On the Simulation of Processors Enhanced for Security in Virtualization

Swapneel C. Mhatre
Dept. of Computer Science & Engg.
National Institute of Technology
Calicut, Kerala, India
mhatreswapneel_p170067cs@nitc.ac.in

Priya Chandran
Dept. of Computer Science & Engg.
National Institute of Technology
Calicut, Kerala, India
priya@nitc.ac.in

Jithin R.
Dept. of Computer Science & Engg.
National Institute of Technology
Calicut, Kerala, India
jithinr550@gmail.com

ABSTRACT

Computer system simulators model the hardware and reduce the time required for the design of the hardware, by exploring the design space and thereby eliminating the time-consuming process of testing each possibility by actually building the hardware. Simulators used in Computer Architecture typically model processors, memories and disks. With the regaining of the importance of virtualization, there is a need for a simulator for virtualization-enhanced processors, especially, with security related enhancements. But adapting existing simulators to model virtualization-enabled processors is a deceptively difficult task, wrought with complications like multiple access levels. In our research, we aim to identify methods for effectively simulating virtualization-enabled processors. This paper reports the results of a preliminary simulation of Architectural Support for Memory Isolation (ASMI), a memory architecture model that provides memory isolation, using ModelSim and highlights the need for a hardware-based simulator for processors enhanced for security in virtualization.

CCS CONCEPTS

• **Computing methodologies → Modeling and simulation**;

KEYWORDS

Simulation, Computer Architecture, virtualization, security, multikernel simulation, Architectural Support for Memory Isolation (ASMI)

1 INTRODUCTION

Simulation is crucial to research in Computer Architecture. A simulator evaluates and predicts the performance of a computer system.

Virtualization [5], [6] has regained importance in the past decade and hence, new processors are developed to support virtualization. Intel Virtualization Technology (VanderPool Technology) (VT) and AMD-V are some new technologies developed. Various hardware enhancements to provide security to virtual machines (VMs) have been proposed in the literature. These include HyperWall [7], Rollback Sensitive Data Memory with Architecture assistance (RSDM-A) and Extended-Hyperwall [4]. HyperWall uses Confidentiality Integrity Protection (CIP) table to provide security to VMs even in the presence of an untrusted hypervisor. RSDM-A uses Rollback Sensitive Data Memory (RSDM) table to separate rollback sensitive data from rollback non-sensitive data and prohibits rollback of rollback sensitive data. Extended-Hyperwall integrates CIP table and RSDM table to prevent rollback based attacks.

With many architectures being developed to support virtualization, there is an increasing demand for simulators to simulate and explore the design space of such architectures. However, it is challenging to develop such simulators, especially, if the instruction set modifications are done at different privilege levels.

Multikernel simulation proposed in [1] is an interesting step towards simulating processors enhanced for security in virtualization and has simulated Extended-HyperWall architecture [4]. The multikernel simulation approach simulates the instructions at different privilege levels by distributing the simulator code between the different privilege levels of the hypervisor or virtual machine monitor (VMM) and operating system kernels. It uses the unused bits in the kernel software to simulate hardware conditions.

Although multikernel simulation has successfully simulated Extended-HyperWall architecture and is able to simulate virtualization at the kernel level, there is a need for integration with lower levels to compare different performance parameters. These parameters include memory access time (with access check), time for rollback and so on. This gives rise to the need for a hardware-based simulator for virtualization.

Architectural Support for Memory Isolation (ASMI) proposed in [3] is a memory architecture model that provides memory isolation and hence security to each virtual machine (VM) even in the presence of an infected hypervisor by making modifications to memory management unit (MMU), central processing unit (CPU) and instruction set architecture (ISA) in Intel-VT architecture. ASMI has been verified through emulation in the hypervisor kernel [3].

This paper reports the results of a preliminary simulation of ASMI using ModelSim and brings out the need for a hardware-based simulator to model virtualization in a better way.

ICPE, April 2018, Germany
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM. . . $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Examples of Simulators

Category	Examples
Processor Simulators	SimpleScalar, Wattch
Memory Simulators	DRAMSim, NVMain
Disk Simulators	SSDcheck, SimpleSSD
Full-System Simulators	gem5, Simics

2 RELATED WORK

Table 1 shows some examples of simulators categorized as processor simulators, memory simulators, disk simulators and full-system simulators [8].

However, these simulators are written in high-level languages and run in the least privileged level. On the other hand, to support virtualization and to provide security in virtualization, processors use instructions executing at different privilege levels. Hence, these simulators cannot be used for modeling virtualization-enabled processors. Modifying these simulators to emulate different privilege levels would result in extremely complex and slow code.

The study of processors with hardware support for virtualization and virtualization security calls for integrating the simulation process with a lower level simulation to study the impact of the enhancement on the delays associated with the hardware enhancements, chip area, sizes, etc. As a first step towards such a study, a preliminary experiment is conducted by simulating ASMI using ModelSim.

3 CASE STUDY: SIMULATION OF ASMI USING MODELSIM

Memory access with ASMI is simulated using ModelSim with the code written in Verilog and the synthesis estimate is obtained using Xilinx Integrated Software Environment (ISE). The memory is modeled with m-bit address bus and n-bit data bus. Thus, memory is organized as $= \frac{2^m X 8}{n} X n$. In ASMI page access, the input is the physical address obtained from the page table and the output is non-zero if the access is allowed and zero if not. If the access is allowed, the physical memory is accessed to get the data. First, the segment ID (SegID) is calculated from the physical address, then the data memory (physical memory) is accessed to get the virtual machine ID (VMID) stored at the location pointed to by SegID in the physical memory and then the VMID obtained is compared with the value stored in the register called VMIDR, which is the identity of the currently running virtual machine on that processor. If the access is allowed, then the physical memory is accessed again to get the data. Three modules are implemented separately - *calculate_SegID* that takes m-bit physical address as the input and gives SegID as output, *data_memory* that has n-bit data bus and m-bit address bus and *compare* that compares VMID with VMIDR and decides whether the access is allowed.

4 RESULTS

For the purpose of simulation, the value of n, the size of SegID and the size of VMID are kept constant and the value of m is varied. For n = 8, size of SegID = 4, size of VMID = 4 and m = 4, 5 and 6, the synthesis estimates obtained for the modules *calculate_SegID*, *data_memory* and *compare* are 7.045 ns., 5.835 ns. and 8.627 ns. respectively. Hence, the time required to decide whether the access is allowed $= (7.045 + 5.835 + 8.627)\,ns. = 21.507\,ns.$ and the total time required to get the data $= (21.507 + 5.835)\,ns. = 27.342\,ns.$ If ASMI is not used, the physical memory is accessed directly and the time required to get the data = 5.835 ns. Thus, the percentage increase in time because of ASMI $= \frac{27.342-5.835}{5.835} X 100\,\% = 368.6\,\%$.

5 CONCLUSION AND FUTURE SCOPE

This paper reports the simulation results of ASMI using ModelSim and concludes that the percentage increase in time because of ASMI to provide memory isolation and security is 468.6 %. However, simulation using ModelSim with an application program executing on it is not only extremely slow but also requires tedious manual efforts to explore different architectural scenarios [2]. Hence, there is a need for more research on the integration of high-level simulation with a lower level tool like ModelSim to efficiently simulate architectural enhancements for virtualization and security in virtualization.

With the increasing demand for virtualization, there is a need for a hardware-based simulator for processor enhancements for security in virtualization including instruction set modifications done at different privilege levels. This will reduce the time required for design space exploration of such processors.

REFERENCES

[1] Lakshya Garg Dr. Priya Chandran and Aditya Kumar. 2017. Multikernel Simulation: A New Approach to Study Rollback Sensitive Memory Architecture. In *Proceedings of the 2017 IEEE 3rd International Conference on Collaboration and Internet Computing.* 437–442.

[2] Sharad Sinha Liang Feng, Hao Liang and Wei Zhang. 2017. HeteroSim: A Heterogeneous CPU-FPGA Simulator. *IEEE Computer Architecture Letters* 16, 1 (2017), 38–41.

[3] Jithin R. and Dr. Priya Chandran. 2016. Dynamic Partitioning of Physical Memory Among Virtual Machines [ASMI: Architectural Support for Memory Isolation]. In *Proceedings of the 31st Annual Symposium on Applied Computing.* 474–476.

[4] Payas Krishna Vinod Reddy Bodasingi Jayachandra Shubham Shoundic, Dr. Priya Chandran and Lakshit Pande. 2016. Extended HyperWall: Hardware Support for Rollback Secure Virtualization. In *Proceedings of the 2016 International Conference on Advances in Computing, Communication and Informatics (ICACCI).* 1674–1681.

[5] James E. Smith and Ravi Nair. 2005. The Architecture of Virtual Machines. *Computer* 38, 5 (May 2005), 32–38.

[6] James E. Smith and Ravi Nair. 2005. *Virtual Machines, Versatile Platforms for Systems and Processes.* Morgan Kaufmann Publishers.

[7] Jakub Szefer and Ruby B. Lee. 2012. Architectural Support for Hypervisor-Secure Virtualization. In *Proceedings of the ASPLOS'12.* 437–449.

[8] Joshua J. Yi and David J. Lilja. 2006. Simulation of Computer Architectures: Simulators, Benchmarks, Methodologies, and Recommendations. *IEEE Trans. Comput.* 55, 3 (2006), 268–280.