10/3/15

PAGE NO.
DATE: / /

Title → A virtualization solution for BYOD with Dynamic Platform context Switch

Authors → Yaozu Dong, Jun Jie Mao, Hai Bing Guan, Jian Li, Yu Chen

Reviewed by :→ Sahil Singh (143050001)

Summary :- This paper tries to address some of the concerns of the BYOD trend where employees are allowed to use same device for work and personal use. Security and isolation of the two worlds in the device is of primary concern, which the paper tackles by allowing the worlds to run as different OSes, managed by an underlying hypervisor. The sol.ⁿ is built using Xen.

Detailed notes

The VMM allows instantiation of several VMs, however only one of them is allowed to run (consume CPU cycles) while others remain in a dormant state and become active either when a user switches to them or an event occurs, such as a message

is received, which is destined to one of the dormant VMs.
The VM currently running is known as FVM (foreground VM) while others are called BVM (Background VM).
The proposed VMM is ~~Designed for storage~~ intended for use on Intel ~~AOD~~ Atom Soc, which is based on x86 core.
VMs are paravirtualized.

~~Memory and storage isolation~~

H/w sharing:-

FVM has exclusive access to H/w, thus any VM running as FVM is able to use its native drivers.

When a switch is to be made from one VM to another the OS of currently running VM (FVM) is instructed to suspend devices, and to store device state to memory (this feature is already present in

native drivers).

This suspend and resume is costly process and function level resetting speeds up the process.

Events and interrupts

Virtual APIC interface of Xen is used to generate timer interrupts for Guests.

Each VM has a module called switch Agency which on occurance of an event, such as reception of an SMS destined for one of the BVMs, initiates a context switch to it so that it can handle the event.

Memory and Storage

Memory and storage is statically partitioned for the VMs.

To prevent one guest from accessing to memory of another guest IMR (isolated memory regions) feature provided by Intel ATOM

is used.

Storage accesses by guests are translated by the VMM by trap & emulate mechanism so that all reads and writes by a guest are limited to its partition.

Power management

Since FVM has exclusive access to all the hardware, the power management policies can be directly applied by Guest VMs. However since the VMs run in Ring 1 and not in Ring 0 access to CPU state traps to VMM, which directly makes the intended changes.

Experimental evaluation

In all the tests performance of Guest VM was similar to non virtualized execution. In particular the tests that involved writing to MMC incurred additional overhead.

Two tests which stand out are 'Pro Std vs FSAA' (GL Benchmark) and power consumption in flight mode, in which the virtualized setup fared better.

Positive Points

(i) Giving exclusive access to FVM of all hardware resources allows the native drivers of VMs to be reused.

(ii) Only one VM runs at a time, while others are dormant, this saves battery.

Negative points

(i) The experiments where virtualized OS performed better than non-virtualized OS need more investigation.

(ii) [illegible]

(iii) Microbenchmarking needs to be done to figure out which operations of OSes are most affected.

Extensions

FVM uses its native drivers to access H/w, while in Xen or KVM a different driver model is used. A comparison of performance b/w them would help test the advantage of the driver model used in this paper, as providing exclusive access to H/w adds additional overhead to context switching VMs.