. Fiorin, G. Palermo, S. Lukovic, V. Catalano, and C. Silvano, "Secure memory accesses on networks-on-chip," *IEEE Transactions on Computers (TC)*, vol. 57, no. 9, pp. 1216–1229, 2008.

[61] K. Hu, A. N. Nowroz, S. Reda, and F. Koushanfar, "High-sensitivity hardware trojan detection using multimodal characterization," in *IEEE/ACM Design Automation & Test in Europe (DATE)*, 2013, pp. 1271–1276.

[62] R. Paseman and A. Orailoglu, "Towards a cost-effective hardware trojan detection methodology," in *IEEE VLSI Test Symposium (VTS)*, 2013, pp. 1–3.

[63] J. Zhang, H. Yu, and Q. Xu, "Htoutlier: Hardware trojan detection with side-channel signature outlier identification," in *Proceedings of IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2012, pp. 55–58.

[64] A. K. Kanuparthi, R. Karri, G. Ormazabal, and S. Addepalli, "A high-performance, low-overhead microarchitecture for secure program execution," in *IEEE International Conference on Computer Design (ICCD)*, 2012, pp. 102–107.

[65] S. Ravi, A. Raghunathan, and S. T. Chakradhar, "Tamper resistance mechanisms for secure, embedded systems," in *IEEE International Conference on VLSI Design*, 2004, pp. 605–.

[66] T. Kgil, L. Falk, and T. N. Mudge, "Chiplock: support for secure microarchitectures," *Special Interest Group on Computer Architecture (SIGARCH) Computer Architecture News*, vol. 33, no. 1, pp. 134–143, 2005.

[67] B. Rogers, M. Prvulovic, and Y. Solihin, "Efficient data protection for distributed shared memory multiprocessors," in *IEEE Parallel Architectures and Compilation Techniques (PACT)*, 2006, pp. 84–94.

[68] G. E. Suh, D. E. Clarke, B. Gassend, M. van Dijk, and S. Devadas, "Efficient memory integrity verification and encryption for secure processors," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2003, pp. 339–350.

[69] J. A. Roy, F. Koushanfar, and I. L. Markov, "Epic: Ending piracy of integrated circuits," in *IEEE/ACM Design Automation & Test in Europe (DATE)*, 2008, pp. 1069–1074.

[70] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *IEEE/ACM Design Automation Conference (DAC)*, ser. DAC '07, 2007, pp. 9–14.

[71] C. Liu, J. Rajendran, C. Yang, and R. Karri, "Shielding heterogeneous mpsocs from untrustworthy 3pips through security-driven task scheduling," in *IEEE Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT)*, 2013, pp. 101–106.

[72] O. S. Hofmann, S. Kim, A. M. Dunn, M. Z. Lee, and E. Witchel, "Inktag: secure applications on an untrusted operating system," *ACM SIGPLAN Symposium*, vol. 48, no. 4, pp. 265–278, Mar. 2013.

[73] S. Checkoway and H. Shacham, "Iago attacks: why the system call api is a bad untrusted rpc interface," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM, 2013, pp. 253–264.

[74] F. Zhang, J. Chen, H. Chen, and B. Zang, "Cloudvisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization," in *SOSP*, ser. SOSP '11. ACM, 2011, pp. 203–216.

[75] X. Chen, T. Garfinkel, E. C. Lewis, P. Subrahmanyam, C. A. Waldspurger, D. Boneh, J. Dwoskin, and D. R. Ports, "Overshadow: a virtualization-based approach to retrofitting protection in commodity operating systems," *SIGOPS*, vol. 42, no. 2, pp. 2–13, Mar. 2008.

[76] D. R. K. Ports and T. Garfinkel, "Towards application security on untrusted operating systems," ser. HOTSEC'08. USENIX Association, 2008, pp. 1:1–1:7.

[77] R. Ta-Min, L. Litty, and D. Lie, "Splitting interfaces: making trust between applications and operating systems configurable," ser. OSDI '06. Berkeley, CA, USA: USENIX Association, 2006, pp. 279–292.

[78] J. Yang and K. G. Shin, "Using hypervisor to provide data secrecy for user applications on a per-page basis," in *ACM SIGPLAN Symposium*, ser. VEE. ACM, 2008, pp. 71–80.

[79] J. M. McCune, Y. Li, N. Qu, Z. Zhou, A. Datta, V. Gligor, and A. Perrig, "Trustvisor: Efficient tcb reduction and attestation," in *IEEE/ACM International Symposium on Security and Privacy*, ser. SP '10. IEEE Computer Society, 2010, pp. 143–158.

[80] J. M. McCune, B. J. Parno, A. Perrig, M. K. Reiter, and H. Isozaki, "Flicker: an execution infrastructure for tcb minimization," *SIGOPS*, vol. 42, no. 4, pp. 315–328, Apr. 2008.

[81] B. Parno, J. R. Lorch, J. R. Douceur, J. Mickens, and J. M. McCune, "Memoir: Practical state continuity for protected modules," in *IEEE/ACM International Symposium on Security and Privacy*, ser. SP '11. IEEE Computer Society, 2011, pp. 379–394.

[82] A. Behnia, R. A. Rashid, and J. A. Chaudhry, "A survey of information security risk analysis methods," *SmartCR*, vol. 2, no. 1, pp. 79–94, 2012.

[83] B. Liu, L. Shi, Z. Cai, and M. Li, "Software vulnerability discovery techniques: A survey," in *2012 Fourth International Conference on Multimedia Information Networking and Security*. IEEE, 2012, pp. 152–156.

[84] A. Apvrille, D. Gordon, S. E. Hallyn, M. Pourzandi, and V. Roy, "Digsig: Runtime authentication of binaries at kernel level." in *LISA*, vol. 4, 2004, pp. 59–66.

[85] X. Zhou, Y. Lee, N. Zhang, M. Naveed, and X. Wang, "The peril of fragmentation: Security hazards in android device driver customizations," in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 409–423.

[86] G. McGraw, *Software security: building security in*. Addison-Wesley Professional, 2006, vol. 1.

[87] J. Sahoo, S. Mohapatra, and R. Lath, "Virtualization: A survey on concepts, taxonomy and associated security issues," in *2010 Second International Conference on Computer and Network Technology*. IEEE, 2010, pp. 222–226.

[88] Y. Wang, J. Zheng, C. Sun, and S. Mukkamala, "Quantitative security risk assessment of android permissions and applications," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2013, pp. 226–241.

[89] M. Tehranipoor and C. Wang, *Introduction to Hardware Security and Trust*. Springer, 2011.

[90] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A novel technique for improving hardware trojan detection and reducing trojan activation time," *IEEE Transactions on VLSI Systems (TVLSI)*, vol. 20, no. 1, pp. 112–125, 2012.

[91] F. Imeson, S. Nejati, S. Garg, and M. Tripunitara, "Non-deterministic timers for hardware trojan activation (or how a little randomness can go the wrong way)," in *10th USENIX Workshop on Offensive Technologies WOOT 16)*, 2016.

[92] A. Waksman, M. Suozzo, and S. Sethumadhavan, "Fanci: identification of stealthy malicious logic using boolean functional analysis," in *ACM Conference on Computer and Communications Security (CCS)*, 2013, pp. 697–708.

[93] "Intelligence threat handbook," http://fas.org/irp/threat/handbook/.

[94] J. Brill, "The intersection of consumer protection and competition in the new world of privacy," *Competition Pol'y Int'l*, vol. 7, pp. 7–313, 2011.

[95] I. Arce, K. Clark-Fisher, N. Daswani, J. DelGrosso, D. Dhillon, C. Kern, T. Kohno, C. Landwehr, G. McGraw, B. Schoenfield *et al.*, "Avoiding the top 10 software security design flaws," *Technical report, IEEE Computer Societys Center for Secure Design (CSD)*, 2014.

[96] J. Chan, "Essentials of patch management policy and practice," Jan 2004, http://www.patchmanagement.org/pmessentials.asp.

[97] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, "Designing and implementing malicious hardware." in *Proceedings of 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, vol. 8, 2008, pp. 1–8.

[98] X. Jiang and Y. Zhou, *A Survey of Android Malware*. New York, NY: Springer New York, 2013, pp. 3–20.

[99] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown," *CoRR*, vol. abs/1801.01207, 2018.

[100] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," *meltdownattack.com*, 2018.

[101] "Hp power manager user guide," https://support.hpe.com/hpsc/doc/public/display?docId=emr_na-c01272419.

[102] "Hp power manager vulnerability cve-2011-0280," https://nvd.nist.gov/vuln/detail/CVE-2011-0280.

[103] "Hp power manager vulnerability cve-2010-4113," https://nvd.nist.gov/vuln/detail/CVE-2010-4113.

[104] "Intel power management controller firmware vulnerability cve-2018-3643," https://nvd.nist.gov/vuln/detail/CVE-2018-3643.

[105] S. Mittal, "Power management techniques for data centers: A survey," *CoRR*, vol. abs/1404.6681, 2014. [Online]. Available: http://arxiv.org/abs/1404.6681

[106] H. David, C. Fallin, E. Gorbatov, U. R. Hanebutte, and O. Mutlu, "Memory power management via dynamic voltage/frequency scaling," in *ACM Proceedings on International Conference on Autonomic Computing*, 2011, pp. 31–40.

[107] F. Farahnakian, P. Liljeberg, and J. Plosila, "Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning," in *22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2014, Torino, Italy, February 12-14, 2014*, 2014, pp. 500–507.

[108] T. M. Research, "Power management integrated circuit (pmic) market (product - voltage regulators, motor control ic, integrated assp power management ic, battery management ic, microprocessor supervisory ic; end use - automotive, consumer electronics, industry, telecom and networking) - global industry analysis, size, share, growth, trends, and forecast 2018 - 2026," 2018.

[109] X. Zhang and M. Tehranipoor, "Case study: Detecting hardware trojans in third-party digital ip cores," in *Proceedings of IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2011, pp. 67–70.

[110] M. Farag and M. Ewais, "Smart employment of circuit redundancy to effectively counter trojans (secret) in third-party ip cores," in *International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, Dec 2014, pp. 1–6.

[111] B. Salamat, T. Jackson, G. Wagner, C. Wimmer, and M. Franz, "Runtime defense against code injection attacks using replicated execution," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 4, pp. 588–601, 2011.

[112] R. S. Chakraborty, F. G. Wolff, S. Paul, C. A. Papachristou, and S. Bhunia, "MERO: A statistical approach for hardware trojan detection," in *International Workshop on Cryptographic Hardware and Embedded Systems*, 2009, pp. 396–410.

[113] N. Lesperance, S. Kulkarni, and K.-T. Cheng, "Hardware trojan detection using exhaustive testing of k-bit subspaces," in *Proceedings of Asia-Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2015, pp. 755–760.

[114] S. Ray, J. Yang, A. Basak, and S. Bhunia, "Correctness and security at odds: Post-silicon validation of modern soc designs," in *IEEE/ACM Design Automation Conference (DAC)*. IEEE, 2015, pp. 1–6.

[115] S. Wei, K. Li, F. Koushanfar, and M. Potkonjak, "Provably complete hardware trojan detection using test point insertion," in *IEEE International Conference on Computer-Aided Design (ICCAD)*, 2012, pp. 569–576.

[116] J. Li and J. Lach, "At-speed delay characterization for ic authentication and trojan horse detection," in *Proceedings of IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2008, pp. 8–14.

[117] Y. Huang, A. Chattopadhyay, and P. Mishra, "Trace buffer attack: Security versus observability study in post-silicon debug," in *2015 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*. IEEE, 2015, pp. 355–360.

[118] Y. Huang, S. Bhunia, and P. Mishra, "Mers: Statistical test generation for side-channel analysis based trojan detection," in *The ACM Conference on Computer and Communications Security (CCS)*, 2016.

[119] M. Banga and M. S. Hsiao, "A novel sustained vector technique for the detection of hardware trojans," in *IEEE International Conference on VLSI Design*, 2009, pp. 327–332.

[120] S. Narasimhan, D. Du, R. S. Chakraborty, S. Paul, F. G. Wolff, C. A. Papachristou, K. Roy, and S. Bhunia, "Hardware trojan detection by multiple-parameter side-channel analysis," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2183–2195, Nov 2013.

[121] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, "Tesr: A robust temporal self-referencing approach for hardware trojan detection," in *Proceedings of IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2011, pp. 71–74.

[122] A. N. Nowroz, K. Hu, F. Koushanfar, and S. Reda, "Novel techniques for high-sensitivity hardware trojan detection using thermal and power maps," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 33, no. 12, pp. 1792–1805, 2014.

[123] D. Hély, J. Martin, G. D. P. Triana, S. P. Mounier, E. Riviere, T. Sahuc, J. Savonet, and L. Soundararadjou, "Experiences in side channel and testing based hardware trojan detection," in *IEEE VLSI Test Symposium (VTS)*, 2013, pp. 1–4.

[124] J. Zhang, G. Su, Y. Liu, L. Wei, F. Yuan, G. Bai, and Q. Xu, "On trojan side channel design and identification," in *IEEE International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 278–285.

[125] R. Rad, J. Plusquellic, and M. Tehranipoor, "A sensitivity analysis of power signal methods for detecting hardware trojans under real process and environmental conditions," *IEEE Transactions on VLSI Systems*, vol. 18, no. 12, pp. 1735–1744, Dec 2010.

[126] G. Di Natale, S. Dupuis, and B. Rouzeyre, "Is side-channel analysis really reliable for detecting hardware trojans?" in *DCIS'2012: XVII Conference on Design of Circuits and Integrated Systems*, 2012, pp. 238–242.

[127] Y. Cao, C.-H. Chang, and S. Chen, "A cluster-based distributed active current sensing circuit for hardware trojan detection," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2220–2231, 2014.

[128] S. Kelly, X. Zhang, M. Tehranipoor, and A. Ferraiuolo, "Detecting hardware trojans using on-chip sensors in an asic design," *Journal of Electronic Testing*, vol. 31, no. 1, pp. 11–26, 2015.

[129] C. Bao, D. Forte, and A. Srivastava, "Temperature tracking: toward robust run-time detection of hardware trojans," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 10, pp. 1577–1585, 2015.

[130] A. Davoodi, M. Li, and M. Tehranipoor, "A sensor-assisted self-authentication framework for hardware trojan detection," *IEEE Design & Test*, vol. 30, no. 5, pp. 74–82, 2013.

[131] Y.-M. Tsai, K.-Y. Huang, H. Kung, D. Vlah, Y. L. Gwon, and L.-G. Chen, "A chip architecture for compressive sensing based detection of ic trojans," in *2012 IEEE Workshop on Signal Processing Systems*. IEEE, 2012, pp. 61–66.

[132] A. Muhtaroglu, G. Taylor, and T. Rahal-Arabi, "On-die droop detector for analog sensing of power supply noise," *IEEE Journal of solid-state circuits*, vol. 39, no. 4, pp. 651–660, 2004.

[133] E. Alon, V. Stojanovic, and M. A. Horowitz, "Circuits and techniques for high-resolution measurement of on-chip power supply noise," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 4, pp. 820–828, 2005.

[134] Intel, "Intel core i7-4650u processor," http://ark.intel.com/products/75114/Intel-Core-i7-4650U-Processor-4M-Cache-up-to-3-30-GHz.

[135] A. Waksman and S. Sethumadhavan, "Silencing hardware backdoors," in *IEEE/ACM International Symposium on Security and Privacy*, 2011, pp. 49–63.

[136] *Single-Phase Wide VIN Range DC/DC Controller for Intel IMVP-6/IMVP-6.5 CPUs*, http://cds.linear.com/docs/en/datasheet/3816f.pdf.

[137] T. E. Carlson, W. Heirman, and L. Eeckhout, "Sniper: exploring the level of abstraction for scalable and accurate parallel multi-core simulation," in *International Conference on Supercomputing*, 2011.

[138] V. Pallipadi and A. Starikovskiy, "The ondemand governor," in *Proceedings of Linux Symposium*, vol. 2, 2006, pp. 215–230.

[139] U. G. Nawathe, M. Hassan, K. C. Yen, A. Kumar, A. Ramachandran, and D. Greenhill, "Implementation of an 8-core, 64-thread, power-efficient sparc server on a chip," *Journal of Solid-State Circuits (JSSC)*, vol. 43, no. 1, pp. 6–20, 2008.

[140] A. Chakraborty, K. Duraisami, A. V. Sathanur, P. Sithambaram, L. Benini, A. Macii, E. Macii, and M. Poncino, "Dynamic thermal clock skew compensation using tunable delay buffers," in *ACM International Symposium on Low Power Electronic Devices (ISLPED)*, 2006, pp. 162–167.

[141] S. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "Varius:a model of process variation and resulting timing errors for microarchitects," *IEEE Transactions on Semiconductor Manufacturing*, vol. 21, pp. 3 –13, 2008.

[142] E. G. Inc., "Enterprise cloud adoption survey 2014: Summary of results," March 2014, http://www.everestgrp.com/wp-content/uploads/2014/03/2014-Enterprise-Cloud-Adoption-Survey.pdf.

[143] C. Colwill, "Human factors in information security: The insider threat–who can you trust these days?" *Information security technical report*, vol. 14, no. 4, pp. 186–196, 2009.

[144] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience*, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[145] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurr. Comput. : Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012. [Online]. Available: http://dx.doi.org/10.1002/cpe.1867

[146] K. Park and V. S. Pai, "Comon: A mostly-scalable monitoring system for planetlab," *SIGOPS*, Jan. 2006. [Online]. Available: http://doi.acm.org/10.1145/1113361.1113374

[147] X. Fu, X. Wang, and C. Lefurgy, "How much power oversubscription is safe and allowed in data centers," in *ACM Proceedings on International Conference on Autonomic Computing*, 2011.

[148] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[149] C. Tan, Y. Xia, H. Chen, and B. Zang, "Tinychecker: Transparent protection of vms against hypervisor failures with nested virtualization," in *IEEE/IFIP International Conference on DSN*. IEEE, 2012, pp. 1–6.

[150] M. Ben-Yehuda, M. D. Day, Z. Dubitzky, M. Factor, N. Har'El, A. Gordon, A. Liguori, O. Wasserman, and B.-A. Yassour, "The turtles project: Design and implementation of nested virtualization." in *USENIX-OSDI*, 2010, pp. 423–436.

[151] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

CURRICULUM VITAE

Rajesh Jayashankara Shridevi

Published Journal Articles

- IcoNoClast: Tackling Voltage Noise in the NoC Power Supply Through Flow-Control and Routing Algorithms, Basu, Prabal, Rajesh JS, Koushik Chakraborty, and Sanghamitra Roy, *IEEE Transactions on Very Large Scale Integration Systems*, (2017)
- Securing Data Center Against Power Attacks, JS, Rajesh, Rajamanikkam, Chidhambaranathan, Chakraborty, Koushik, Roy, Sanghamitra. *Journal of Hardware and Systems Security.* (2019)
- Security Measures Against a Rogue Network-on-Chip, JayashankaraShridevi, Rajesh, Dean Michael Ancajas, Koushik Chakraborty, and Sanghamitra Roy, *Journal of Hardware and Systems Security*, (2017).

Published Conference Papers

- Tackling voltage emergencies in NoC through timing error resilience, JS, Rajesh, Dean Michael Ancajas, Koushik Chakraborty, and Sanghamitra Roy, *IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*,(2015).
- Runtime detection of a bandwidth denial attack from a rogue network-on-chip, JS, Rajesh, Dean Michael Ancajas, Koushik Chakraborty, and Sanghamitra Roy, *Proceedings of the 9th International Symposium on Networks-on-Chip (NOCS)*, (2015).
- Catching the flu: Emerging threats from a third party power management unit, JS, Rajesh, Chidhambaranathan Rajamanikkam, Koushik Chakraborty, and Sanghamitra Roy. *Proceedings of the 53rd Annual Design Automation Conference (DAC)*, (2016).

- BoostNoC: Power efficient network-on-chip architecture for near threshold computing, Rajamanikkam, Chidhambaranathan, Rajesh JS, Koushik Chakraborty, and Sanghamitra Roy, *Proceedings of the 35th International Conference on Computer-Aided Design* (2016).

- PRADA: combating voltage noise in the NoC power supply through flow-control and routing algorithms, Basu, Prabal, Rajesh JS, Koushik Chakraborty, and Sanghamitra Roy, *Proceedings of the 2016 Conference on Design, Automation & Test in Europe*, (2016).