General-purpose computing with VirtualBox on Genode/NOVA

Norman Feske

<norman.feske@genode-labs.com>
1. VirtualBox

2. NOVA microhypervisor and Genode

3. Transplantation of VirtualBox to NOVA

4. Demo

5. War stories

6. Project Turmvilla

7. The Book “Genode Foundations”
Outline

1. VirtualBox

2. NOVA microhypervisor and Genode

3. Transplantation of VirtualBox to NOVA

4. Demo

5. War stories

6. Project Turmvilla

7. The Book “Genode Foundations”
Architecture overview

General-purpose computing with VirtualBox on Genode/NOVA
Starting up a VM process

VM process

open

/dev/vboxdrv

kernel

vboxdrv.ko
VM process running

```
root mode
VM process
load VMMR0
/dev/vboxdrv
kernel
vboxdrv.ko
non-root mode
VMMR0 / Hypervisor
```

General-purpose computing with VirtualBox on Genode/NOVA
General-purpose computing with VirtualBox on Genode/NOVA
Flow of a virtualization event

- **root mode**
  - VM process
  - /dev/vboxdrv
  - vboxdrv.ko
  - VMMR0 ?
    - yes
    - no
  - VM RUN returns

- **non-root mode**
  - world switch

**Guest OS**
Risks for desktop virtualization

- root mode
- non-root mode
- Guest OS
- VM
- process
- /dev/vboxdrv
- kernel
- vboxdrv.ko
- VMMR0 / Hypervisor
- highly complex
Risks for desktop virtualization

General-purpose computing with VirtualBox on Genode/NOVA
Risks for desktop virtualization

root mode

VM
process

access control?

/dev/vboxdrv

kernel

vboxdrv.ko

highly complex

VMMR0 / Hypervisor

non-root mode

Guest OS

General-purpose computing with VirtualBox on Genode/NOVA
Risks for desktop virtualization

- Root mode
  - Authorized to change the kernel
  - Highly complex
  - Access control?

- Non-root mode
  - Guest OS
  - VM
  - Process
  - /dev/vboxdrv
  - Kernel
  - vboxdrv.ko
  - Highly complex

VMMR0 / Hypervisor
1. VirtualBox

2. NOVA microhypervisor and Genode

3. Transplantation of VirtualBox to NOVA

4. Demo

5. War stories

6. Project Turmvilla

7. The Book “Genode Foundations”
NOVA architecture

- **Guest OS**
- **VMM**
- **Apps**
- **Resource management**
- **Drivers**
- **NOVA Microhypervisor**
- **9,000 SLOC**

Non-root mode:
- **Kernel**

Root mode:
- **Kernel**

General-purpose computing with VirtualBox on Genode/NOVA
Flow of a virtualization event

User-level VMM

Guest OS

NOVA

VMCS

world switch

copy

General-purpose computing with VirtualBox on Genode/NOVA
Genode OS architecture

→ Application-specific TCB
Genode OS framework

General-purpose computing with VirtualBox on Genode/NOVA
Genode combined with virtualization
Seoul VMM on top of Genode/NOVA

Unmodified Guest OS

Kernel

virtual CPU

virtual RAM

virtual device

VMM

Init

Resource Multiplexer

Device Driver

Core

NOVA Hypervisor

User Mode

Privileged Mode

General-purpose computing with VirtualBox on Genode/NOVA
Idea

Device models and features of VirtualBox

+ Security of the Genode/NOVA architecture
1. VirtualBox
2. NOVA microhypervisor and Genode
3. Transplantation of VirtualBox to NOVA
4. Demo
5. War stories
6. Project Turmvilla
7. The Book “Genode Foundations”
Identify the interesting parts

Entire VirtualBox code base
  > 4 million lines of code (sloccount)

Narrowed to the interesting parts
  > 2 million lines of code

src/VBox/VMM  src/recompiler
src/VBox/Main  src/libs/liblzf-3.4
src/VBox/Runtime  src/libs/liblzf-3.4/cs
src/VBox/Devices  src/libs/libxml2-2.6.31
src/VBox/Storage  src/libs/zlib-1.2.6
src/VBox/GuestHost  include/VBox
src/VBox/Disassembler  include/iprt
src/VBox/HostServices
Porting the VirtualBox Runtime to Genode

- Facilitate Genode’s existing infrastructure
  - 3rd-party software management tools
  - FreeBSD libc
  - Standard C++ library
  - POSIX threads

Most parts of the POSIX runtime could be reused.
Facilitate Genode’s existing infrastructure

- 3rd-party software management tools
- FreeBSD libc
- Standard C++ library
- POSIX threads

→ Most parts of the POSIX runtime could be reused
Enable subsystems one by one
Enable subsystems one by one

- Guest memory (accessed by recompiler and device models)
  \[\text{RAM, MMIO}\]
Enable subsystems one by one

- Guest memory (accessed by recompiler and device models)
  RAM, MMIO
- I/O-port handling
Enable subsystems one by one

- Guest memory (accessed by recompiler and device models)
  - RAM, MMIO
- I/O-port handling
- PGM, HWACCM, TM
Enable subsystems one by one

- Guest memory (accessed by recompiler and device models)
  *RAM, MMIO*
- I/O-port handling
- PGM, HWACCM, TM
- Device models, PDM, BIOS
Enable subsystems one by one

- Guest memory (accessed by recompiler and device models)
  *RAM, MMIO*
- I/O-port handling
- PGM, HWACCM, TM
- Device models, PDM, BIOS
- Host drivers
  - Using the “Basic front end”
  - Reimplement SDLConsole interface
A look inside a VM process

- Execution Manager
- Recompiler
- Instruction Emulator
- Hardware Acceleration
- VM Enter
- VM Exit
Start with executing the recompiler only

- Recompiler
- Execution Manager
- Hardware Acceleration
- Instruction Emulator
Simple test scenario

General-purpose computing with VirtualBox on Genode/NOVA
Increasing guest complexity

1. Custom-made Genode OS scenarios
Increasing guest complexity

1. Custom-made Genode OS scenarios

2. Small Linux-based images (Tinycore, GRML)
Increasing guest complexity

1. Custom-made Genode OS scenarios

2. Small Linux-based images (Tinycore, GRML)

3. Windows XP
General-purpose computing with VirtualBox on Genode/NOVA
Move scenario to NOVA
General-purpose computing with VirtualBox on Genode/NOVA

Move scenario to NOVA

- PS/2 driver (Input)
- VESA driver (Framebuffer)
- VirtualBox
- Rump FS (File system)
- AHCI driver (Block)

VDI image

Init

Core

kernel

NOVA
Entering non-root mode

- Recompiler
- Execution Manager
- Hardware Acceleration
- Instruction Emulator
- VM Enter
- VM Exit
- IRQs
Entering non-root mode

- VBox VM state $\leftrightarrow$ NOVA UT CB state
Entering non-root mode

- VBox VM state ↔ NOVA UTCB state

- Virtualization of guest memory
  \((EPT\ faults)\)
Entering non-root mode

- VBox VM state ↔ NOVA UT CB state
- Virtualization of guest memory
  \((EPT\ faults)\)
- Enter VT-x conservatively
  \((if\ protected\ mode\ and\ paging\ enabled)\)
Entering non-root mode

- VBox VM state ↔ NOVA UTCB state
- Virtualization of guest memory (EPT faults)
- Enter VT-x conservatively (if protected mode and paging enabled)
- Inject IRQs into recompiler
Entering non-root mode

- VBox VM state $\leftrightarrow$ NOVA UTCB state
- Virtualization of guest memory
  \((EPT\ faults)\)
- Enter VT-x conservatively
  \((if\ protected\ mode\ and\ paging\ enabled)\)
- Inject IRQs into recompiler
- Later: IRQ injection via NOVA into VT-X
Adding features

**Additional drivers**
- Networking

**Guest tools**
- Shared folders
- Host clock
- Mouse-pointer synchronization
Update to VirtualBox 4.3

- Basic front end no longer supported

- Use of main front end code to NOVA port
  - Custom console implementation
  - Shortcut XPCOM middleware
  - Support for using .vbox files
1. VirtualBox

2. NOVA microhypervisor and Genode

3. Transplantation of VirtualBox to NOVA

4. Demo

5. War stories

6. Project Turmvilla

7. The Book “Genode Foundations”
Windows 7 running in VirtualBox directly on top of NOVA
Adaptation of VirtualBox to Genode/NOVA

Ported code
- 400,000 lines of code (sloccount)

New code
- 6,200 lines (sloccount)
  \[hm, iommio, ioint, mm, pdm, pgm, sup\]

Modifications of the original code
- 510 lines added
- 120 lines removed
Current state and outlook

- Usable performance, optimization ongoing
Current state and outlook

- Usable performance, optimization ongoing
- Focused on VT-X, SVM not regularly tested
Current state and outlook

- Usable performance, optimization ongoing
- Focused on VT-X, SVM not regularly tested
- Reduces TCB complexity to two orders of magnitude
Current state and outlook

- Usable performance, optimization ongoing

- Focused on VT-X, SVM not regularly tested

- Reduces TCB complexity to two orders of magnitude

- Useful for building appliances in high-security computing
Current state and outlook

- Usable performance, optimization ongoing
- Focused on VT-X, SVM not regularly tested
- Reduces TCB complexity to two orders of magnitude
- Useful for building appliances in high-security computing
- Stepping stone for using Genode as a general-purpose OS
1. VirtualBox

2. NOVA microhypervisor and Genode

3. Transplantation of VirtualBox to NOVA

4. Demo

5. War stories

6. Project Turmvilla

7. The Book “Genode Foundations”
War stories
War stories

- Invalid guest state
War stories

- Invalid guest state
- TLB consistency
War stories

- Invalid guest state
- TLB consistency
- Interrupt handling
War stories

- Invalid guest state
- TLB consistency
- Interrupt handling
- Large files in shared folders
Outline

1. VirtualBox
2. NOVA microhypervisor and Genode
3. Transplantation of VirtualBox to NOVA
4. Demo
5. War stories
6. Project Turmvilla
7. The Book “Genode Foundations”
Project Turmvilla

- Use of Genode as our day-to-day OS
Project Turmvilla

- Use of Genode as our day-to-day OS
- VirtualBox as migration path
Project Turmvilla

- Use of Genode as our day-to-day OS
- VirtualBox as migration path
- Reference platform: Lenovo Thinkpad x201
Turmvilla functional requirements

- Wireless networking
Turmvilla functional requirements

- Wireless networking
- Storage (SATA drivers + file system)
Turmvilla functional requirements

- Wireless networking
- Storage (SATA drivers + file system)
- Graphics (driver + GUI stack)

General-purpose computing with VirtualBox on Genode/NOVA
Turmvilla functional requirements

- Wireless networking
- Storage (SATA drivers + file system)
- Graphics (driver + GUI stack)
- User input (PS/2 and USB HID)
Turmvilla functional requirements

- Wireless networking
- Storage (SATA drivers + file system)
- Graphics (driver + GUI stack)
- User input (PS/2 and USB HID)
- Integration of guest OS and Genode
Turmvilla functional requirements

- Wireless networking
- Storage (SATA drivers + file system)
- Graphics (driver + GUI stack)
- User input (PS/2 and USB HID)
- Integration of guest OS and Genode
- A fallback!
Turmvilla dual-boot setup

Partitions of physical disk

- genode
  - ext2
  - /boot
  - /genode
  - linux.vdi

- home (LUKS encrypted)
  - ext4

- boot / rescue
  - ext4 4GiB
  - GRUB
  - GNU/Linux
  - X11
  - VirtualBox
  - mount home
  - mount genode

Virtual-machine disks

- Rich GNU/Linux
  - ext3

- home (LUKS encrypted)
  - ext4
Turmvilla Genode scenario

VirtualBox

Noux

Init

Nitpicker

GUI

Window Manager

CLI monitor

timer
acpi_drv
acpi_report_rom
platform_drv
ahci_drv
part_blk
log_file_terminal
log
rump_fs
wifi_drv
ps2_drv
usb_drv
fb_drv
rtc_drv
trace_subject_reporter
input merger
report rom
nitpicker
wm_report_rom
wm
layouter
decorator
vbox_pointer
shared_fs
config_fs
config_rom
rom
cli_nit_fb
cli_terminal
Current state:

- My primary OS since the beginning of June
- Team at Genode Labs starts migration
Turmvilla state and current focus

Current state:
- My primary OS since the beginning of June
- Team at Genode Labs starts migration

Work in progress:
- Tiled and tabbed window manager
- Intel graphics driver
- NOVA kernel-resource management
- Capability-based desktop environment
Outline

1. VirtualBox
2. NOVA microhypervisor and Genode
3. Transplantation of VirtualBox to NOVA
4. Demo
5. War stories
6. Project Turmvilla
7. The Book “Genode Foundations”
The Book “Genode Foundations”

Thank you

Genode OS Framework
http://genode.org

Genode Labs GmbH
http://www.genode-labs.com

Source code at GitHub
http://github.com/genodelabs/genode