Denali Isolation Kernel

Introduction

- What is meant by Scale?
    - By "Scale" they mean that they want to increase the number of VMs running on a single machine as much as possible without taking too much of a performance hit.
- What is an isolation kernel? (how is it different/same than a VMM)
    - The goal of an isolation kernel is to provide total isolation of processes w/ no sharing. This is in contrast to normal OS's which encourage sharing through global file systems and see a "good enough" solution to isolation
    - Runs a single process with a library OS in each VM

Case for IKs

- What is the key motivation for IKs?
    - Total isolation of processes
    - Reduce complexity by only doing a portion of the virtualization. Creating a simple VMM is not tough, but creating the perfect system is extremely difficult so only simple interfaces for hardware were created and everything else ignored.
    - Support a large number of internet services on a single machine. They want lightweight specialized operating systems, but this idea didn't really catch on. Partly because it would take a lot of time to port applications and Operating Systems to this structure. Also, most users don't want to have their services idle for 9,999/10,000 CPU cycles. Basically, the researchers just wanted to see how far they could push the boundaries instead of creating a commercial product.
- Why not just use an OS? (w/processes)
    - Provides better security isolation than an Operating System in order to avoid the lower-level attacks. The researchers figured that a simple, smaller system would have fewer security holes and a broken service would have no effect on other VM's.
- Why are "Zipf workloads" relevant?
    - Zipf: You have a small number of applications that run a lot and a large number of applications that run a little each creating a Log style distribution. The result was a long queue of requests that can provide great throughput, but provides very poor latency. The latency can help determine how many VM's can be run on a single machine. The focus of the evaluations was on throughput and ignored latency.

Danali Design

- How is the ISA different/similar to x86? Why is this important
    - Only uses a subset of x86 and they run it directly on the hardware. They wanted to eliminate the difficult to employ instructions to make it easier to maintain user/kernel space boundaries.
- New instructions: What are they, why needed?
    - To assist the VMM, they added two new instructions to serve as hints:

Denali Isolation Kernel

        - Idle_with_timeout: Very similar to a "yield to" command
            - Tells the IK that the virtual processor is idle for the moment, so it doesn't need to worry about handling processing for this vProcessor and gives up its CPU time in exchange for a higher priority.
            - The timeout becomes important to wake the VM back up after a certain time in case it needs to do something like retransmit a lost TCP packet
        - Terminate virtual processor
- Memory Architecture: Describe. How is this similar to/different than typical VMM?
    - Memory Architecture, there is no Virtual Memory for the guest, but rather one single large address space. This means that the VM does not really control its own memory. This works well enough in the static environment, but how would it hold up for advanced services? There are lots of trust issues here.
- IO/devices: what are virtual interrupts? (how different from physical interrupts)
    - When a physical interrupt occurs, Denali will trap those and then issue virtual interrupts to the appropriate VMs
    - They batch the virtual interrupts until the VM's quantum comes up to process the interrupt. In addition, they added a packet_send and packet_receive function to abstract the lower level network interface. Otherwise, incoming packets are batched creating potential issues for time based TCP.

Denali Implementation

- Two Policies for CPU: gatekeeper and scheduler; what are these, why are they needed?
    - Gatekeeper – chooses a subset of active machines to admit into the system
    - Scheduler – controls context switching among admitted machines
- Memory Management: Static Swap region per VM: why needed?
    - Static swap region per VM and 16 MB per VM which is enough for serving a small number of webpages. Essentially, the authors were expecting extremely small almost impractical services.
- When are page tables allocated/used?
    - They are allocated when the VM is initialized.
- I/O devices: VMs can only process packets during their quantum; why is this important?
    - This is important to prevent constant context switching (which could cause thrashing with 1000 VMs) when new packets become available for a VM that is not loaded by the Gatekeeper

Ilwaco Library OS

- What is a "Library OS"?
    - A library OS is a software library that a compiled program is linked to which provides functionality typically handled by the OS
    - Contains

        - TCP/IP stack: Simpler to have it as a library in the file system that links in to the application. This creates performance issues, but increases simplicity.
        - Thread package
        - I/O drivers
- What are pros/cons of this approach?
    - It provides a smaller footprint, however there is much less hardware protection
- Is there a file system?
    - There is no real file system, just disk blocks which are not reused.