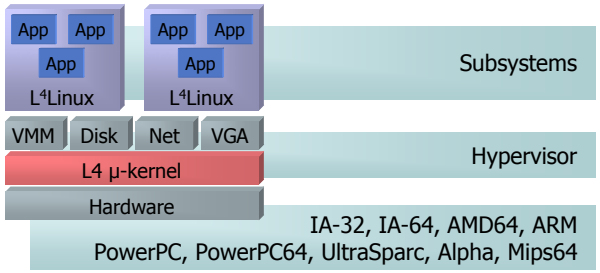
Flexible and Scalable Virtual Machines

Volkmar Uhlig Joshua LeVasseur Espen Skoglund Uwe Dannowski

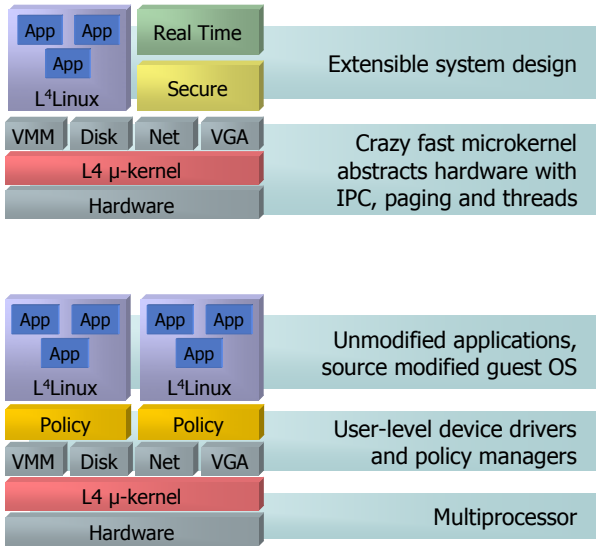
System Architecture Group
Universität Karlsruhe

Virtualization Architecture

The guest OS runs on a virtual architecture defined by the hypervisor. No special hardware features are needed to efficiently multiplex the virtual machines.

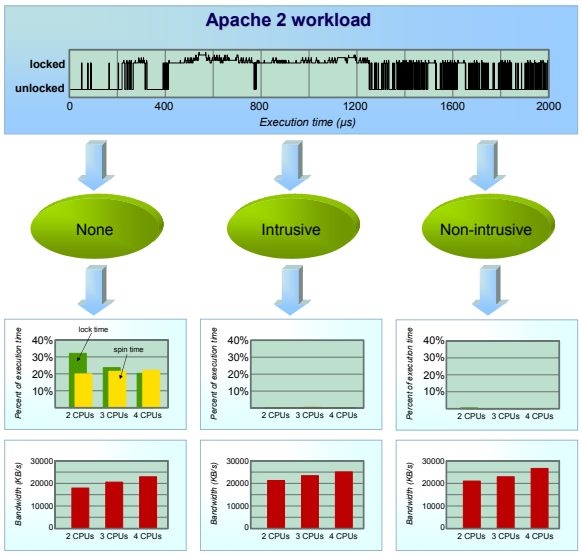


The guest kernel and applications can use the fundamental microkernel primitives to construct flexible and untrusted OS extensions, showing a way out of the legacy dependence trap in OS development.



Lock Holder Preemption Avoidance

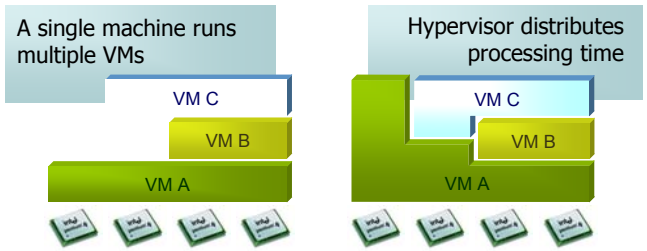
Flexible scheduling of multi-processor virtual machines can preempt guest kernel lock-holders. This increases lock-holding and spinning times.



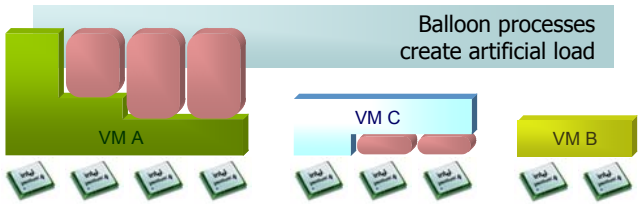
We can modify the guest kernel to provide hints to the hypervisor saying that it holds a lock (intrusive). Or, the hypervisor can monitor the guest to determine when the guest kernel holds locks (non-intrusive), e.g., when the guest runs at “privileged-level”. The hypervisor uses this knowledge to avoid preempting lock-holders.

Time Ballooning

The hypervisor allows flexible processor resource allocation policies. This can lead to uneven distribution of processing time within a single virtual machine, giving the illusion of non-uniform CPU speeds.



Commodity operating systems can not handle asymmetric CPU speeds, resulting in suboptimal internal load distribution. The hypervisor injects balloon processes into the guest to create artificial load on slower processors. The guest kernel redistributes real processes according to actual processing resources.



Virtual machine architecture powered by L4Ka::Pistachio.

http://L4Ka.org/

Universität Karlsruhe (TH)

intel