A set of new application domains is emerging...

- pushing dynamic content generation code into caches and CDN
- pushing "grassroots" or third-party Internet services into hosting infrastructure
- deploying new routing and content location services into peer-to-peer systems or application-layer meshing infrastructure (e.g., FreeNet, RCN)
- deploying measurement experiments into existing infrastructure (e.g., NLAN)
- ... and the list goes on.

These applications all share several properties:

- they require the ability to safely execute untrusted code
- they benefit from the ability to scale to a large number of concurrent protection domains per physical host (hundreds, or perhaps even thousands)
- they necessitate a large degree of resource multiplexing, and correspondingly, they require performance isolation
- the degree of information sharing between applications is small

The dangers of virtualization...

- Virtualization requires trapping and emulating "privileged" instructions, such as I/O
- Virtualization requires emulated hardware

Longer-term:
- attempt to scale the Yakima VMM up to 500 VMs, and see what breaks, and see what breaks
- understand the strengths/limitations of a single processor VMM/VM system

Near term:
- use a high-resolution image of the world
- Denali (VMM) view of the world
- Key Idea

Elements of Denali's para-virtual machine architecture

- instruction set: based on x86 instruction set
- most instructions execute natively without requiring monitor intervention

Key research question: How should we design a virtual architecture to support a large number of concurrent VMs (rather than the other way around)?
**Denali: Lightweight virtual machines for distributed and networked systems**

Andrew Whitaker, Marianne Shaw, and Steven D. Gribble  
Department of Computer Science and Engineering, The University of Washington

*SOSP 2001 Poster session submission*

Advances in networking and computing technologies have accelerated the proliferation of infrastructure such as content distribution, caching, middleware services, and network measurement testbeds. In conjunction with this, several application domains have begun to emerge that are not well-supported by existing technologies, such as distributing dynamically generated web content, rapidly deploying untrusted Internet services into hosting infrastructure, and injecting network measurement code into network experimentation infrastructure.

Although several sandboxing technologies have been proposed, none have the combination of water-tight isolation and the ability to scale to a large number of protection domains required by these new applications. The Denali project seeks to remedy this situation by constructing a software platform that provides both performance and security isolation to hundreds (or perhaps thousands!) of untrusted applications/OSs executing concurrently on a single physical host.

To achieve this high degree of scalability and isolation, we are exploring the use of virtual machine monitors (VMMs). A VMM is a software layer that runs immediately on top of the hardware/software boundary, virtualizing all resources to give higher-level virtual machines (VMs) the illusion of their own dedicated physical machine. While VMs are known to have strong isolation properties, they have traditionally been regarded as a heavyweight mechanism, permitting only a small number of virtual machines to execute concurrently. As a step towards building lightweight VMs, we are exploring the notion of "para-virtualization", in which the virtual architecture that we expose to VMs is similar to but slightly divergent from the underlying physical architecture. Allowing this divergence permits optimizations that improve performance and scalability, and reduce implementation complexity.

Using the Flux OSKit, we have implemented an x86 VMM, called Yakima, which directly runs on the bare hardware. Yakima uses para-virtualization in several ways, including the delivery of batched interrupts upon context-switching into a VM, providing a reduced set of simplified I/O devices, and the elimination of virtual memory and user/kernel boundaries in a VM. We have developed an OS library (similar to an Exokernel libOS) tuned to our para-virtual architecture. This library, which contains preemptive threads and a networking stack, has been used to implement a Yakima-supported supervisor VM (able to create, destroy, and allocate system resources to individual VMs) and also a web server application.

We are currently in the process of quantifying Yakima's performance and addressing the challenge of resource management across virtual machines. To fully isolate one VM from another, each VM's resource usage (e.g., CPU consumption, I/O rates, memory footprint) must be bounded by the VMM. Once we have completely isolated lightweight VMs, we intend to leverage this new mechanism to explore several research topics: (1) VMs as sandboxes to enable web servers to dynamically inject new content-generation code into CDNs or web caching systems; (2) using VMs to enable untrusted code authors to upload new Internet services into a virtual hosting platform; and (3) the role of VMs as a resource container in a cluster-of-workstations to create the effect of isolated, dynamically resizable "virtual clusters" within a physical cluster.