A Virtualized Separation Kernel for Mixed Criticality Systems

Ye Li, Richard West and Eric Missimer

Boston University

March 2nd, 2014

Motivation

- Mixed criticality systems requires component isolation for safety and security
  - Integrated Modular Avionics (IMA), Automobiles
- Multi-/many-core processors are increasingly popular in embedded systems
- Multi-core processors can be used to consolidate services of different criticality onto a single platform

Motivation

- Many processors now feature hardware virtualization
  - ARM Cortex A15, Intel VT-x, AMD-V
- Hardware virtualization provides opportunity to efficiently partition resources amongst guest VMs

H/W Virtualization + Resource Partitioning = Platform for Mixed Criticality Systems

Related Work

Existing virtualized solutions for resource partitioning

- Wind River Hypervisor, XtratuM, PikeOS
- Xen, PDOM, LPAR

Traditional Virtual Machine approaches too expensive

- Require traps to VMM (a.k.a. hypervisor) to multiplex and manage machine resources for multiple guests
- e.g., 1500 clock cycles VM-Enter/Exit on Xeon E5506

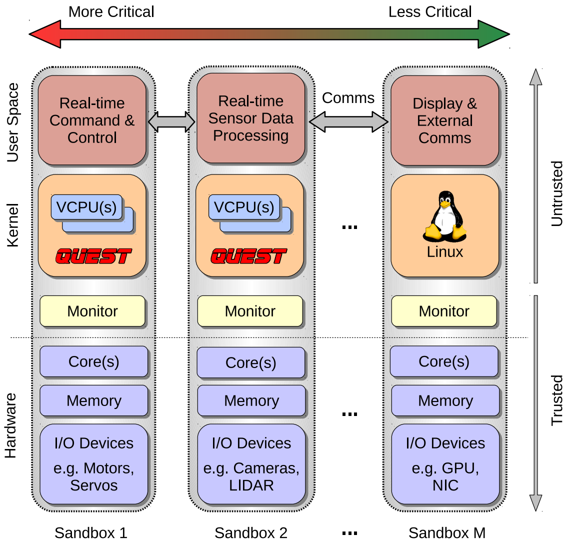
We want to eliminates hypervisor intervention during normal virtual machine operations

Contribution

Quest-V Separation Kernel

- Uses H/W virtualization to partition resources amongst services of different criticalities
- Each partition, or **sandbox**, manages its own CPU, memory, and I/O resources without hypervisor intervention
- Hypervisor only needed for bootstrapping system + managing communication channels between sandboxes

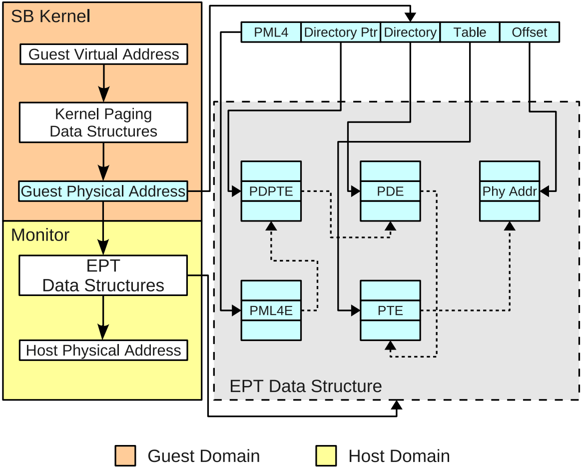
Overview


More Critical
Less Critical
User Space
Kernel
Hardware
Real-time Command & Control
Real-time Sensor Data Processing
Comms
Display & External Comms
VCPU(s)
QUEST
VCPU(s)
QUEST
Linux
...
Monitor
Monitor
Monitor
Core(s)
Core(s)
Core(s)
Memory
Memory
Memory
...
I/O Devices e.g. Motors, Servos
I/O Devices e.g. Cameras, LIDAR
I/O Devices e.g. GPU, NIC
Untrusted
Trusted
Sandbox 1
Sandbox 2
...
Sandbox M

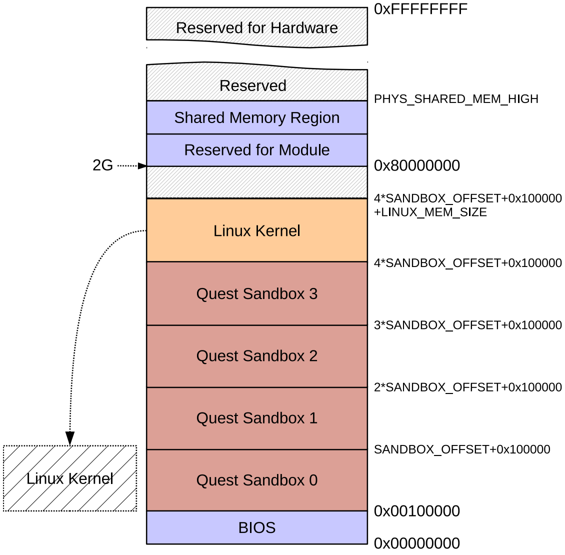
Memory Partitioning

- Guest kernel page tables for GVA-to-GPA translation
- EPTs (a.k.a. shadow page tables) for GPA-to-HPA translation
  - EPTs modifiable only by monitors
  - Intel VT-x: 1GB address spaces require 12KB EPTs with 2MB superpaging

Memory Partitioning



Quest-V Linux Memory Layout



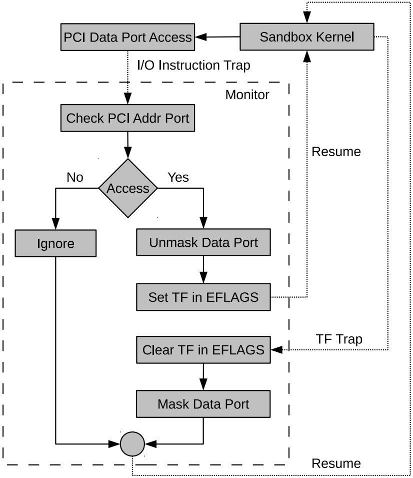
I/O Partitioning

- I/O devices statically partitioned
- Device interrupts directed to each sandbox
  - Eliminates monitor from control path
  - I/O APIC redirection tables protected by EPT
- EPTs prevent illegal access to memory mapped I/O registers
- Port-addressed I/O registers protected by bitmap in VMCS
- Monitor maintains PCI device "blacklist" for each sandbox
  - (Bus No., Device No., Function No.) of restricted PCI devices

I/O Partitioning

PCI devices in blacklist hidden from guest during enumeration

- Data Port: $0xCFC$ Address Port: $0xCF8$



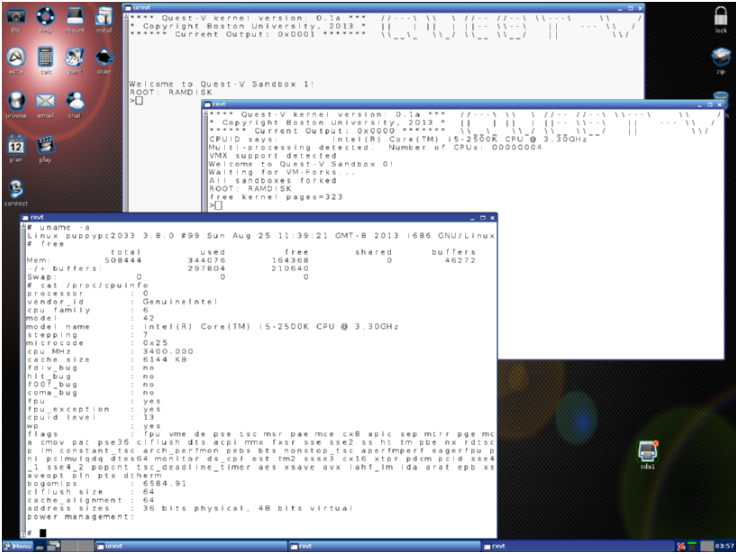
CPU Partitioning

- Scheduling local to each sandbox
  - Avoids monitor intervention
  - Partitioned rather than global
- Native Quest kernel uses VCPU real-time scheduling framework (RTAS '11)

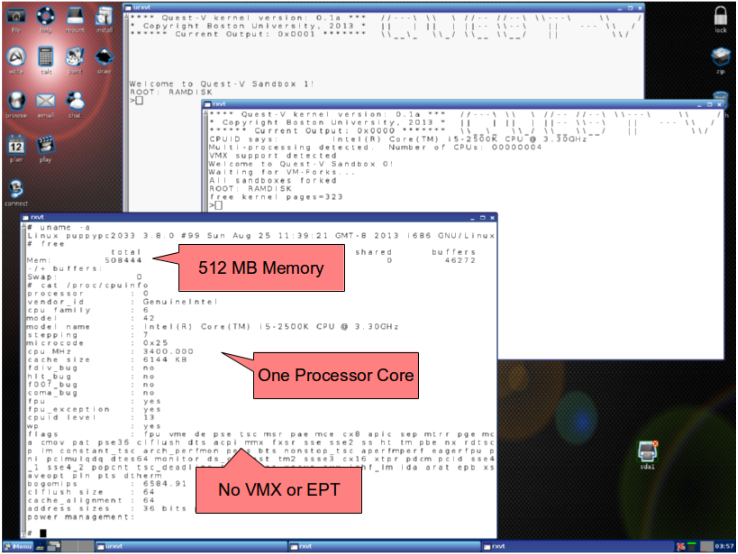
Linux Front End

- Most likely serving low criticality legacy services
- Based on Puppy Linux 3.8.0
- Runs entirely out of RAM including root filesystem
- Low-cost paravirtualization
  - Less than 100 lines
  - Restrict observable memory
  - Adjust DMA offsets
- Grant access to VGA framebuffer + GPU
- Quest native SBs tunnel terminal I/O to Linux via shared memory using special drivers

Quest-V Linux Screenshot



Quest-V Linux Screenshot



Monitor Intervention

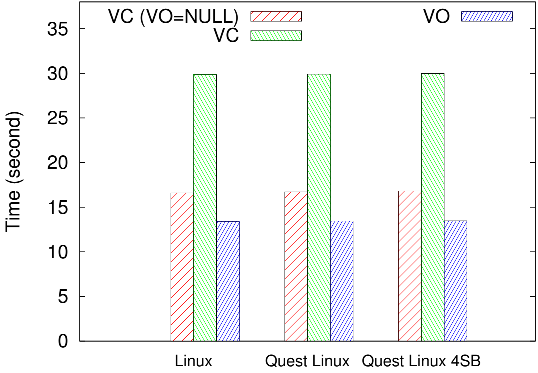
During normal operation, we observed only one monitor trap every 3 to 5 minutes caused by cpuid.

	No I/O Partitioning	I/O Partitioning (Block COM and NIC)
Exception	0	9785
CPUID	502	497
VMCALL	2	2
I/O Inst	0	11412
EPT Violation	0	388
XSETBV	1	1

Table : Monitor Trap Count During Linux Sandbox Initialization

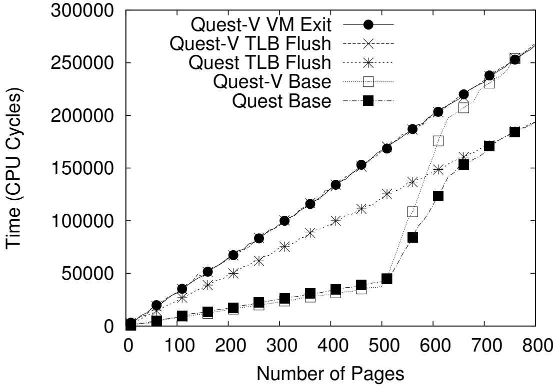
Quest-V Performance Overhead

- Measured time to play back 1080P MPEG2 video from the x264 HD video benchmark
- Intel Core i5-2500K HD3000 Graphics



Memory Virtualization Cost

- Example Data TLB overheads
- Intel Core i5-2500K 4-core, shared 2nd-level TLB (4KB pages, 512 entries)



Conclusions

- Quest-V separation kernel built from scratch
  - Distributed system on a chip
  - Uses (optional) hardware virtualization to partition resources into sandboxes
  - Protected communication channels between sandboxes
- Sandboxes can have different criticalities
  - Native Quest sandbox for critical services
  - Linux front-end for less critical legacy services
- Sandboxes responsible for local resource management
  - Avoids monitor involvement

Ongoing and Future Work

- Online fault detection and recovery
- Technologies for secure monitors
  - e.g., Intel TXT, Intel VT-d
- Micro-architectural Resource Partitioning
  - e.g., shared caches, memory bus

Thank You!

For more details, preliminary results, Quest-V source code and forum discussions. Please visit:

www.questos.org