Kitten: A Lightweight Operating System for Ultrascale Supercomputers

Presentation to the New Mexico Consortium
Ultrascale Systems Research Center
August 8, 2011

Kevin Pedretti
Senior Member of Technical Staff
Scalable System Software, Dept. 1423
ktpedre@sandia.gov

SAND2011-5626P

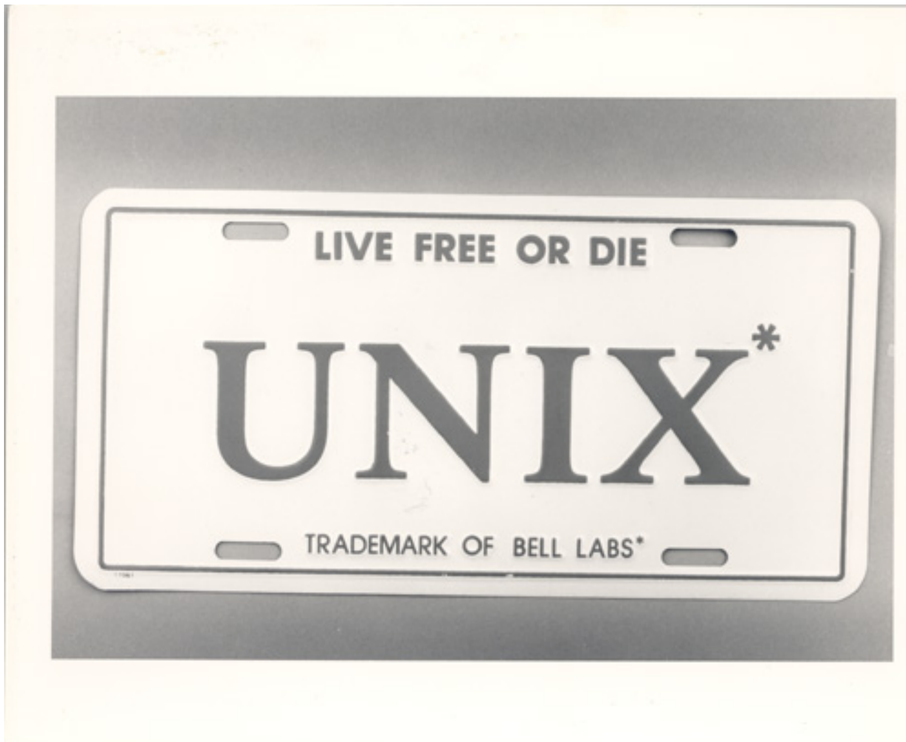
Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

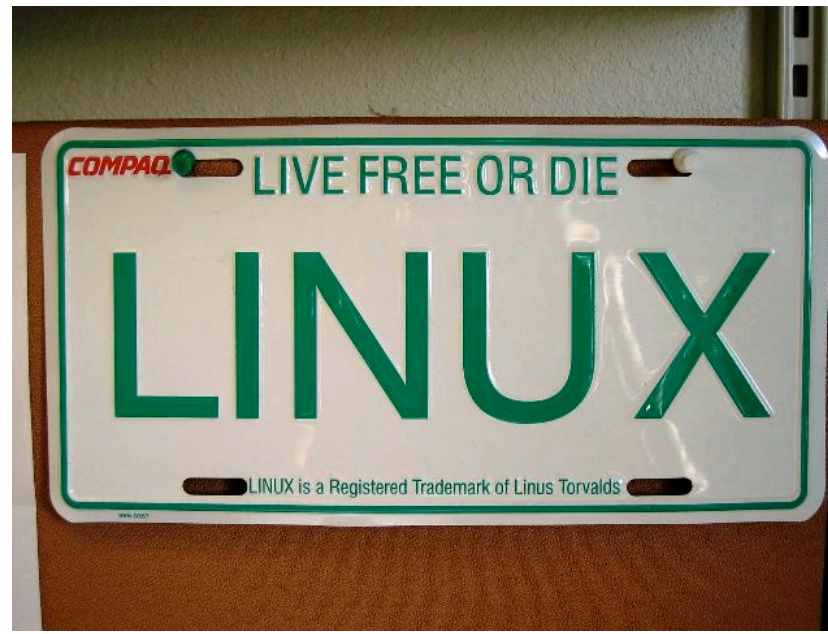
Outline

- Introduction
- Kitten lightweight kernel overview
- Future directions
- Conclusion

Sandia National Laboratories

Four+ Decades of UNIX





Operating System = Collection of software and APIs
Users care about environment, not implementation details
LWK is about getting details right for scalability,
both within a node and across nodes

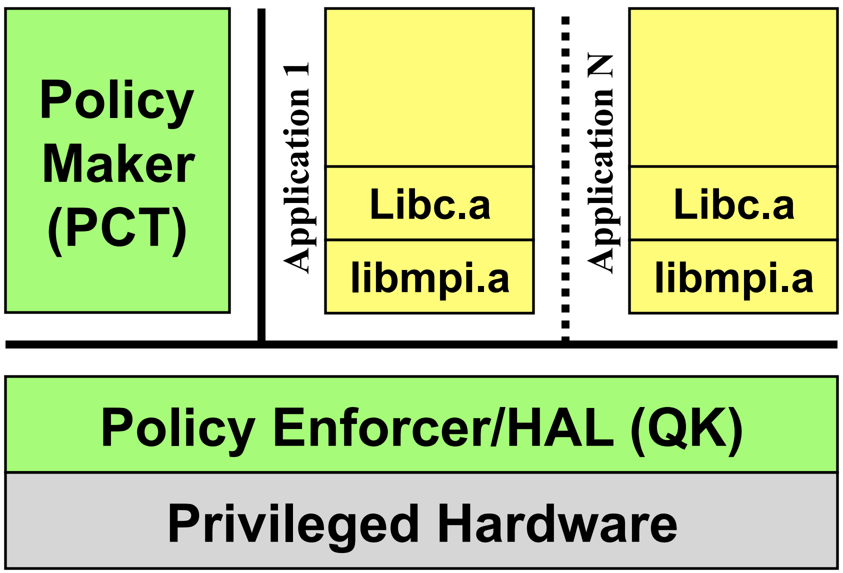
Sandia National Laboratories

Sandia Lightweight Kernel Targets

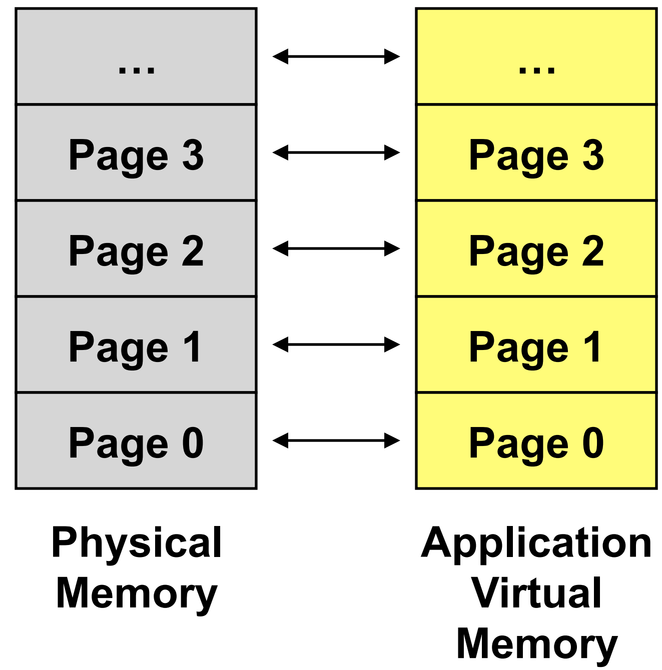
- Massively parallel, extreme-scale, distributed-memory machine with a tightly-coupled network
- High-performance scientific and engineering modeling and simulation applications
- Enable fast message passing and execution
- Offer a suitable development environment for parallel applications and libraries
- Emphasize efficiency over functionality
- Move resource management as close to application as possible
- Provide deterministic performance
- Protect applications from each other

Lightweight Kernel Overview

Basic Architecture



Memory Management



- POSIX-like environment
- Inverted resource management
- Very low noise OS noise/jitter
- Straight-forward network stack (e.g., no pinning)
- Simplicity leads to reliability

Lightweight Kernel Timeline

1990 – Sandia/UNM OS (SUNMOS), nCube-2

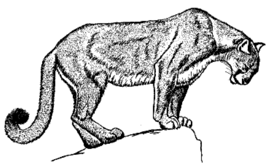
1991 – Linux 0.02

1993 – SUNMOS ported to Intel Paragon (1800 nodes)

1993 – SUNMOS experience used to design Puma

- **First implementation of Portals communication architecture**

1994 – Linux 1.0



1995 – Puma ported to ASCI Red (4700 nodes)

- **Renamed Cougar, productized by Intel**

1997 – Stripped down Linux used on Cplant (2000 nodes)

- **Difficult to port Puma to COTS Alpha server**
- **Well-defined Portals API**

2002 – Cougar ported to ASC Red Storm (13000 nodes)

- **Renamed Catamount, productized by Cray**

2004 – IBM develops LWK (CNK) for BG/L/P/Q (2011 Sequoia 1.6M cores)

2005 – IBM & ETI develop LWK (C64) for Cyclops64 (160 cores, dance hall)

2007 – Kitten development begins, Aug. 2007

2007 – Cray releases Compute Node Linux for Cray XT3/4/5/6 systems

2009 – Tilera develops "Zero Overhead Linux" for TILEPro (64 cores, 2D mesh)

2009 – Argonne ZeptoOS Linux for BG/P, "Big Memory" Linux kernel patches

Sandia National Laboratories

Outline

- Introduction
- **Kitten lightweight kernel overview**
- Future directions
- Conclusion

Many Drivers for Starting Fresh

- **SUNMOS/Puma/Cougar/Catamount shortcomings**
  - Closed source (*) and export controlled
  - Limited multi-core support
  - No support for multi-threaded applications (*)
  - Custom glibc port and compiler wrappers
  - No NUMA / PCI / ACPI / APIC / Linux driver support / Signals / …
  - No Virtual Machine Monitor capability
- **Needed more modern platform for research**
  - Less complex code-base enables rapid prototyping
  - Exascale R&D in runtime systems and programming models
  - Bring-up of new chips, simulated or real (IBM CNK argument)

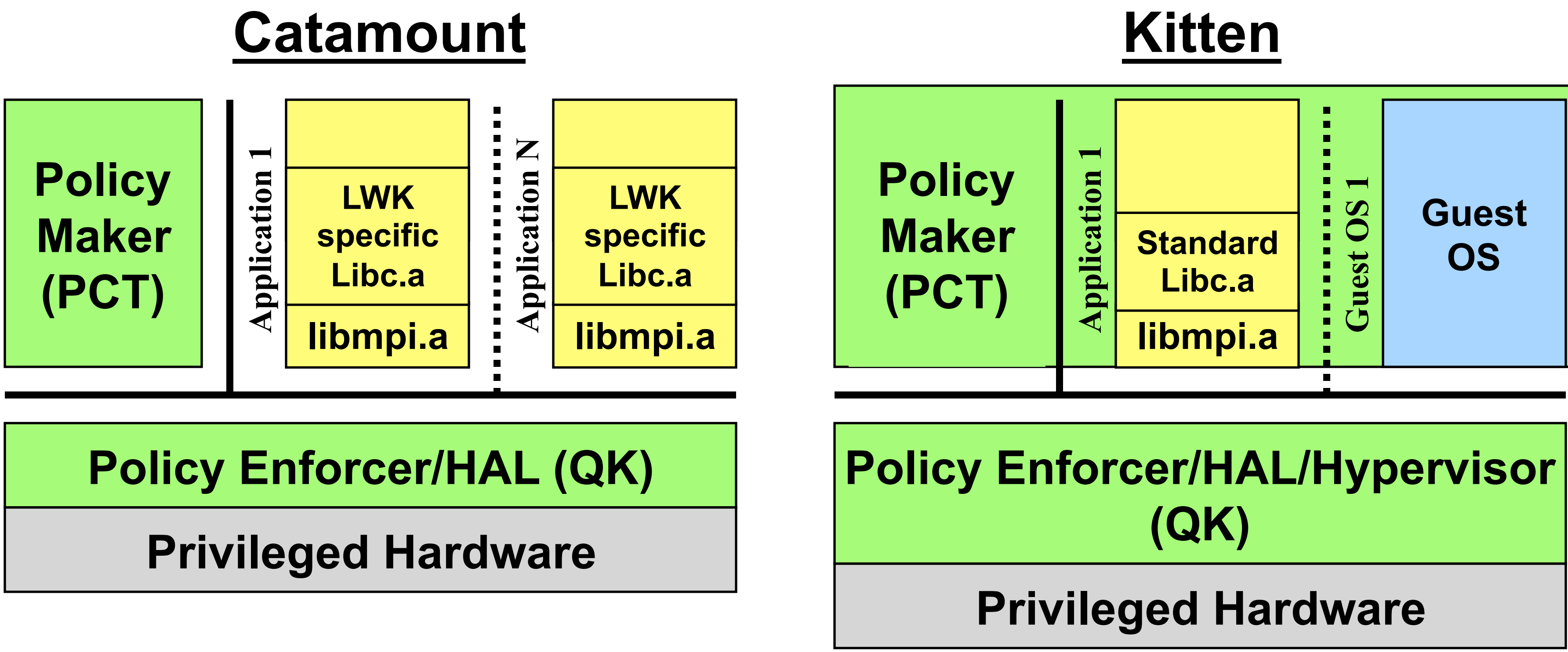
Focus on what's important rather than working around problems that shouldn't exist

Kitten Lightweight Kernel

- **Maintain important characteristics of prior LWKs**
- **Better match user, vendor, and researcher expectations -> Looks and feels like Linux, support multicore + threads**
- **Available from http://code.google.com/p/kitten**
- **FY08-10 LDRD project, currently CSSE + ASCR funded**

Sandia National Laboratories

LWK Architecture



Major changes:

- QK includes hypervisor functionality
- QK provides Linux ABI interface, relay to PCT
- PCT provides function shipping, rather than special libc.a

Leverage Linux and Open Source

- **Repurpose basic functionality from Linux Kernel**
  - Hardware bootstrap
  - Basic OS kernel primitives
  - PCI, NUMA, ACPI, IOMMU, …
- **Innovate in key areas**
  - Memory management, multi-core messaging optimization
  - Network stack
  - Leverage runtime feedback
  - Fully tick-less operation, but short duration OS work
- **Boots identically to Linux, drop-in replacement for CNL**
- **Open platform more attractive to collaborators**
  - Collaborating with Northwestern Univ. and Univ. New Mexico on lightweight virtualization for HPC, http://v3vee.org/
  - Potential for wider impact

POSIX Threads + OpenMP Support

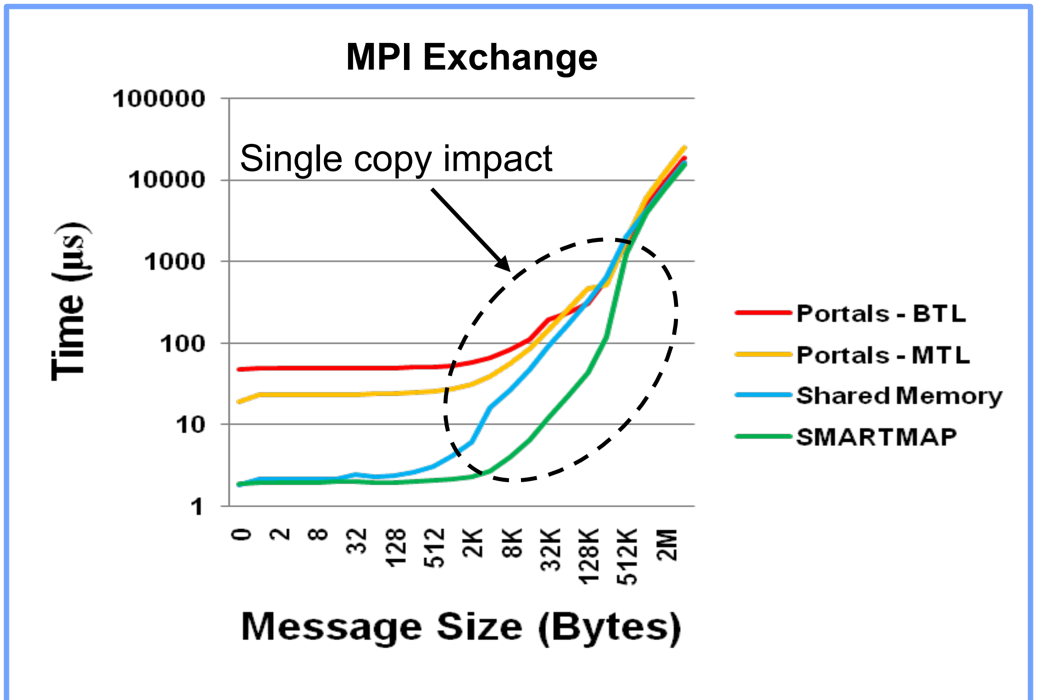
- **Kitten user-applications link with the standard GNU C library installed on the Linux host**
- **GNU C includes POSIX threads implementation called NPTL**
- **NPTL relies on Linux futex() system call, Kitten supports**
  - Futex() = Fast user-level locking
  - Atomic instructions used to manipulate futexes
  - Only trap to OS Kernel when futex is contended, uncontended case requires no syscalls
- **Compilers typically build OpenMP support on top of POSIX threads -> Kitten supports OpenMP**
- **Kitten supports many threads per core**
  - Each core has private run queue
  - Round-robin preemptive scheduling
  - No automatic load-balancing between cores

Key Memory Management APIs

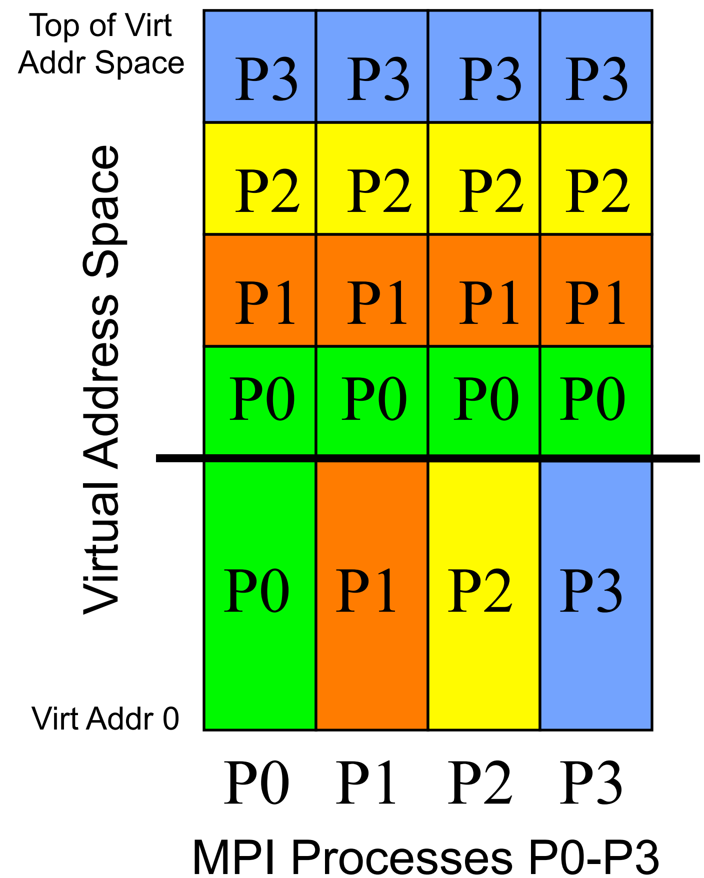
- **Address space creation/destruction**
  - `extern int aspace_create(id_t id_request, const char *name, id_t *id);`
  - `extern int aspace_destroy(id_t id);`
- **Create virtual memory regions**
  - `extern int aspace_add_region(id_t id, vaddr_t start, size_t extent, vmflags_t flags, vmpagesize_t pagesz, const char *name);`
- **Allocate physical to virtual memory**
  - `extern int aspace_map_pmem(id_t id, paddr_t pmem, vaddr_t start, size_t extent);`
- **Map one address space into another**
  - `extern int aspace_smartmap(id_t src, id_t dst, vaddr_t start, size_t extent);`

SMARTMAP Eliminates Unnecessary Intra-node Memory Copies

- Basic Idea: Each process on a node maps the memory of all other processes on the same node into its virtual address space
- Enables single copy process to process message passing (vs. multiple copies in traditional approaches)



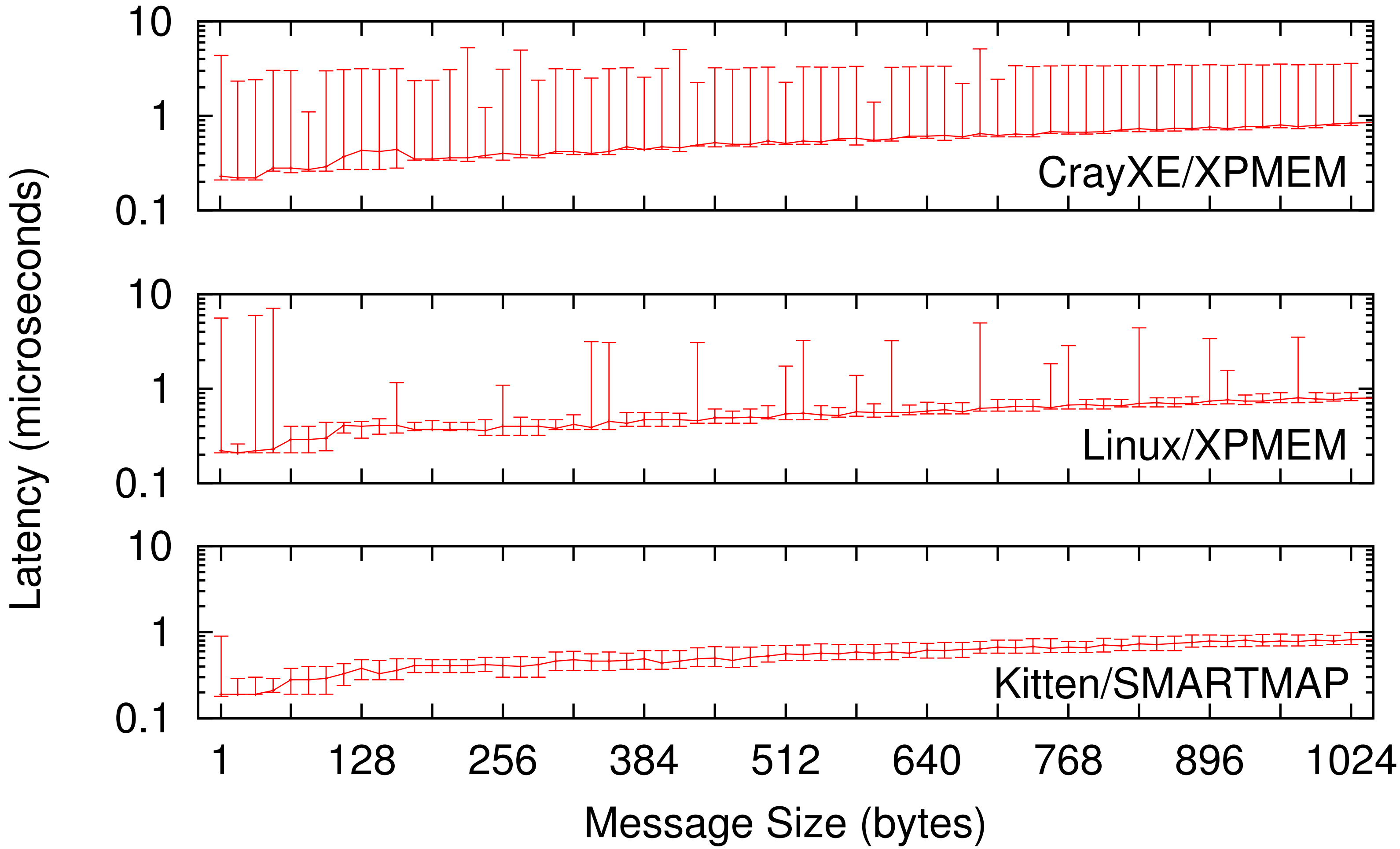
SMARTMAP Example



For more information see SC'08 paper

Sandia National Laboratories

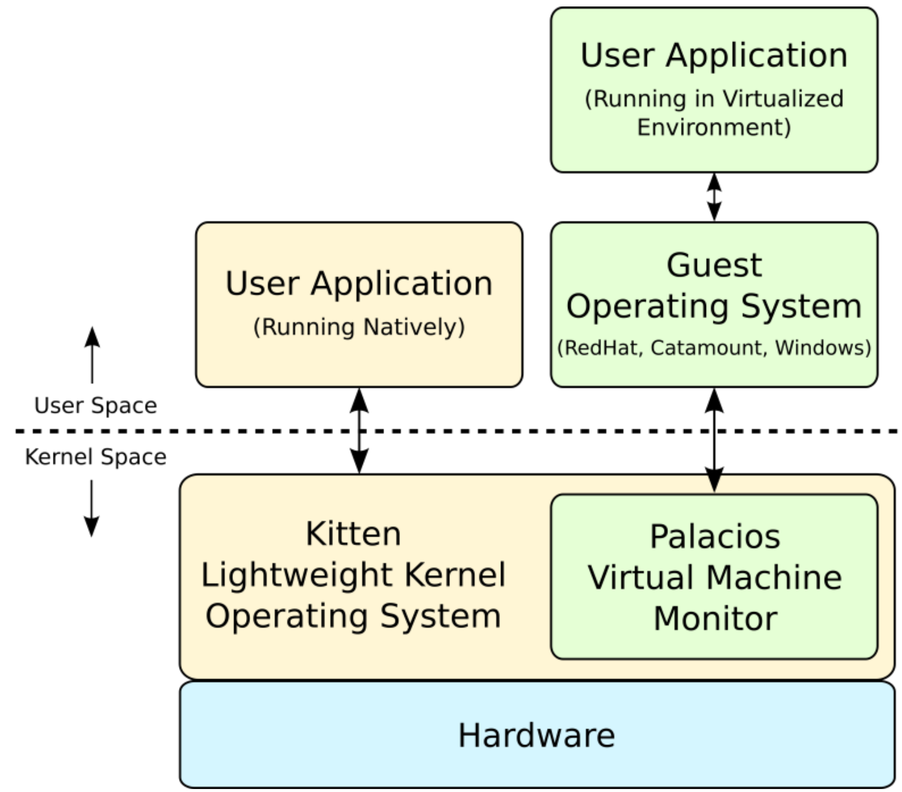
SHMEM Ping-Pong Latency
Kitten Demonstrates Low Variability



Sandia National Laboratories

Kitten Provides a Scalable Virtualization Environment for HPC

- Lightweight Kernels (LWK) traditionally have limited, fixed functionality
- Kitten LWK addresses this limitation by embedding a virtual machine monitor (collaboration with Northwestern Univ. and Univ. of New Mexico)
- Allows users to “boot” full-featured guest operating systems on-demand
- System architected for low virtualization overhead; takes advantage of Kitten’s simple memory management
- Conducted large scale experiments on Red Storm using micro-benchmarks and two full applications, CTH and Sage



For more information see IPDPS’10 + VEE’11 papers

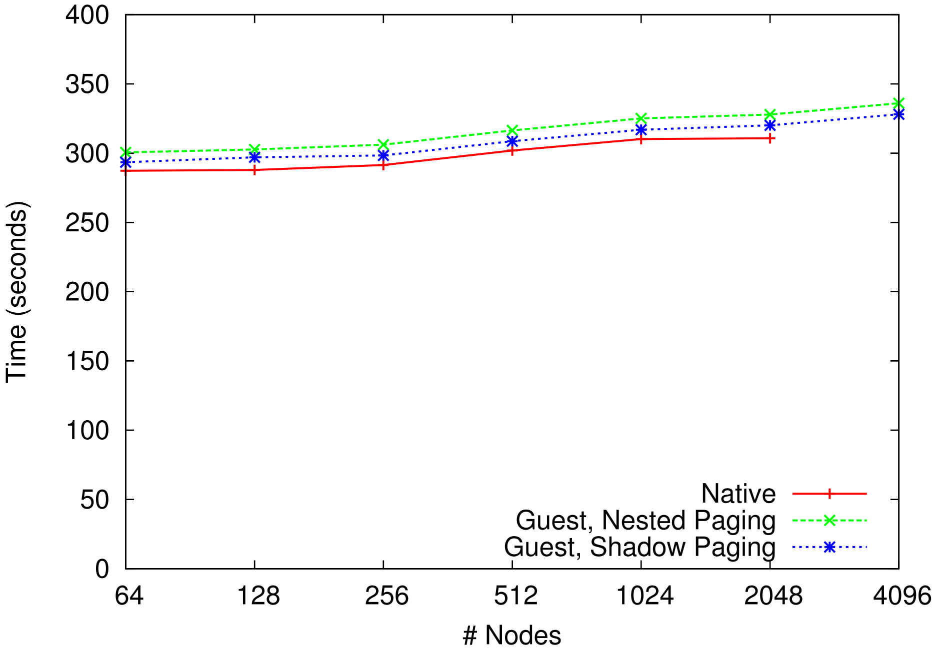
HPC Virtualization Has Many Use Cases

- X-Stack researchers
  - Enable large-scale testing without requiring dedicated system time
  - Emulate and evaluate novel hardware functionality before it is available (e.g., global memory)
  - Enable hardware/software co-design
- End-users
  - Load full-featured guest OS, or app-specific OS
  - Dynamically replace runtime with one more suitable for the user's workload (e.g., a massive number of small jobs)
  - Cyber-security experiments using commodity OSes, run multiple OSes per compute node
  - System administrators test new vendor software without taking machine out of production

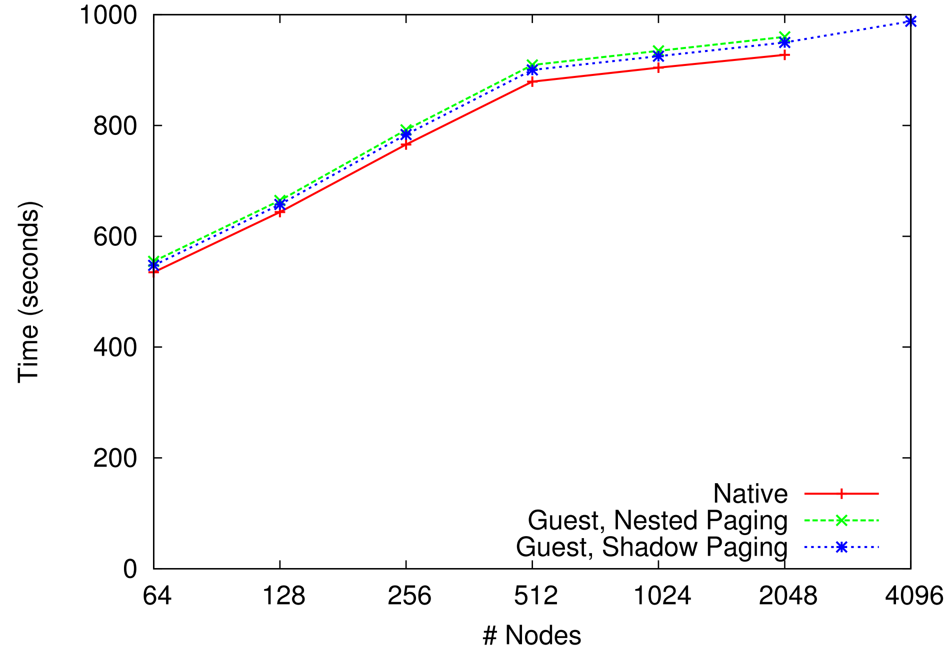
Highlight Results from Red Storm Virtualization Experiments

Native is Catamount running on ‘bare metal’, Guest is Catamount running as a guest operating system managed by Kitten/Palacios

CTH
shaped charge, weak scaling



Sage
timing_c, weak scaling



Performance when executing in virtual machine within 5% of native

Sandia National Laboratories

Outline

- Introduction
- Kitten lightweight kernel overview
- Future directions
- Conclusion

Sandia National Laboratories

Future Directions

- OS support for exascale runtime systems
  - Functional partitioning of cores (network progress engines, I/O, resiliency, …)
  - Exploit SMARTMAP capability
  - Continue to get out of the way, let runtime/app manage resources
- Performance experiments
  - Infiniband performance, eliminate memory pinning
  - Cray XE6
- GPU support
- Continue work on HPC-focused virtual machine monitor capability, Palacios

Conclusion

- ***Kitten is a modern, open-source LWK platform that supports multi-core processors (N cores, multiple threads per core), advanced intra-node data movement (SMARTMAP), current multi-threaded programming models (via Linux user-space compatibility), commodity HPC networking (Infiniband), and full-featured guest operating systems (Palacios virtualization)***

- ***Well-positioned for exascale HW/SW co-design and collaboration***

Sandia National Laboratories

Acknowledgments

- Patrick Bridges (U. New Mexico)
- Ron Brightwell (Sandia)
- Peter Dinda (Northwestern U.)
- Kurt Ferreira (Sandia)
- Trammell Hudson (OS Research)
- Jack Lange (U. Pittsburgh)
- Mike Levenhagen (Sandia)
- Alex Merritt (Georgia Tech)

Sandia National Laboratories