“Bare Metal”—Speeding Up Java™ Technology in a Virtualized Environment

Joakim Dahlstedt

CTO, Java Runtime Products Group
BEA Systems
http://www.bea.com

TS-3792

Project Bare Metal

Server virtualization and Java™ technology

Learn about a technology that virtualizes your Java technology-based application **transparently** without losing performance.

Agenda

Hypervisor optimized server Java technology

Project Bare Metal Overview

Looking Under the Hood

Performance Analysis

Virtualization Layers and Isolation

Going Forward

Summary

Weird Magic?

No, this is old technology applied in a new environment

"Strange are the ways of men,
And strange the ways of God!
We tread the mazy paths
That all our fathers trod."

Robert Louis Stevenson

Agenda

Hypervisor optimized server Java technology

Project Bare Metal Overview

Looking Under the Hood

Performance Analysis

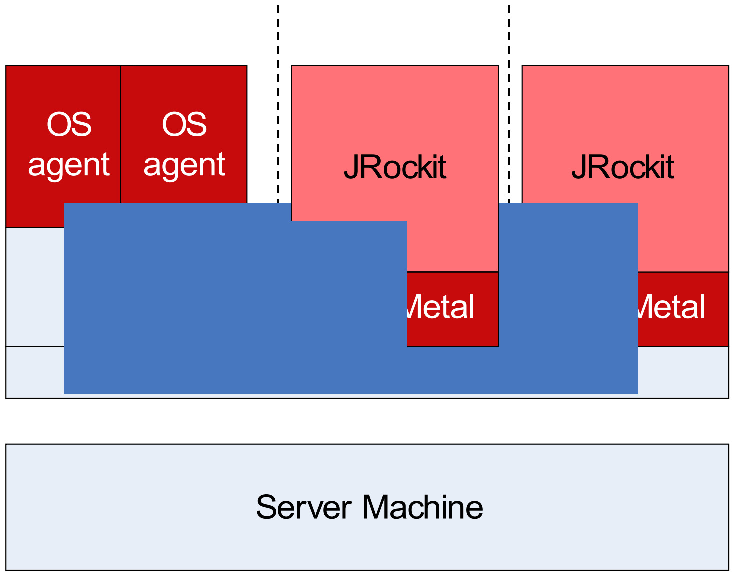
Virtualization Layers and Isolation

Going Forward

Summary

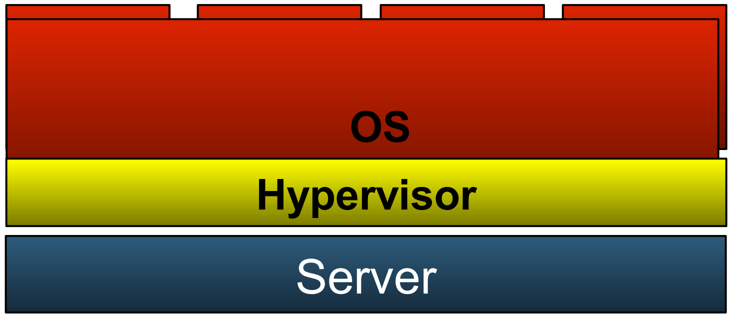
Overview—Bare Metal Architecture

- Start from normal OS
- Run Java code on separate virtual machine
- No OS in the way for the Java VM
- High performance
- Good resource control
- Efficient virtual device drivers
- OS file-system through agent
- 3rd party JNI using agent



Hypervisors ⇨ Server Virtualization

- Software partitioning
  - Divide a machine into multiple virtual machines
  - One server becomes many
- Like an OS micro-kernel—very few functions
  - Resource isolation/partitioning
  - Scheduling of virtual machines



Server Virtualization: Cost Reductions

- Server Consolidation
  - Put multiple operating systems on the same server
  - Put multiple isolated applications on the same server

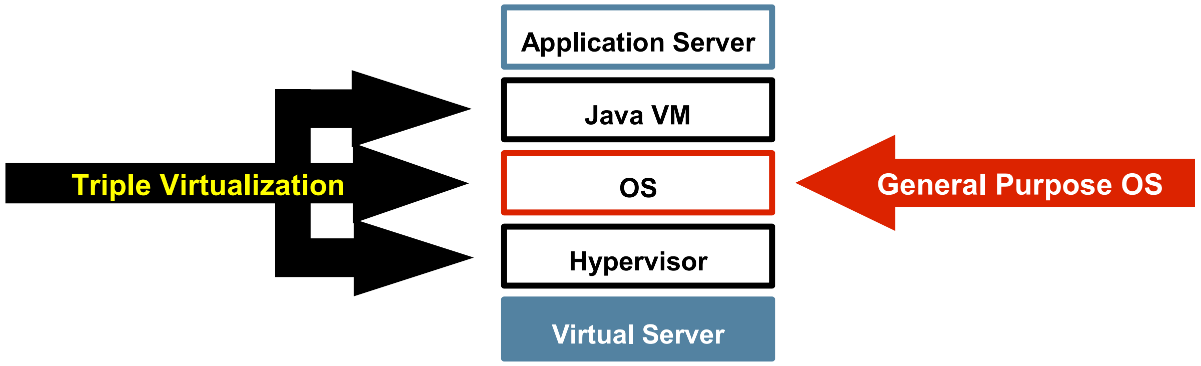
- Simplified IT Management
  - Add new virtual servers without waiting for new hardware
  - Save snapshots of running applications on file
  - Move running applications off servers

Higher utilization and easier management!

Server Virtualization: New Functions

- Resource control and isolation
  - Guarantee a minimum amount of a hardware resource (CPU, memory, networking) an application gets
- Suspend/resume
  - Temporarily freeze an application and then thaw it as if nothing happened; like suspend resume on a laptop
- Store to image/restore from image
  - Store a running application to disc; later restart from that image as if nothing happened like laptop hibernate
- Live migration
  - Move a running image from one box to another with minimal (sub-second) downtime

Server Virtualization: Room for Performance Improvements



- Triple virtualization
  - Triple virtualization by hypervisor, OS, and Java VM
  - Virtualization layers uncoordinated: GC, swapping, thread scheduling, etc
  - Redundant activites in each layer
- Large and slow general purpose OS
  - Increases footprint
  - Increases maintenance
  - Decreases performance

Java Platform Optimized for a Hypervisor

Removing the OS-Java VM conflicts

- An idea
  - Remove the OS
  - Make hypervisor and Java VM aware of each other
- Teamwork (Java technology and hypervisor) to optimize:
  - Raw speed/pausetimes
  - High-availability functionality (suspend/resume/migrate)
  - Reduced memory footprint

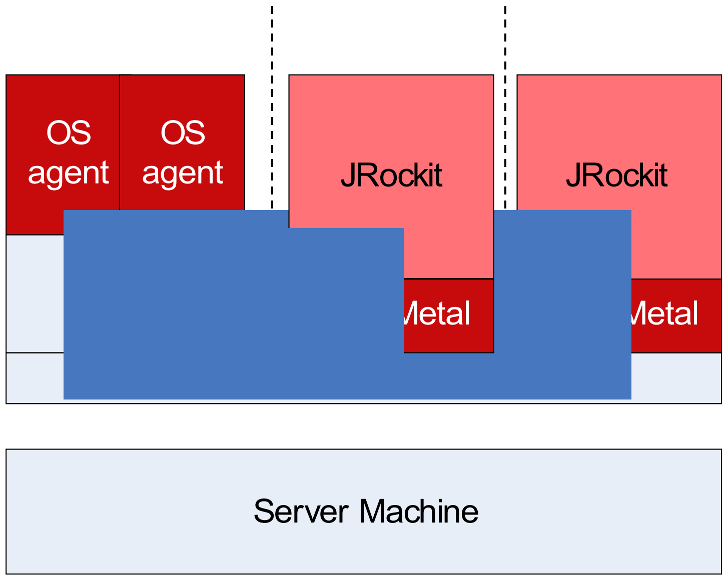
Agenda

Hypervisor optimized server Java technology

- Project Bare Metal Overview
- **Looking Under the Hood**
- Performance Analysis
- Virtualization Layers and Isolation
- Going Forward
- Summary

Overview—Bare Metal Architecture

- Start from normal OS
- Java based operations and native operations on separate machines
- No OS in the way for the Java VM
- High performance
- Good resource control
- Efficient virtual device drivers
- OS file-system through agent
- 3rd party Java native interface using agent



How You Can Use It

- Very much like normal Java code

```
> java_vmware HelloWorld
> java_xen HelloWorld

or

> java_vmware -i 192.168.0.100 HelloWorld
```

- A new ”OS” instead of a new process is started
- Normal `top`/Task Manager will only show CPU utilization of agent

Threads and Context Switching

- Very light-weight threads
- Context-switching about as expensive as a method call
- Thread-contention—directed yields and smart spinning to avoid unnecessary waste of CPU cycles
- Initial implementation—no SMP

TCP/IP Implementation

- TCP/IP stack inside the Bare Metal container
- Optimized for Java technology
- Focus on the Java based protocols (TCP/UDP/...)
- Network data is not passed through to the agent
- Network data sent directly from Java platform through the hypervisor out on the network

Local and Remote File System

- Posix file system
- Mounting a virtual file system
  - Some directories go to Java VM-local disk
  - Other directories go to agent and end up on the normal OS
- Allows fast access to Java VM-local disk
- Allows backup tools etc to work as normally on the operating system
- Initial implementation—slow

Posix-like Environment for the Java VM

- Core libc functionality implemented
  - Malloc, free, str*, printf, open, close etc.
- Pthread implementation
- Version 1.0 of Bare Metal ran an unmodified version of JRockit for Linux
- Coming versions optimized for Bare Metal

Execution of Java Native Interface Code

- The Java native code is executed within the Bare Metal container
- 3rd party Java native interface code
  - Calls are detected
  - Sent as a request to the OS-process
  - The OS-process unmarshals and executes

Hypervisors Made Bare Metal Feasible

- The barrier to entry for a new OS was too high
  - Bare Metal can coexist with the OS
  - OS filesystems, scripts and backups continue to work
  - Bare Metal launched as a normal process
- Supporting various device drivers was too expensive
  - On a hypervisor there is only one device of each kind
- Hypervisors change the rules of the game
  - Bare Metal gives JVM™ software need a mechanism to adapt

Bare Metal Is Not a Good Fit When

- The application uses 3rd party native code excessively
  - It will always be slow for BM
  - The 3rd party native code is compiled for a specific OS—Not Bare Metal
- The application uses the OS filesystem excessively
  - Sending file operation request through to the agent results in increased overhead
  - Future versions of Bare Metal will target this
- The application needs a graphical display
  - Bare Metal is a server environment: no screen, no GUI, no sound

Agenda

Hypervisor optimized server Java technology

Project Bare Metal Overview

Looking Under the Hood

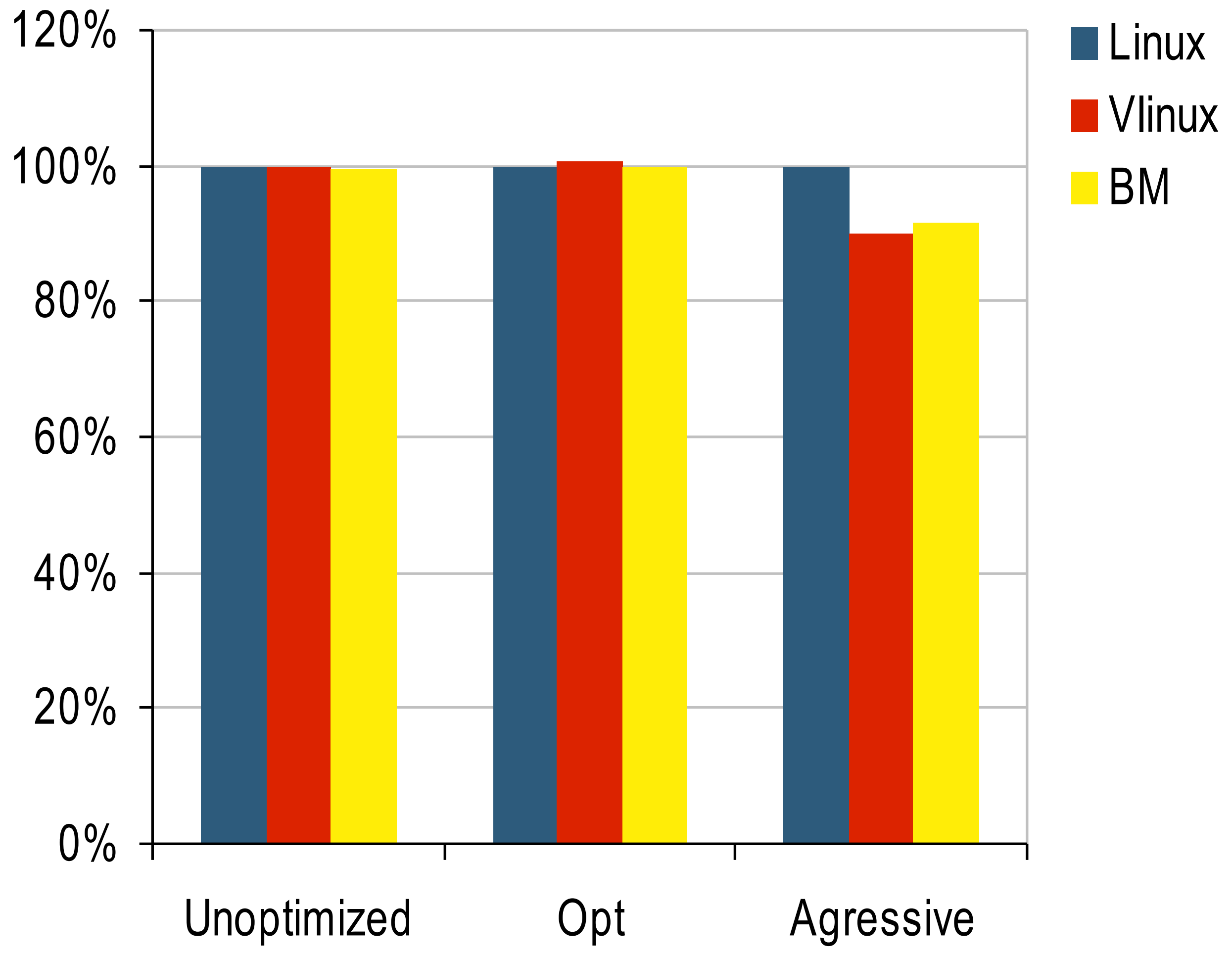
Performance Analysis

Virtualization Layers and Isolation

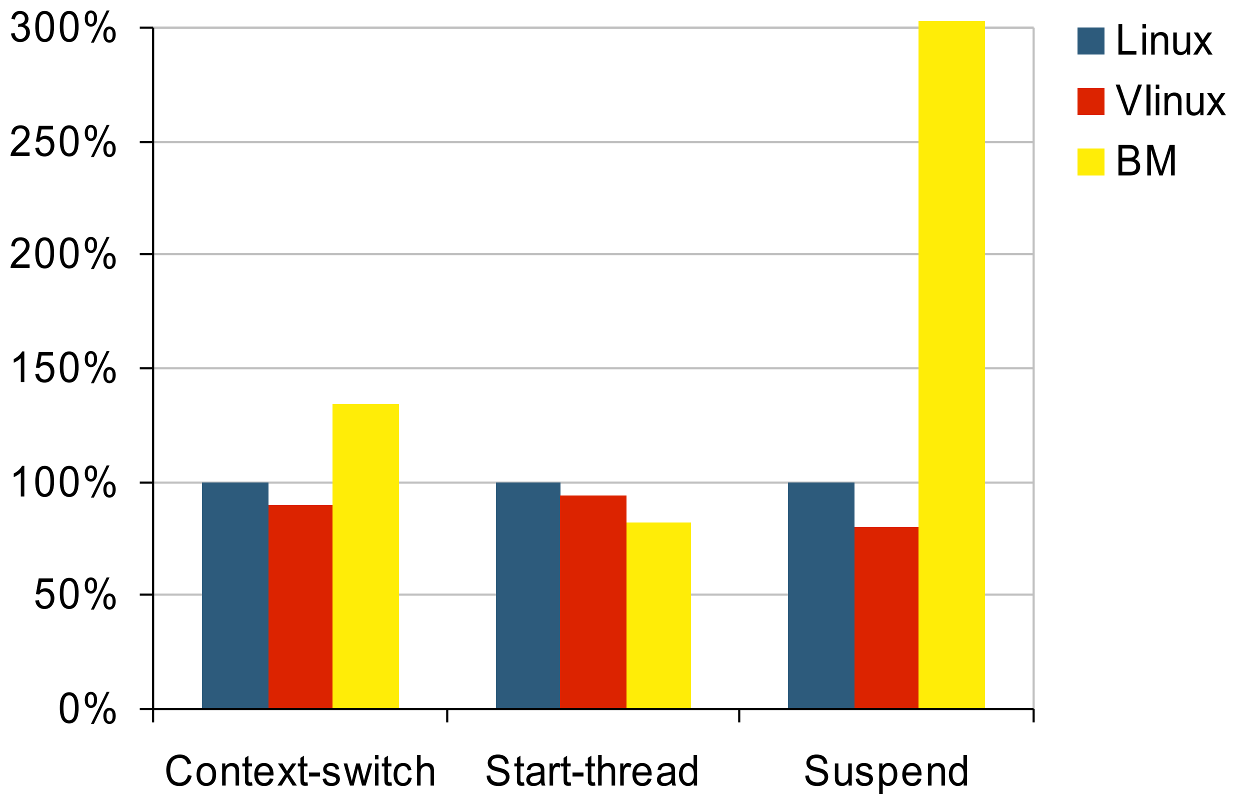
Going Forward

Summary

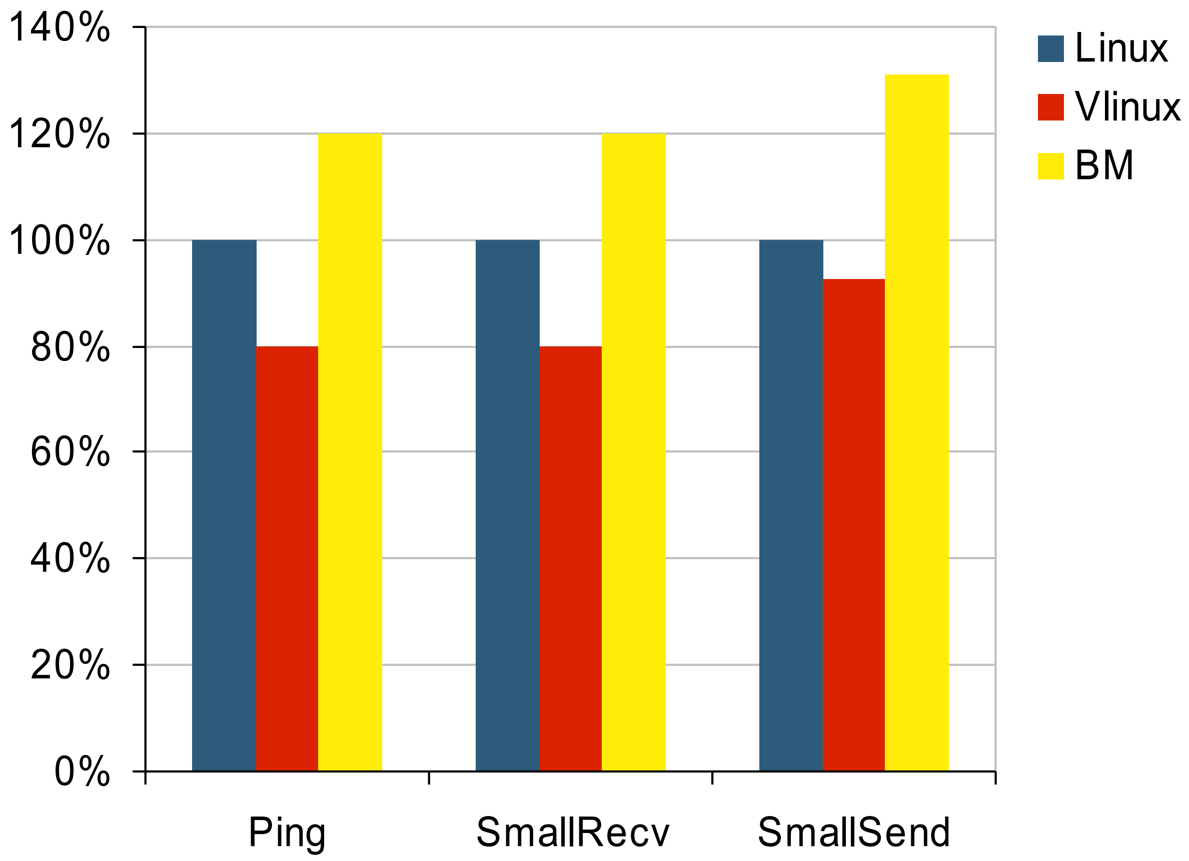
SPECjbb2005



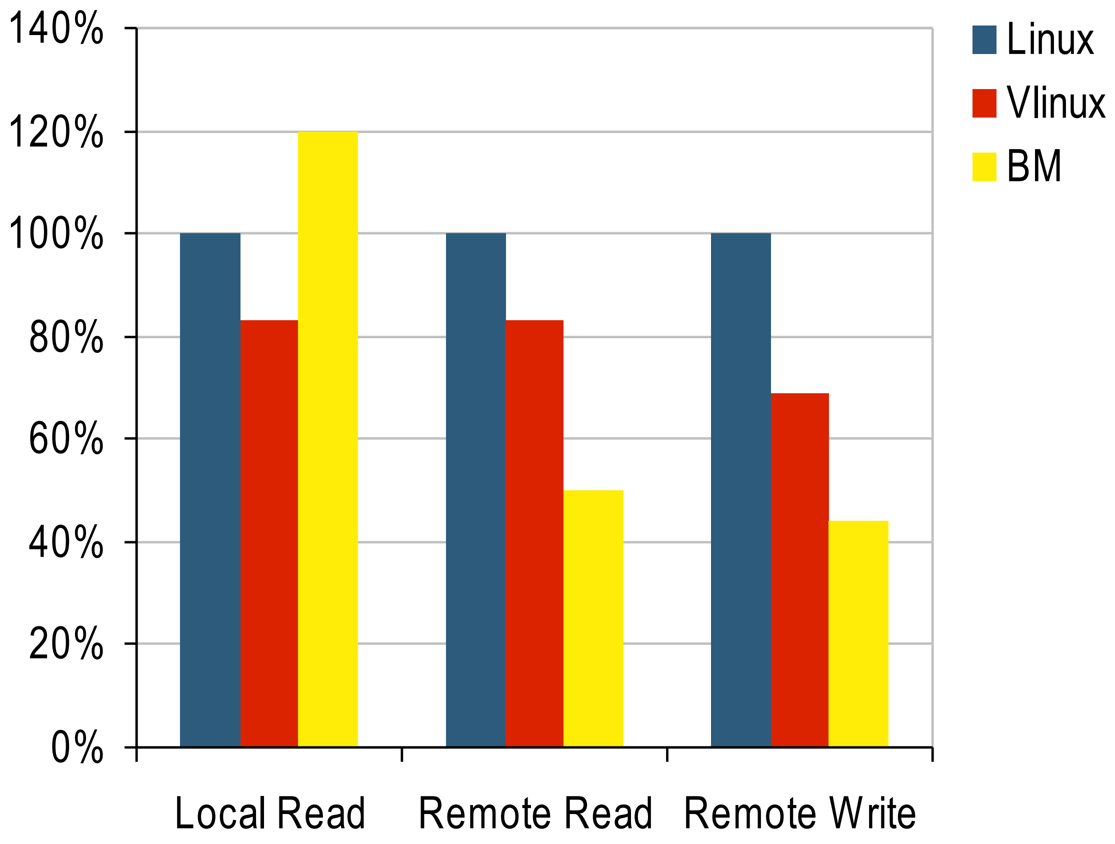
Thread Performance



Networking Performance



File Performance



Agenda

Hypervisor optimized server Java technology

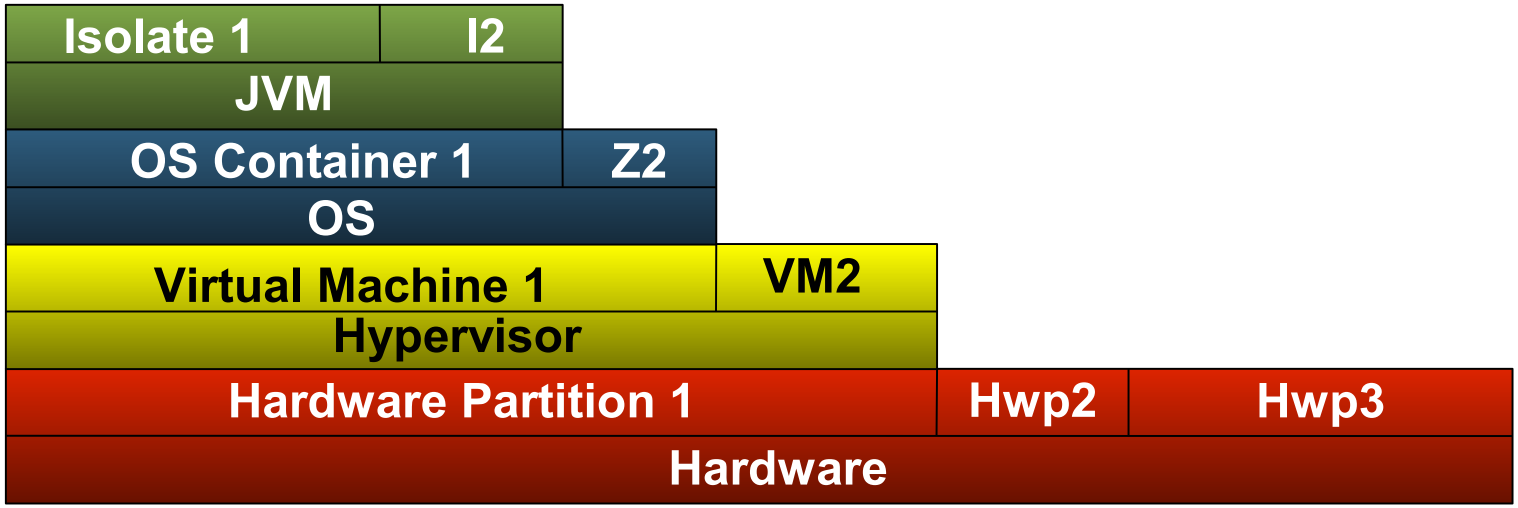
- Project Bare Metal Overview
- Looking Under the Hood
- Performance Analysis
- **Virtualization Layers and Isolation**
- Going Forward
- Summary

JSR 121 Application Isolation API Specification

- Problem: how to enable efficient execution of multiple Java VMs on the same box
  - Resource requirements—memory footprint
  - Efficiency—startup time and execution
- Multi-tasking VM (MVM)
  - Reference implementation for JSR 121
  - Developed by SunLabs—modifications to HotSpot
  - Java VM-level virtualization
- Disadvantages
  - Hard to control native code
  - Requires changes to the Java VM

Different Layers of Virtualization

• Hardware-level	• LPAR, nPar
• Hypervisor-level	• Xen, VMware, vPar, ...
• OS-level	• Solaris, Virtuozzo
• Java VM-level	• MVM



OS Level vs. Hypervisors

- Solaris containers
  - Creation of virtual servers
  - Resource isolation
  - Resource metering
  - Resource control

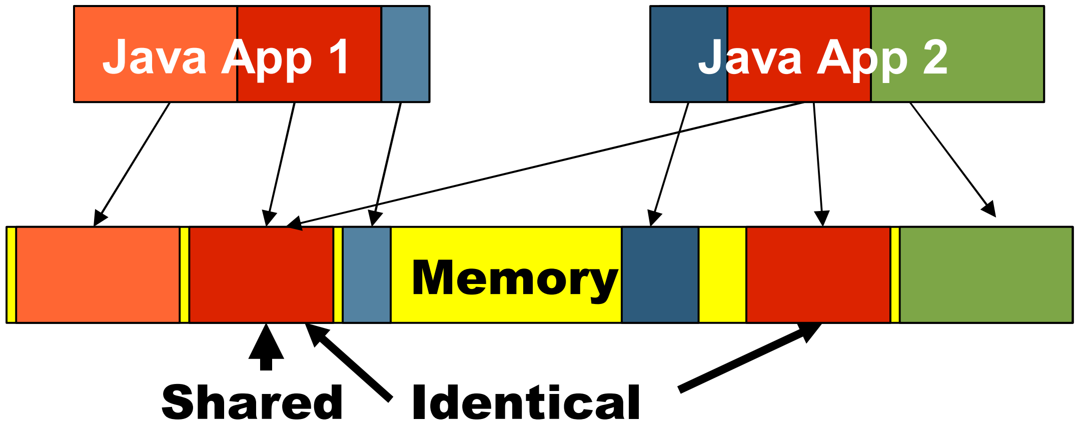
- VMware
  - Creation of virtual servers
  - Resource isolation
  - Resource metering
  - Resource control
  - Suspend/resume
  - Live migration

Can Project Bare Metal Help?

- Yes, we think so
  - Resource control—already built-in
  - Can be modified to control native code safely
  - No additional changes to the Java VM necesary
  - Reduced memory footprint
  - Startup-time reductions?

Future: Shared Memory Java VMs

- Reduce memory footprint
  - Use hypervisor page-sharing functionality
  - Even ”read-mostly” memory can be shared
- Share identical memory between Java VMs
  - Java VMs cooperate and reorder memory to be merged
  - Even share identical but different Java objects



JVM-Level vs. Cooperative Hypervisor

- MVM
  - Efficient isolation for multiple Java VMs
  - Resource metering
  - Resource control

  - Faster startup time?
  - Has smaller footprint?

- Bare Metal Approach
  - Efficient isolation for multiple Java VMs
  - Resource metering
  - Resource control

  - Thaw from frozen state
  - Migrates live instances
  - Isolates native code

Agenda

Hypervisor optimized server Java technology

Project Bare Metal Overview

Looking Under the Hood

Performance Analysis

Virtualization Layers and Isolation

Going Forward

Summary

Maturing the Bare Metal Technology

- Getting JRockit certified on Bare Metal
- Multiprocessor support
- Improved filesystem support
- Driving out bugs and bottlenecks
- JRockit takes advantage of Bare Metal
- Hypervisor extensions for Java technology
- Implement JSR-121

Resource Management and Java VMs

- Resource Management has been poor
  - Ability to measure how much resources the Java VM is using was introduced in Java 5
- JRockit is extending Resource Management
  - To control how much resources that are used
  - To measure resources usage at the thread-level
- JSR 284 will standardize Resource Management functionality

Agenda

Hypervisor optimized server Java technology

Project Bare Metal Overview

Looking Under the Hood

Performance Analysis

Virtualization Layers and Isolation

Going Forward

Summary

Summary

- Project Bare Metal optimizes Java code execution on a hypervisor
- Hypervisors can divide a physical machine into multiple virtual machines
- Bare Metal can be an alternative to MVM on the server-side to let many Java VMs run efficiently on the same box
- Initial performance of Java technology on Bare Metal is promising but many optimizations remain

For More Information

- Bare Metal
    - BEA dev2dev—http://dev2dev.bea.com
- Virtualization software
    - VMware—http://www.vmware.com
    - Xen—http://www.xensource.com
- Similar or related products and projects
    - Squawk—http://research.sun.com/projects/squawk/
    - JNode—http://www.jnode.org
    - Sanos—http://www.jbox.dk/sanos
    - Azul Systems—http://www.azulsystems.com

Call To Action

- Are you interested in evaluating Bare Metal?
  - Are you running VMware ESX Server/Xen?
  - Java EE 5?
  - No native code?
  - Not heavily dependent on file system performance?
- Contact us!
  - joakim.dahlstedt@bea.com

Q&A

DEMO

Project Bare Metal Live

“Bare Metal”—Speeding Up Java™ Technology in a Virtualized Environment

Joakim Dahlstedt

CTO, Java Runtime Products Group
BEA Systems
http://www.bea.com

TS-3792