Analysis of Intel Pentium's Ability to Support a Secure Virtual Machine Monitor

John Scott Robin(US Air Force) Cynthia E Irvine (Naval Postgraduate School)

1. Introduction

I. Benefits of VM

VMM's used as testbeds.

Multiple OS

Isolate untrusted applications

OS upgradation and running of older OS simultaneously.

System Software for Scalable computers (e.g. Disco)

II. VMM Characteristics

Refer to Goldberg's paper [Formal Requirements for Virtualizable 3rd Generation Architectures].

Identical interface, identical to underlying hardware. Exception to this rule is the timing dependencies, reduced set of available resources and IO.

Control of System Resources.

Efficiency [A large percentage of VM's instructions executed natively].

III. VMM Modules

Dispatcher, Allocator and interpretor.

IV. Secure VMM Benefits

Whether Intel can provide the benefits of a secure VMM

2. VMM Requirements

a. VMM Types (Goldberg's Definitions)

CSIM -> Full emulation

VMM-> Statistically dominant set of instructions executed natively

Hybrid VMM -> All privileged instructions are emulated.

Based on how VMM gets control of the machine for executing those instructions that can't directly be executed by a VM, VMM's classified as type I and type II.

b. Definitions

Definitions of sensitive (those instructions that can not be directly executed on hardware)and privileged instructions (trap to hardware when executed from a lower privileged domain). When all sensitive instructions are privileged, then virtualizable architecture. **Sensitive non-privileged** instructions cause problems for VMM.

c. Virtualizable Requirements for Type-I and Type-II VMM.

Any questions?? Discussions??

Hybrid VMM, second requirement, any comments??

3. **Intel's Ability to support virtualization:**

   Rule 2, 3A, 3B, 3C, 3D (intersection for requirements of type-I, type-II and HVM) makes processor non-virtualizable.

   Intel satisfied requirement 1 & 2. However, it has 17 sensitive-unprivileged instructions that do not trap when executed from a unprivileged domain.

   All 17 instructions are next discussed in detail. Any questions?

   What is RPL, DPL, CPL?? Refer section 4.5 in Intel's System Programmer's Manual 3A.

   Refer to the following sections in Intel's System Programmer's Manual 3A for the understanding the following sections of the paper:

   Section 3.2.3, refer section 3.4.2 in manual.

   Section 3.2.4, refer section 4.8.3 and 4.8.4 in manual

   Section 3.2.5, refer section 4.10.4 in manual

4. **Secure VMM's in Pentium and other architectures like VAX and Alpha.**

   4.3: How to Virtualize Pentium architecture.

   a. Pure emulation

   b. Os/API emulation like Wine, emulates a different OS for applications.

   c. Virtualization of sensitive unprivileged instructions. Analyze instructions, set breakpoints before problem instructions, inspect code and take necessary steps.

      Discuss issues related to jumping into an already analyzed block of code.

      Other virtualization issues.

      (**Section 4.3.3**) Virtualized GDT and IDT.

      (**Section 4.3.4**) Virtual CPU's, Physical Memory, Paravirtualization for IO devices, Network

   4.4: Security Issues with type-II VMM's. -> Mostly trivial

   4.5: How to make Pentium's more secure??

   Device drivers? Implement all of them or just certain specific types?

   Suggestions:

   1. **Change all 17 instructions to privileged. -> backward incompatible.**
   2. **A new instruction before executing all 17 instructions that make them privileged.**

   Discussion: Vanderpool architecture – What Intel Plans to do.

Questions

Sharath George

Why do they assume that all guest OSes run on ring 3? Xen is a contradiction to this.

Wouldnt changing all sensitive instructions to be priviledge lead to massive context swtiching from app to kernel and make the whole system drastically slower?

Mike Wood

1. **Adding the new instruction that can dynamically make previously non-privileged instructions execute as privileged instructions seems like a slick solution, avoiding backward compatibility issues. How much of a performance hit would be paid for such a feature? Was this not mentioned because the solution itself is entirely infeasible?**
2. I **think I recall Andy mentioning something about QEmu doing emulation stu for tricky privileged stu in VMWare - would this then qualify as the hybrid VMM approach?**

Andreii

One thing that annoyed me was a typo on in section 2.5 ("tow" instead of "two"). One thing that surprised me was the institution to which one of the authors belongs (U.S. Air Force).

The questions I have are

1) **what does the U.S. Air Force have to do with virtualization?** and

2) why did Intel make the decisions it made about those instructions discussed in the paper?

Mike Tsai

X86 has been evolving from quite some time now. The architecture itself was not designed to support virtualization from the beginning. However, those problematic instructions do not seem to require a complete redesign of the processor to fix (I am probably wrong, since I am no expert on CPU design).

Qiang Wei

1. What kind of virtual machine could be called as a secure VMM? What are the properties?

2. The paper seems to confuse the concept of VMM and VM somewhere. The paper mentioned that "a VMM provides an execution environment almost identical to the original machine". Should it be VM instead of VMM?

Gang Peng

I don't really understand the requirement 1 mentioned in this paper. Could you explain it a little bit?
In section 4.1, there is a term "real problem state". What does this mean?

Kati

1. **How do you think about the authors' assessment of Intel's suitability? Do you share it, or do you think it is too exaggerated?**
2. **What are the criteria that need to be fulfilled by a virtualizable processor?**

Petter

The paper points out potential security problems in existing VMs—some of which, unfortunately, seem largely specious. It is certainly true that files can be transferred between VMs using services such as FTP, but how is this a security problem rather than a feature? FTP is not inherently unsecure (if executed, within each OS, with the proper priveleges)