Partitioned Reliable Operating System Environment

Eric Van Hensbergen (bergevan@us.ibm.com)
Agenda

- Background
  - Motivation
  - Virtualization Overview
- PROSE Approach
- Preliminary Performance Analysis
- Noise/OS Interference Analysis
- Status Update
- Future Work
Background

- **Motivation**: Push the mainstream heavy-weight operating systems out of the way.

- **Why**:  
  - Finer grain of control over system services: scheduling, memory allocation, interrupt handling (or lack thereof)  
  - Reliability: application-specific kernels are likely to be smaller and may even be verifiable using formal methods  
  - Hardware support: Enable use of hardware-specific features which may not be well-matched to generalized mainstream operating system.
Virtualization

Kernel <-> Hypervisor Interface

Hypervisor

Hardware <-> Hypervisor Interface

Hardware Platform
PROSE Approach

- Run applications in stand-alone partition
- Enable execution environment which makes starting a partition as easy as starting an application
- Development environment allowing creation of specialized kernels as easy as developing an application (library-OS)
- Resource sharing between library-OS partitions and traditional partitions keeping library-OS kernels simple and reliable
- Extensions to allow bridging resource sharing and management across the entire cluster.
- Unified communication protocol for resource sharing and control with built-in failure detection and recovery.
rHype: IBM's Research Hypervisor for Power

- Small (~30k lines of code for both x86 & PowerPC)
- Developed as a validation test for Cell virtualization features and as a research platform for LPAR research
- Uses same system interfaces as IBM's commercial Power virtualization engine
- Open Sourced: http://www.research.ibm.com/hypervisor
Transparent Application Development Process

Original Application

PROSE Application

Custom OS Library

1. abc
2. MPI
3. App

lib
OS

App

my
OS

App
Library OS Components

application(s)

virtual machine

library interfaces

Thread Library

network
corso
time

sys svc gw

standard I/O

9P Filesystem

Scheduler

Channel I/O

library OS services (kernel libc)
Resources Sharing via File Name Space

- **Hardware Devices**
  - Disk
    - /dev/hda1
    - /dev/hda2
  - Network
    - /dev/eth0
    - /dev/tap0
    - /dev/tap1
    - tun/tap driver

- **System Services**
  - TCP/IP Stack
    - /net
    - /arp
    - /udp
    - /tcp
    - /clone
    - /stats
    - /0
    - /1

- **Application Services**
  - Database
    - /sql
    - /clone
    - /0
    - /1
    - /query
    - /result
  - GUI
    - /ctl
    - /data
    - /listen
    - /local
    - /remote
    - /status
    - /win
    - /clone
    - /2

- **File System**
  - /mnt/9p_root
  - /mnt/common_fs
  - /mnt/remote_nfs
PROSE I/O
Performance Experimental Setup

Both

- IBM JS20 Blade
- SLOF Firmware
- 4 GB DRAM Memory
- Single* 1.66 GHZ 970

arlx112

- Linux 2.6.10
- Running GUPS w/128MB set size

arlx113

Controller Partition
- Linux 2.6.10
- 64 MB of memory

PROSE Partition
- GUPS + lib-os
- 1 GB of memory
- GUPS w/128MB set size
- Console & Time over 9P
Sparse Memory Benchmark Performance
Noise Control w/PROSE & Hypervisors

- Allow strict control of percentage of CPU devoted to application versus system daemons and I/O requests
- Can eliminate jitter associated with interrupt service routines
- Provides a higher degree of determinism than vanilla Linux, but does so at a performance cost
Noise Analysis Experimental Setup

Both

- IBM JS20 Blade
- SLOF Firmware
- 4 GB DRAM Memory
- Single * 1.66 GHZ 970

arlx112

- Linux 2.6.10

```c
for(i=0; i<num_samples; i++) {
    start = mftb();
    for(w=0; w<work_len; w++);
    stop = mtftb();
    delta[i] = stop-start;
}
```

arlx113

Controller Partition
- Linux 2.6.10
- 64 MB of memory

PROSE Partition
- Application + lib-os
- 1 GB of memory
- Console & Time over 9P
Noise Comparison

Linux Idle

![Graph of Linux Idle](image)

Linux Loaded

![Graph of Linux Loaded](image)

PROSE Idle

![Graph of PROSE Idle](image)

PROSE Loaded

![Graph of PROSE Loaded](image)
rHype scheduler explanation

- Simple fixed-slot round-robin scheduler.
- Quanta is determined by special HDEC counter (default quanta=20ms)
- Partitions can be given greater share of CPU by being assigned multiple slots.
Potential Interrupt Policies

- **Hypervisor Serviced Interrupts**
  - ISR runs in hypervisor context

- **Partition Preempting Interrupts**
  - Partition with ISR preempts current partition

- **Hypervisor Mitigated Interrupts**
  - Hypervisor queues interrupt for delivery to partition

- **Hardware Based Interrupt Routing**
Phase Scheduling Noise

- FWQ aren't aligned to scheduler quanta
- Noise is exacerbated by fixed length scheduling slots.
- Fixed noise ratio based on HDEC length
Status

- Implementing a PAPR compliant CR/Q transport for 9P which could be used by IBM's commercial hypervisor.

- Thread module has been implemented and will be available as part of PROSE libraries.

- Prototype Xen 9P transport was implemented with reliability/fail-over capabilities. Needs to be moved to new code-base.

- Working to support a JVM running on top of PROSE in order to be able to run a large scale commercial workload for performance analysis.
Future Work

- **Performance Experiments**
  - Continue on track to being able to run a large commercial workload instead of microbenchmarks.

- **Noise Experiments**
  - Experiment with dynamic scheduling policy which adapts slot-scheduler based on idle yielding.
  - Repeat experiments with different interrupt service policies.
  - Repeat experiments with different virtualization implementations (Xen, VMware, IBM Virtualization Engine, etc.)
  - Repeat experiments with a standard benchmark w/ I/O dependencies instead of relying on microbenchmarks.
  - SMP studies.
Acknowledgments

This work would not be possible without the contributions of Jimi Xenidis, Michal Ostrowski, Orran Krieger, and the rest of the rHype team. This work was supported in part by the Defense Advanced Research Projects Agency under contract no. NBCH30390004.

http://www.research.ibm.com/prose
http://www.research.ibm.com/hypervisor
http://www.research.ibm.com/systemsim
BACKUP SLIDES
Results - Linux

Idle

Loaded
Results - PROSE

Idle

Loaded
HDEC Sensitivity