On a New Hardware Trojan Attack on Power Budgeting of Many Core Systems

Yiming Zhao*, Xiaohang Wang*, Yingtao Jiang†, Yang Mei†, Amit Kumar Singh‡ and Terrence Mak§
*School of Software Engineering, South China University of Technology, Guangzhou, China,
{201530661741@mail, xiaohangwang@}.scut.edu.cn
†Department of Electrical and Computer Engineering, University of Nevada, Las Vegas, USA,
{yingtao.jiang, mei.yang}@unlv.edu
‡School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK, a.k.singh@essex.ac.uk
§School of Electronics and Computer Science, University of Southampton, Southampton, UK, tmak@ecs.soton.ac.uk

***Abstract*—In this paper, we study stealthy false-data attacks that exploit the vulnerabilities of power budgeting scheme in NoC, which can cause catastrophic denial of service (DoS) effects. Essentially, when a power budget request packet is routed through a Trojan-infected network-on-chip node's router, the power budget request can be unknowingly modified. The global manager then tends to make really bad power budget allocation decisions with all the tampered power requests it received. That is, legitimate applications will be victimized with lower power budgets than what they initially asked for, and thus, could suffer serious performance degradation; malicious applications, on the other hand, may be entitled to high power budgets and thus see performance boost that they do not deserve. Our study has shown that this new type of DoS attack can be initiated and sustained by a simple hardware Trojan (HT) circuit that is extremely hard to be detected. The effects of this new DoS attack are simulated using a network model, and all the major parameters and factors that impact the attack effects are identified and quantified.**

***Index Terms*—power budgeting, Hardware Trojan**

I. Introduction

Security of servers, data centers, mobile computing and Internet of Things is largely dependent on the security of the many-core chips that these facilities/systems have adopted. Unfortunately, many-core chips are susceptible to be attacked by hardware Trojans (HT) which can be easily implanted to the chips at any stage from design to manufacturing, as chip designing and manufacturing is going global, and licensing of third party IP cores is becoming a commonplace in many-core chip designs. In contrast to a modern many-core chip with billions of transistors and complex functionalities, an HT circuit has an extremely low transistor count, making it hardly visible, and thus difficult to be detected by most known offline HT detection methods [1], [2]. If many-core devices infected with HTs finally get deployed, they can create catastrophic effects, including dangerous security breach and severe performance degradation [3], [4]. HTs have been designed to explore many different types of vulnerabilities of a many-core chip that may have, and in this paper, we show one such vulnerability that roots in the power budgeting scheme adopted in the chip.

Power budgeting is necessary in many-core chips [5], as the total power budget typically is not sufficient to meet the power needs for all the cores to run at their peak performance simultaneously. Given the fact that the power available to the cores is limited, the core designated as the global manager allocates power among all the cores [6], [7]. To make a fair and optimized power budgeting decision, the global manager needs to solicit budget requests from all the threads/applications running at different cores. Each power budget request is packetized and routed to the global manager through the chip's networked communication infrastructure, known as the network-on-chip (NoC).

Provided the hacker's agents can gain access to the global manager through a simple hardware Trojan embedded in an NoC router, the power budgeting system can be then attacked where power requests initiated by various cores are deliberately modified for harm. That is, power requests from the malicious applications (legitimate applications) will be increased (or decreased) to be higher (or lower) value than what were actually requested. Upon receiving the manipulated power budget requests, the global manager, irrespective of the power budgeting algorithms [8], [9] it runs, always allocates power budgets to favor the malicious applications, but penalize the legitimate applications. As a result of such a stealth (a.k.a. false-data) attack against the power-budgeting scheme, a new many-core chip may experience significant performance degradation and even complete malfunctioning.

In this paper, we present a DoS attack caused by HT and its circuit design that can launch and sustain aforementioned stealth attack against the power-budgeting scheme. As effectiveness of HT-enabled DoS attacks can be attributed to multiple factors, an attack model is built in this paper to quantify their impacts.

The rest of this paper is organized as follows. Section II introduces the background information and surveys relevant previous studies. Section III provides a design of hardware Trojan that enables a DoS attack that hackers tamper and steal power budgets from other threads. Section IV defines the related system parameters of the attack model, and section V reports the simulation results under different attack

This research program is supported by Natural Science Foundation of Guangdong Province 2018A030313166, the Science and Technology Research Grant of Guangdong Province No. 2017A050501003, Pearl River S&T Nova Program of Guangzhou No. 201806010038.

scenarios using the proposed attack model. Finally, Section VI concludes the paper.

II. Background and Previous Studies

In this section, we will first review the power budgeting schemes that have been applied to many core systems. We then categorize various hardware Trojan enabled DoS attacks seen in many-core systems.

A. Power Budgeting in Many-Core Systems

We assume a many-core system of interest follows a tiled architecture where each tile consists of a core, an L1 cache, an L2 cache bank, and a router [10]. Each core can operate at any of the preset frequencies, and a higher frequency leads to higher performance at a cost of higher power consumption. As the total power budget is limited, power budgeting aims to optimize the overall performance by choosing an appropriate frequency for each core. The decisions on frequency selection, or equivalently power allocation, are made by a special core designated as the global manager [6], [11]. To make an optimized power budgeting decision, the global manager needs to solicit budget requests from all the threads/applications running at different cores. Power allocation problem can be solved following many different strategies, such as heuristics [8], control theory [12] or dynamic programming [9].

B. DoS Attack

DoS attack enabled by hardware Trojan (HT) is becoming a serious threat to modern many-core chips [3], [13]. Various circuit level HT detection techniques have been proposed, based on logic testing and side channel analysis during the post-silicon test/validation process [1], [14]. Most HTs have tiny area compared to the whole chip, the HTs detection methods cannot detect all the HTs.

DoS attacks on a many-core chip can target different components of the chip, including the memory system [15], and the network-on-chip (NoC) [16], [17]. With the help of hardware Trojans (HTs) [3], [4], [13], DoS attacks can be classified as 1) *flooding attack* [18], [19], where a large volume of useless packets floods a victim node and saturates it; 2) *packet drop attack*, where some packets are dropped or directed to some malicious nodes so that the victim node can never receive a single packet designated to it [20]; 3) *privilege escalation attack* [21], where an average user process is granted the privileges of a supervisor so that it can steal passwords; and 4) *routing loop attack* [2], where packets that pass the malicious node will be routed back to the source node, effectively blocking the source core from communicating with any other cores.

III. HT-Enabled DoS Attacks Targeting the Power Budgeting System

In this section, we show that proposed DoS attack can be launched targeting the power budgeting scheme of a many-core chip by implanting HTs in NoC's routers. And a detailed design of such hardware Trojan is shown.

A. Packet Frames

Generally speaking, a data packet arriving at a network node or a router has 4 fields: the source address (16 bits in this study), the destination address (16 bits), packet type (32 bits) and payload field (32 bits). Additional information, if needed, can be included into an optional field that is named so.

As shown in Fig. 1(a), a packet will be recognized as a power request packet if its packet type field is POWER_REQ. In this case, the payload of the packet is the power request value.

A packet will be recognized as a Trojan configuration packet if its packet type is CONFIG_CMD as shown in Fig. 1(b). For a configuration packet, its source address is actually the attacker's ID and the packet type field includes the CONFIG_CMD, global manager and activation signal. A configuration packet is meant to be sent to an implanted HT core by attacker to set up the HT for attack.

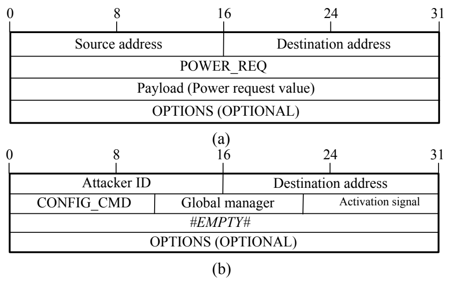


Fig. 1: (a) Normal power request packet. (b) Attackers configuration packet.

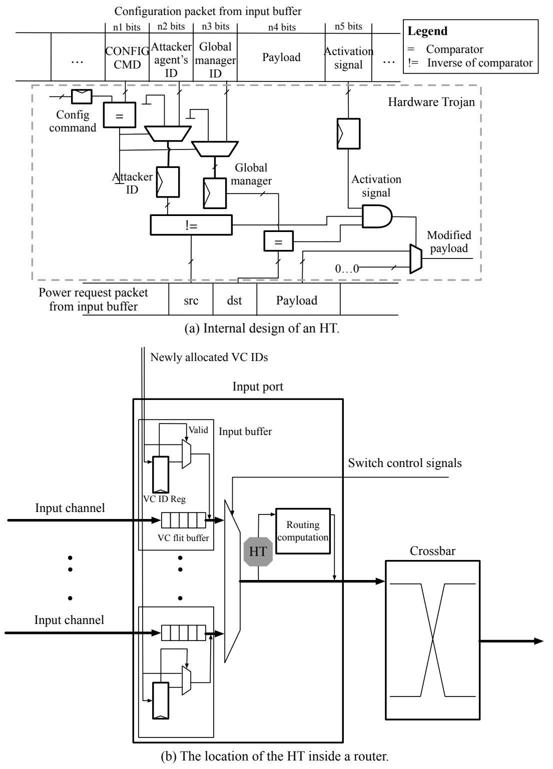
B. HT Attack Process

When a hacker is about to attack, it broadcasts the configuration packet that includes the global manager's ID, its own core ID and the activation signal. When the configuration packet arrives at the infected nodes, the HT stores the global manager's ID and the attacker's ID in its local registers, if it has not done so. The HT's activation state is set by the activation signal in the configuration packet. The hacker can launch different attack modes by setting the right activation signal, and the configuration packets may be sent out periodically for the control of attack strategies. For example, if the attacker agents want the HTs to be active in a specific cycle time, a series of configuration packets can be sent with activation signals alternated to be ON and OFF.

After configuration, when a victim's data packet passes through those routers with implanted HT, if the HT is not activated, the packet is forwarded normally (i.e., no modification to the packet will ever be made). Otherwise, the HT checks whether the data packet's destination is the global manager and the source is not hacker's agent. If so, the triggering module triggers the functional module to modify the value of the power request field.

C. HT Circuits Implementation

To launch the proposed HT-enabled attack, an HT circuit includes some registers to store the configuration parameters from the hackers' agents. The triggering module scans data packets, and if a packet is found to match the configuration parameters stored in the HT's local register, the functional module is then enabled if HT is activated. The functional module basically manipulates the payload of the packet that the power request is changed to a smaller value. As Fig. 2 shows, an HT has 3 comparators and 2 registers that sit between the router's input buffer and the routing computation module.



(a) Internal design of an HT.

(b) The location of the HT inside a router.

Fig. 2: Hardware Trojan Design

D. HT's Area and Power

Each HT has an area of 12.1716μm^2 and consumes 0.55018μW power, as reported by Synopsys Design Compiler under 45nm TSMC library. As a comparison, a router with 4 virtual channels and 5-flit depth First-Input-First-Output has a total area of 71814 μm^2 and consumes a total power of 31881μW, obtained from DSENT. In a simple term, an HTs area and power is about 0.017% and 0.0017% of a single router, making such an HT hard to be detected. For a whole chip, such as section V-C shows, 60 HTs almost infect all packets in a 512-node chip, the HTs' area is about 730.296 μm^2 and consume 33.0108μW power. 60 HTs's area and power is about 0.002% and 0.0002% of all routers in a 512-node chip.

IV. Evaluating the DoS Attack Effect

Aforementioned DoS attack tends to cause victim applications/threads to suffer from serious performance degradation while attacker's malicious application may see considerable performance gain. The effectiveness of the proposed DoS attack can be attributed to a number of factors listed below:

- Number of hardware Trojans in the NoC.
- Locations of the hardware Trojans and the global manager. When an HT is close to the global manager, since more budget request packets are likely to pass the HT before reaching the global manager, this HT can make more packet modifications, and thus can have higher impacts on the attack effect.
- Application's sensitivity to its power budget. Performance impacts due to power budget changes can be application-specific. For example, performance of an instruction-bounded application is typically hit harder than that of memory-bounded applications.

A. Measures of Attack Effect

Definition 1 *Application k's performance θ_k is defined by*

$$\theta_k = \sum_{j \in \mathcal{C}_k} IPC(j,k,f_j) \cdot f_j \tag{1}$$

where θ_k is application k's instruction per clock (IPC) value, $\mathcal{C}_k$ is the set of cores running application k's threads, and $IPC(j,k,f_j)$ is core j's IPC.

Definition 2 *Application k's performance change Θ_k is defined as*

$$\Theta_k = \frac{\theta_k}{\Lambda_k} \tag{2}$$

where θ_k is the application k's IPC value with HTs and Λ_k is the application k's IPC without HTs.

Definition 3 *The attack effect of the proposed DoS attack $Q(\Delta,\Gamma)$ is defined as*

$$Q(\Delta,\Gamma) = \frac{V \cdot \sum_{a\in\Delta} \Theta_a}{A \cdot \sum_{v\in\Gamma} \Theta_v} \tag{3}$$

where V and A are the numbers of victims and attackers respectively. Δ and Γ are the sets of attacker applications and victim applications respectively.

If attacker's performance improves, or victim's performance degrades, the value of $Q(\Delta,\Gamma)$ would increase as well. In a simple word, the larger value $Q(\Delta,\Gamma)$ has, the stronger an attack is.

B. Factors Impacting Attack Effect

There are a number of factors that can impact the attack effects.

1) Application's sensitivity to power budget: Applications have different performance changes when their V/F values vary.

Definition 4 *Core j's sensitivity to power budget while running application z, denoted as $\phi(j,z)$ is defined by*

$$\phi(j,z) = \sum_{i=1}^{s-1} \left| \frac{IPC(j,z,\tau_i) - IPC(j,z,\tau_{i+1})}{\tau_i - \tau_{i+1}} \right| \tag{4}$$
$$\tau_1 < \tau_2 < \cdots < \tau_s$$

where τ_i and τ_{i+1} are the available frequency levels, $IPC(j, z, \tau_i)$ is core j's IPC when it is running application z's thread with a frequency level of τ_i.

Definition 5 *Application k's sensitivity to power budget is defined by*

$$\Phi_k = \frac{\sum_{i \in \mathcal{C}_k} \phi(i, k)}{|\mathcal{C}_k|} \tag{5}$$

where $\mathcal{C}_k$ is the set of cores running application k's threads.

2) Parameters characterizing system architecture: The distribution of HTs has performance implications on both attackers and victims.

Definition 6 *Assume there are a total of m malicious nodes. The coordinates of the m HTs' virtual center is defined by*

$$\omega_X = \frac{\sum_{i=1}^{m} X_{M_i}}{m}, \omega_Y = \frac{\sum_{i=1}^{m} Y_{M_i}}{m} \tag{6}$$

where X_{M_i} and Y_{M_i} are x and y coordinates of the malicious nodes' respectively.

Definition 7 *Distance between the global manager and the virtual center of those HTs denoted as ρ is defined by*

$$\rho = \mathbf{MD}(O, \Omega) \tag{7}$$

where O is the global manager's location, and Ω is the HTs' virtual center.
If the virtual center of HTs is far away from the global manager, measured as long distance between the two, fewer packets with power requests actually pass through the nodes infected with HTs. In other words, performance of the system has a lower probability to be affected.

Definition 8 *HTs' density denoted as η is the average Manhattan distance between HTs' virtual center and each malicious nodes. It measures the variance of the HT distribution, which is defined as*

$$\eta = \frac{\sum_{i=1}^{m} \mathbf{MD}(\Omega, M_i)}{m} \tag{8}$$

where m is the number of malicious nodes.
A higher density indicates a large number of malicious nodes are around the virtual center, and more packets may be intercepted and modified, leading to a higher infection rate.

C. Modeling Attack Effects

Attack effect can be quantified using a linear model as follows.

$$Q(\Delta, \Gamma) \approx a_1 \times \rho + a_2 \times \eta + a_3 \times m + \sum_{j=1}^{V} b_j \times \Phi_{\gamma_j} + \sum_{k=1}^{A} c_k \times \Phi_{\delta_k} + a_0 \tag{9}$$

where $a_i, i = 0, 1, 2, 3,\ b_j, c_k,\ 1 \leqslant j \leqslant V, 1 \leqslant k \leqslant A$ are the regression coefficients, γ_j and δ_k are jth and kth victim/attacker application, ρ is the HT's distance between HTs' virtual center and the global manager, η is the HT's distribution density, m is the number of malicious nodes, Φ_{γ_j} and Φ_{δ_k} are the victim and the attacker applications' sensitivities to power budget, respectively.

Following the attack effect model, we formulate a problem to maximize the attack effect by selecting the proper distance and density of HTs. The constraints are the area of the HT circuits, or equivalently, the number of HTs that will be selected by the attacker. This attack effect optimization problem can be formulated as follows.

$$\max_{\rho, \eta, m} \quad Q(\Delta, \Gamma) \tag{10}$$

$$\text{subject to} \quad m \leqslant M_{HT} \tag{11}$$

where M_{HT} is the maximum number of malicious nodes selected by the hackers.

To solve the problem, one can exhaustively enumerate all possible values for above mentioned three metrics: 1) number of HTs, 2) distance between the global manager and the virtual center of HTs, 3) HTs distribution density.

V. Experimental Results

A. Experimental Setup

Experiments are performed to evaluate the attack effect with the number of HTs and HTs' distribution. The experiments are using an event-driven many-core simulator in C++ [22]. The simulated architecture is a shared memory structure, with each core having its own L1 cache and a shared L2 cache bank [10]. A tile is connected to a local network interface and a router, and together they form one node of the NoC. Table I lists the simulator configuration. Multi-threaded applications can have their threads running on different cores. Communications between threads take place through the NoC. The messages in the network are generated by memory transactions including read/write access requirements. Table II lists the benchmarks used for performance evaluation, and they are selected from PARSEC and SPLASH-2. In the following experiments, we select a 16 $\times$ 16 2D mesh with adaptive routing as the underline NoC architecture.

B. Evaluating the Infection Rate

Fig. 3 (a) compares the infection rates when the global manager is at different locations and the system size is 64. One can see that along with the increase of the number of HTs, the infection rate increases as well. If the global manager is placed at the corner of the chip, the corresponding

TABLE I: Configuration used in the simulation

Number of processors	256 (Alpha ISA 64 compatible)
Fetch/Decode/Commit size	4/4/4
ROB size	64
L1 D cache(private)	16 KB, two-way, 32B line, two cycles, two ports, dual tags
L1 I cache(private)	32 KB, two-way, 64B line, two cycles
L2 cache(shared) MESI protocol	64 KB slice/node, 64B line six cycles, two ports
Main memory size	2 GB, latency 200 cycles
On-chip network parameters	
NoC flit size	72-bit
Data packet size	5 flits
Meta packet size	1 flit
NoC latency	router two cycles, link one cycle
NoC Virtual Channel (VC) number	4
NoC buffer	5 × 5 flits
Routing algorithm	XY Routing

TABLE II: Benchmarks used in the simulation

PARSEC	streamcluster, swaptions, ferret, uidanimate, blackscholes, freqmine, dedup, canneal, vips
SPLASH-2	barnes, raytrace

infection rate is more than 20% higher than that of the case where the global manager is placed at the center, assuming there are more than 10 HTs. When the system size is 512, a similar trend is observed as shown in Fig. 3 (b). The reason is that a packet loaded with a power budget request has to travel a long distance to reach the global manager that is at the corner of the chip. Longer routing distance increases the chance that such a packet is intercepted and modified by an HT node.

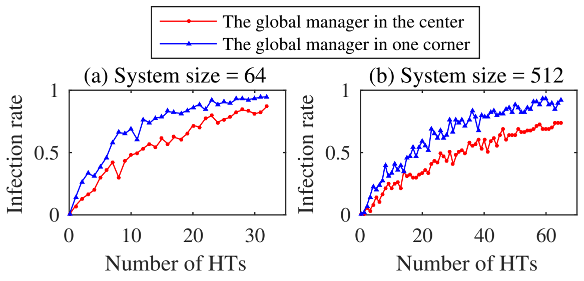


Fig. 3: Infection rate comparison with different HT numbers when the system size is (a) 64, and (b) 512.

Fig. 4 compares the infection rates of 3 different cases: (i) HTs are all placed close to the center of the chip, (ii) HTs are randomly distributed, and (iii) HTs are placed to a concentrated area near one corner. Here, the global manager is assumed to be at the center of the chip.

From Fig. 4, one can see that the infection rates of the cases that the HTs are near the center location of the chip are higher than those of the other two cases. For example, in Fig. 4 (a), when the system size is 256, the infection rate of the case that the HTs are close to the center is 1.59× and 9.85× more than those of the other two cases, respectively. The reason is that, in the case HTs are around the center, more packets with power requests are likely to be intercepted and modified. In the case when HTs are randomly distributed, fewer packets with power requests will be attacked. In the case when HTs are clustered around one corner of the chip, some packets with power requests will never be attacked.

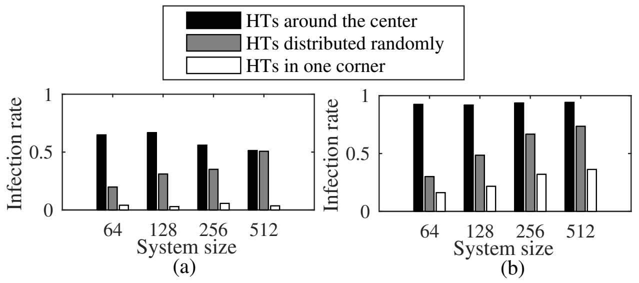


Fig. 4: Infection rate comparison with different HT distributions, when the number of HTs is (a) $\frac{1}{16}$, (b) $\frac{1}{8}$ of the system size.

C. *Evaluating Attack Effect*

We use the mixes of a few benchmarks to test the attack effects of the proposed DoS attack. The numbers of attackers and victims are set to be 1, 2, and 3, and they can be mixed to obtain four combinations, as tabulated in Table III.

TABLE III: Benchmark combinations used in the experiments

Combination	**Attackers**	**Victims**
Mix-1	barnes, canneal	blackscholes, raytrace
Mix-2	freqmine, swaptions	raytrace, vips
Mix-3	canneal	barnes, vips, dedup
Mix-4	barnes, streamcluster, freqmine	raytrace

Fig. 5 shows each mix's Q value with respect to infection rate. Each application is set to have 64 threads and run on a chip with 256 cores. Overall speaking, a higher infection rate leads to a larger Q value. In the case of Mix-4 which has three attackers, the Q value reaches its peak (i.e., 6.89) at the infection rate of 0.9. In Fig. 6 (a), one can see that when the infection rate is 0.5, the performance of the attackers is improved by as much as 1.2×, and the victim's performance drops by 0.6×. In Fig. 6 (c), when there are 3 victims, and the infection rate is 0.5, the performance improvement of the attacker is as much as 1.35×. In Fig. 6 (d), when there are 3 attackers, and the infection rate remains at 0.5, the victims' performance degrades by as much as 0.8×.

Next, the attack effect of an NoC system with HTs optimally placed (select number of HTs, distance between the global manager and virtual center of HTs, HTs distribution density which solve Eqns. 10) is compared with that of a system with randomly placed HTs. In the case that there are 16 HTs in the chip and the global manager is in the center of the chip, the attack effect of the NoC with optimal

HT distribution is about 30% higher than that of NoC but with a random HT distribution for the mixes of 1, 2 and 3. More significant improvement (by as much as 110%) in attack effect is seen in the case of mix-4. In a simple word, solving the attack effect maximization problem in Eqns. 10 can indeed improve the attack effects substantially.

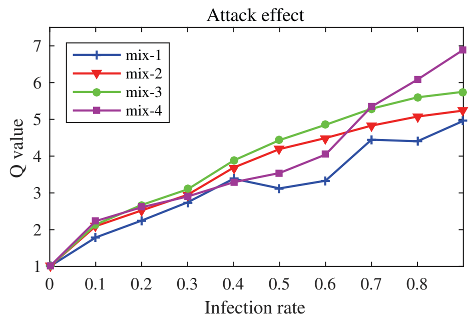


Fig. 5: Attack effect comparison with different infection rates.

VI. Conclusion

In this paper, we proposed a new HT-enabled DoS attack that hackers can steal power budget from other applications/threads in many-core chips. Once the power request packet from a victim application/thread traverses through a malicious router infected with a special HT, the power request is modified. As a result, the victim applications suffer from poorer performance due to lower power budgets that they were granted to operate on, while the malicious applications/threads can be assigned with excessive power budgets. The effects of the DoS attack include performance degradation of the victims and performance boost of the malicious applications/threads, both of which were found to be related to various system parameters in this paper. This new DoS attack can cause catastrophic consequences, and more research on detection and protection against such attacks is needed.

References

[1] K. Chrysanthou, P. Englezakis, A. Prodromou, A. Panteli, C. Nicopoulos, Y. Sazeides, and G. Dimitrakopoulos, “An online and real-time fault detection and localization mechanism for network-on-chip architectures,” *ACM Trans. Architecture and Code Optimization*, vol. 13, no. 2, pp. 22:1–22:26, 2016.

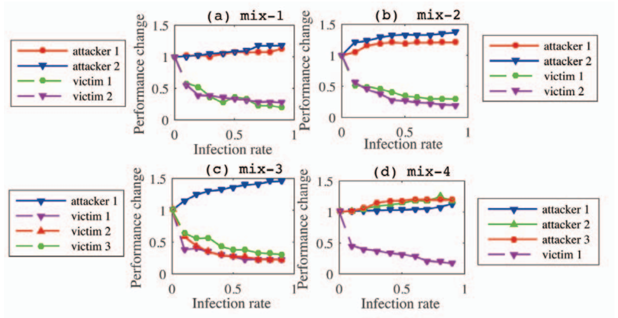


Fig. 6: Applications’ performance changes for each mix.

[2] A. Kulkarni, Y. Pino, and T. Mohsenin, “Svm-based real-time hardware trojan detection for many-core platform,” in *Int’l Symp. Quality Electronic Design (ISQED)*, 2016, pp. 362–367.

[3] H. Li, Q. Liu, and J. Zhang, “A survey of hardware trojan threat and defense,” *Integration, the VLSI Journal*, vol. 55, pp. 426–437, 2016.

[4] W. Burleson, O. Mutlu, and M. Tiwari, “Invited-who is the major threat to tomorrows security?: You, the hardware designer,” in *Proc. Design Automation Conf.*, 2016, pp. 1–5.

[5] A. Rezaei, D. Zhao, M. Daneshtalab, and H. Wu, “Shift sprinting: fine-grained temperature-aware noc-based mcsoc architecture in dark silicon age,” in *Proc. Design Automation Conf.*, 2016, pp. 155:1–155:6.

[6] X. Wang and J. F. Martínez, “Rebudget: trading off efficiency vs. fairness in market-based multicore resource allocation via runtime budget reassignment,” *ACM SIGPLAN Notices*, vol. 51, no. 4, pp. 19–32, 2016.

[7] A. Sharifi, A. K. Mishra, S. Srikantaiah, M. Kandemir, and C. R. Das, “PEPON: performance-aware hierarchical power budgeting for NoC based multicores,” in *Proc. Int’l Conf. Parallel Architectures and Compilation Techniques*, 2012, pp. 65–74.

[8] X. Li, G. Yan, Y. Han, and X. Li, “Smartcap: user experience-oriented power adaptation for smartphone’s application processor,” in *Proc. Conf. Design, Automation and Test in Europe*, 2013, pp. 57–60.

[9] X. Wang, B. Zhao, T. Mak, M. Yang, Y. Jiang, and M. Daneshtalab, “On fine-grained runtime power budgeting for networks-on-chip systems,” *IEEE Trans. Computers*, vol. 65, no. 9, pp. 2780–2793, 2016.

[10] X. Wang, B. Zhao, L. Wang, T. Mak, M. Yang, Y. Jiang, and M. Daneshtalab, “A pareto-optimal runtime power budgeting scheme for many-core systems,” *Microprocessors and Microsystems*, vol. 46, pp. 136–148, 2016.

[11] S. M. Zahedi and B. C. Lee, “Ref: resource elasticity fairness with sharing incentives for multiprocessors,” *ACM SIGARCH Computer Architecture News*, vol. 42, no. 1, pp. 145–160, 2014.

[12] K. Ma and X. Wang, “Pgcapping: exploiting power gating for power capping and core lifetime balancing in cmps,” in *Proc. Int’l Conf. Parallel Architectures and Compilation Techniques*, 2012, pp. 13–22.

[13] S. K. Haider, C. Jin, and M. van Dijk, “Advancing the state-of-the-art in hardware trojans design,” *arXiv preprint arXiv:1605.08413*, 2016.

[14] M. Hussain and H. Guo, “Packet leak detection on hardware-trojan infected nocs for mpsoc systems,” in *Proc. Int’l Conf. Cryptography, Security and Privacy*, 2017, pp. 85–90.

[15] T. Moscibroda and O. Mutlu, “Memory performance attacks: denial of memory service in multi-core systems,” in *Proc. Symp. USENIX Security*, 2007, pp. 18:1–18:18.

[16] J. Frey and Q. Yu, “A hardened network-on-chip design using runtime hardware trojan mitigation methods,” *Integration, the VLSI Journal*, vol. 56, pp. 15–31, 2017.

[17] L. S. Indrusiak, J. Harbin, and M. J. Sepulveda, “Side-channel attack resilience through route randomisation in secure real-time networks-on-chip,” in *Int’l Symp. Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, 2017, pp. 1–8.

[18] T. Boraten and A. K. Kodi, “Mitigation of denial of service attack with hardware trojans in noc architectures,” in *Proc. IEEE Int’l Symp. Parallel and Distributed Processing*, 2016, pp. 1091–1100.

[19] C. Gómez, M. E. Gómez, P. López, and J. Duato, “Reducing packet dropping in a bufferless noc,” in *Proc. European Conf. Parallel Processing*, 2008, pp. 899–909.

[20] A. Kulkarni, Y. Pino, and T. Mohsenin, “Adaptive real-time trojan detection framework through machine learning,” in *IEEE Int’l Symp. Hardware Oriented Security and Trust (HOST)*. IEEE, 2016, pp. 120–123.

[21] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, “Designing and implementing malicious hardware.” *Large-Scale Exploits and Emergent Threats (LEET)*, vol. 8, pp. 1–8, 2008.

[22] X. Wang, M. Yang, Y. Jiang, P. Liu, M. Daneshtalab, M. Palesi, and T. Mak, “On self-tuning networks-on-chip for dynamic network-flow dominance adaptation,” *ACM Trans. Embedded Computing Systems (TECS)*, vol. 13, no. 2s, pp. 73:1–73:21, 2014.