The NOVA Microhypervisor

Udo Steinberg

Germany Microprocessor Lab, Intel Labs
INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

Intel may make changes to specifications and product descriptions at any time, without notice.

All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Nehalem, Westmere, Sandy Bridge and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

Intel, Intel Inside, Xeon, and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

Copyright © 2012 Intel Corporation.

Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM) and, for some uses, certain computer system software enabled for it. Functionality, performance or other benefits will vary depending on hardware and software configurations and may require a BIOS update. Software applications may not be compatible with all operating systems. Please check with your application vendor.

*Other names, brands, and logos may be claimed as the property of others.
Virtualization

A virtual machine is defined to be an "efficient, isolated duplicate of a real machine"

Popek & Goldberg (1974)

The x86 instruction set used to contain "virtualization holes"

Possible ways to address these virtualization holes:
- Paravirtualization
- Binary Translation
- Make changes to the architecture

VT-x designed to close x86 virtualization holes by introducing new processor modes of operation
- VMX root mode / VMX non-root mode
Intel® Virtualization Technology (VT-x)

VM Exit
- Guest-to-Hypervisor transition
  - External Events (Interrupts)
  - Sensitive Instructions
- Save guest state in VMCS
- Load host state from VMCS

VM Entry
- Hypervisor-to-Guest transition
- Save host state in VMCS
- Load guest state from VMCS
- Possibility to inject events
Optimization of VT Transition Latencies

Virtual-Machine Control Structure (VMCS)

- Holds guest/host register state in physical memory region
- Accessed via VMREAD/VMWRITE instructions
- Enables processor implementations to cache VMCS data on-die

Significant reductions of VT transition latencies over the years
A monolithic hypervisor is a single point of failure.

*Other names, brands, and logos may be claimed as the property of others.
The trusted computing base is defined as

- "the collection of hardware, software, and setup information on which the security of a system depends"
- "a small amount of software and hardware that security depends on and that we distinguish from a much larger amount that can misbehave without affecting security"

Butler Lampson (1992)

From a security perspective it is desirable to

- Implement fine-grained functional disaggregation
- Enforce the principle of least authority (POLA)
- Minimize the TCB for each application and VM
Example: Xen*

P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield (2003):

„Xen and the Art of Virtualization“

*Other names, brands, and logos may be claimed as the property of others.
Example: KVM


*Other names, brands, and logos may be claimed as the property of others.

*Other names, brands, and logos may be claimed as the property of others.
Security in Virtualized Environments

Virtualization layer is a security-critical part of the system

- Can contain exploitable vulnerabilities
- Replaces physical isolation with logical isolation
- Increases the trusted computing base
- Requires additional configuration and maintenance

Loss of isolation has severe impact

- Subversion of the hypervisor compromises all VMs at once
- Facilitates attacks from below the OS kernel
Vulnerabilities are Real

VMware ESX*

- CVE-2008-2100:
  „Multiple buffer overflows in VIX API“
- CVE-2009-1244:
  „Unspecified vulnerability in the virtual machine display function“

Xen*

- CVE-2007-4993:
  „pygrub allows local users in the guest domain to execute arbitrary commands in domain0“
- CVE-2008-3687:
  „Heap-based buffer overflow in the flask_security_label function“
- CVE-2012-0217:
  „64-bit PV guest privilege escalation vulnerability“

*Other names, brands, and logos may be claimed as the property of others.
Securing the Virtualization Layer

General idea: Make the virtualization layer as small as possible

- Use virtualization features of modern Intel® CPUs to reduce software complexity
  - VT-x, VT-d, Nested Paging
  - No Paravirtualization, No Binary Translation

- Fine-grained functional disaggregation, multiple components
  - Microhypervisor (privileged)
  - User-level virtual-machine monitor per VM (deprivileged)
  - User-level device drivers and applications (deprivileged)

- Principle of least privilege among all components
NOVA OS Virtualization Architecture

Source code available: https://github.com/IntelLabs/NOVA

*Other names, brands, and logos may be claimed as the property of others.
NOVA Microhypervisor

Combines hypervisor and 3rd-gen. microkernel functionality
- Based upon previous microkernel research, mostly L4
- Inspired by features from EROS, Pebble

Goals
- High-performance, low-complexity, secure and scalable hypervisor
  - Use modern hardware virtualization features
- Co-host secure applications and VMs (legacy reuse)
- Fix some shortcomings of original L4 kernels
  - Lack of Communication Control
  - Lack of MP Support
  - Priority Inversion during Synchronous IPC

Research and Development
- 2006-2011 Technische Universität Dresden
- Since 2012 Intel Labs
Capabilities

• Reference to a kernel or hardware object plus auxiliary data
  - Immutable to the user - cannot be inspected, modified or accessed directly
    • User references a capability via an integral number (capability selector)
  - Implementation
    • Align all kernel objects on a cacheline boundary
    • Store access permissions in lower 5 bits of the pointer

• Types
  - Memory Capability
  - I/O Port Capability
  - Object Capability

• Operations
  - Invocation
  - Delegation, Revocation, Translation
Microhypervisor Abstractions

Microhypervisor implements 5 types of objects:

- Protection Domain
- Execution Context
- Scheduling Context
- Portal
- Semaphore

Hypercall interface uses capabilities for all operations.
Communication

- Communication is
  - Synchronous
  - CPU-Local
- Destination is a portal, not a specific thread
- Internal PD structure not revealed to other party
- Reply capability refers to caller, auto-destructed on invocation
During IPC, client donates its timeslice to the server
- Kernel switches from thread C to thread S without changing the current timeslice.
- Bypasses the scheduler during IPC and thereby improves IPC performance

Effect is priority inheritance, but only until S is preempted
- If the kernel fails to recognize the dependency between C and S after the preemption, S will consume its own timeslice s instead of timeslice c.

Result: Priority Inversion

Details: A Real-Time Programmer’s Tour of General-Purpose L4 Microkernels, EURASIP 2008
Priority Inversion

- High-priority thread $H$ blocked by low-priority thread $L$ holding $S$.

- Unbounded priority inversion if $M$ prevents $L$ from running and thus from releasing $S$.

- Solution: Priority Inheritance
  - Server inherits priority of all its clients for the duration of their requests
  - Hypervisor tracks dependencies
  - Servers do not need time of their own
Helping and Donation

Donation:
- Scheduler follows communication link from C to S

Helping:
- H retries its operation; switches to S if rendezvous fails

Both mechanisms are transitive

Udo Steinberg
Intel Labs / GML
VM Exit Handling

- User-level VMM implements complex x86 interface
- Transfer of guest state between VM and VMM via synchronous message passing (IPC)
Interrupt Handling

- Interrupt fan-out to multiple components via semaphores
- Recall of virtual CPUs to inject interrupt vectors
Memory Management

Each protection domain has up to 3 different page tables

- **Host page table (2 or 4 levels)**
  - Defines the layout of applications running directly on top of the hypervisor
  - 3 GB (32bit) or 128 TB (64bit) host-virtual address space
  - Upper portion used by the hypervisor itself

- **Nested page table (4 levels)**
  - Defines the physical memory layout of virtual machines
  - Zero-based contiguous guest-physical memory

- **DMA page table (up to 6 levels)**
  - Defines DMA regions of host-virtual or guest-physical address space
  - DMA access to other regions aborted by IOMMU
Memory Virtualization with Nested Paging

Nested paging increases page walk from 4 levels up to 24 levels

Udo Steinberg

Intel Labs / GML
Memory Virtualization without Nested Paging

On processors without support for nested paging, the hypervisor must
- Walk the guest page table
- Walk the host page table
- Create a shadow page table

MMU configured to use the shadow page table
- Hypervisor must intercept all guest page faults and TLB flushes

Behavior of a virtual TLB
Device Drivers

Device drivers are the most prominent source of bugs

- Move device drivers out of the privileged code base
- Implement them as regular user-level applications

Virtual machines benefit from having direct access to devices

- No need to emulate a virtual device in software
- Higher performance due to fewer VM exits

Virtualization layer must control use of DMA and interrupts.
Intel® Virtualization Technology (VT-d)

DMA Remapping Engine (IOMMU)

Context Cache

Translation Cache

B:D:F Addr Len

Memory Access with Physical Address

Address Translation Structures: Domain A

Address Translation Structures: Domain B

Context Table

Device 1

Device 2

Device n

Udo Steinberg

Intel Labs / GML
Impact of Attacks in NOVA

Attack from Guest OS
• Hypervisor attack surface is message-passing interface
• VM can compromise or crash its associated VMM

Attack from VMM
• Hypervisor attack surface includes hypercall interface
• Access to other components controlled by capabilities

Attack from Device Driver
• DMA and interrupt usage restricted by IOMMU
• Hypervisor resources never exposed to user level
Interrupt-Style Execution Model

- One kernel stack per core
  - No unwinding due to "noretuns" functions
- Continuations
  - Encode the remaining execution path of blocked or preempted execution contexts
  - Resume at the top of the kernel stack
- Kernel entry and exit directly inside the TCB
Preemption Points

- Kernel stack contents lost during context switch
- Thread switch in interrupt context impossible
  - Set hazard field instead
- Check for preemptions in long-running code paths
  - Save continuation state
  - Perform context switch
Performance Implications

• Single-stack design reduces cache and TLB footprint
  - No need to restore caller-saved registers
  - No misaligned return stack after context switch
  - Return to user from anywhere on the kernel stack

• Only long-running operations enable interrupts
  - Short code paths can keep interrupts disabled
  - Some code paths MUST have interrupts enabled to avoid deadlock
  - Deferred context switch at next serializable point

• Hazards
  - Efficient means to encode special conditions that can be checked later, e.g. when returning to user mode
  - Example: Preemption, Recall, FPU active, RCU quiescent state
IPC Performance Comparison

**CPU Clock Cycles**

- **Entry + Exit**
  - L4Ka::Pistachio: 158
  - L4/Fiasco: 158
  - NOVA: 158

- **Kernel**
  - L4Ka::Pistachio: 282
  - L4/Fiasco: 255
  - NOVA: 120

- **TLB Effects**
  - L4Ka::Pistachio: 508
  - L4/Fiasco: 312
  - NOVA: 281

**Notes:**
- -23% other PD
- -33% same PD

**Intel® Core™ i7 2.66 GHz**

Roundtrip Inter-Process Communication between two threads
Overhead of the Virtualization Layer

Kernel-Compile Benchmark:
  CPU: Intel® Core™ i7 2.67 GHz
VM Configuration:
  Single virtual CPU, virtual disk
  512 MB Guest Memory, EPT+VPID

Details: NOVA: A Microhypervisor-Based Secure Virtualization Architecture, Eurosys 2010

*Other names, brands, and logos may be claimed as the property of others.
Performance and TCB Size Comparison

Kernel-Compile Benchmark:
- **CPU**: Intel® Core™ i7 2.67 GHz
- **VM Configuration**: Single virtual CPU, virtual disk
- **512 MB Guest Memory, EPT+VPID**

Details: [NOVA: A Microhypervisor-Based Secure Virtualization Architecture](#), Eurosys 2010

*Other names, brands, and logos may be claimed as the property of others.*
Status

Microhypervisor

• Runs on x86 machines with Intel® VT-x or AMD*-V
• Supports 32-bit and 64-bit, SMP, Nested Paging, IOMMU
• Works with different user-level environments

User-Level Virtual Machine Monitor

• Implements virtual device models: NIC, SATA, VGA, PCI, ...
• Supports direct assignment of host devices to VMs
• No 64-bit support yet

*Other names, brands, and logos may be claimed as the property of others.
Summary

• Disaggregated virtualization layer provides additional isolation boundaries and improves security

• Excellent performance using Intel® hardware virtualization features: VT-x, VT-d, Nested Paging

• NOVA microhypervisor is a research prototype
  - Reduced size of the trusted computing base by an order of magnitude, compared to monolithic hypervisors, while exceeding their performance