ARM Architecture-based System Virtualization: Xen ARM open source software project

Sang-bum Suh Vice President, Jae-min Ryu Research Staff
{sruk.suh, jm77.ryu}@samsung.com
S/W Platform Team
DMC Research Center
SAMSUNG Electronics Co.

2011.08.

Xen Summit North America 2011
Agenda

- Overview
  - History of Xen ARM
  - Use Cases

- Xen ARM: Core
  - Xen ARM Virtualization
  - Performance Comparison

- Xen ARM Application: Security
  - Mobile Malware
  - Access Control

- Xen ARM Application: Real-time
  - Xen ARM: Preemption
  - Real-time Performance
Overview
History of Xen ARM

'04
x86 Xen Hypervisor Release
(Cambridge university)

'08
Xen ARM 1\textsuperscript{st} Release:
ARM9 Xen Hypervisor, Mini-OS
(Samsung)

'09
Xen ARM 2\textsuperscript{nd} Release:
Paravirtualized Linux kernel
(v2.6.24), Xen tool
(Samsung)

'10
Xen ARM 3\textsuperscript{rd} Release:
ARM11MPCore Support
(Samsung)

Xen ARM 4\textsuperscript{th} Release:
Performance Optimization
(Samsung)

Xen-ARM Open Source Community

- Samsung leads the Xen ARM project
- \url{http://wiki.xensource.com/xenwiki/XenARM}

Supported Hardware & Guest OS

- ARM926EJ-S (i.MX21, OMAP5912)
- Xscale 3rd Generation Architecture
  (PXA310, Samsung SGH-i780)
- ARM1136/ARM1176(Core Only)
- Goldfish (QEMU Emulator)
- Versatile Platform Board
- ARM11MPCore (Realview PB11MP)
- Cortex-A9 (Tegra250)

- Linux v2.6.11, v2.6.18, v2.6.21, v2.6.24, v2.6.27
  (multicore supported)
- uC/OS-II
Use Cases

1. **HW Consolidation**: AP (Application Processor) and BP (Baseband Processor) can share multicore ARM CPU SoC in order to run both Linux and Real-time OS efficiently.

2. **OS Isolation**: Important call services can be effectively separated from downloaded third party applications by Xen ARM combined with access control.

3. **Rich User Experience**: Multiple OS domains can run concurrently on a single smartphone.
Xen ARM: Core
Xen ARM Virtualization

Goals

- Light weight virtualization for secure 3G/4G mobile devices
  - High performance hypervisor based on ARM processor
  - Fine-grained access control fitted to mobile devices

Architecture of Xen-ARM

VM 0
- Application
- Backend Drivers
- Native Drivers
- VM Interface

Secure Xen-ARM hypervisor
- Domain Manager
- Resource Allocator
- Access Control

VM 1
- Application
- Frontend Drivers
- VM Interface

Hardware
- Peripheral Devices
- CPU
- System Memory
- Flash Memory
Xen ARM Virtualization

**CPU virtualization**
- Virtualization requires 3 privilege CPU level, but ARM supports 2 level
  - Xen-ARM mode: supervisor mode (most privileged level)
  - Virtual kernel mode: User mode (least privileged level)
  - Virtual user mode: User mode (least privileged level)

**Memory virtualization**
- VM’s own memory should be protected from others
  - Xen-ARM switches VM’s virtual address space using MMU
  - VM is not allowed to manipulate MMU directly

**I/O virtualization**
- Split driver model of Xen-ARM
  - Client & Server architecture for shared I/O devices
    - Client: frontend driver
    - Server: native/backend driver
Performance Comparison

Micro-benchmark Results

- **Evaluation Environments:** Samsung Blackjack Phone
  - CPU: Xscale PXA310, 624MHz
  - L1 Cache: 32KB + 32KB
  - L2 Cache: 256KB (Disabled)
  - Memory: 128MB
  - Guest OS: Linux-2.6.21

LMBENCH Micro Benchmark (Bandwidth)

Higher is better

LMBENCH Micro Benchmark (latency)

Lower is better
**Performance Comparison**

**Micro-benchmark Results**

- **Evaluation Environments:** nVidia Tegra250
  - CPU: Cortex-A9 1GHz Dual Core
  - L1 Cache: 32KB + 32KB
  - L2 Cache: 1MB
  - Memory: 1GB
  - Guest OS: Linux-2.6.29

![Chart showing performance comparison between Native Linux and Para-virtualized Linux](chart.png)

- Lower is better
Performance Comparison

**Benchmark Results**

- **Evaluation Environments:** Samsung Blackjack Phone
  - CPU: Xscale PXA310, 624MHz
  - L1 Cache: 32KB + 32KB
  - L2 Cache: 256KB (Disabled)
  - Memory: 128MB
  - Guest OS: Linux-2.6.21

**LMBENCH Micro Benchmark** (latency)

- Higher is better
- **Xen/ARM**
- **L4**

**AIM7 Macro Benchmark**

- S: size(byte)
- P: # of processes

Normalized Performance vs. Number of Tasks

- Native Linux
- Xen/ARM
- L4
Xen ARM Application For Security
Mobile Malware

- Number of mobile malware
  - More than 420 mobile phone viruses (2008)
  - Tens of thousands of infections worldwide

- Concerns about mobile phone security – by market
Access Control

**Definition:** Access control is a system which enables an authority to control area and resources in a given physical facility or computer-based information system [source: Wikipedia]

**Problem with performance isolation**

- **Availability threat:** denial of service (DoS) attack from a compromised domain in a mobile device
  - **CPU overuse:** a greater share of CPU time than initial allocation
  - **Performance degradation:** The Performance of other domains that share the same I/O device with the compromised domain
  - **Battery drain**

---

[Diagram showing access control systems and issues]
Access Control

- Fine-grained I/O access control module in the IDD and coarse-grained access control module in Xen
- Estimation of CPU consumption by each virtual I/O operation using regression analysis
  - Network and storage devices
- I/O access control enforcement based on the policy and regression equations

Target HW spec: XScale 624MHz, 128MB DRAM

Regression Analysis: Network
\[ f_{\text{NET}, \, T_x}(x) = 370.18 + 0.01 \times x \]

Regression Analysis: Storage
\[
\begin{align*}
    f_{\text{MTD}, \text{READ}}(x) &= 250 + 0.24 \times x \\
    f_{\text{MTD}, \text{READOOB}}(x) &= 533 + 0.06 \times x \\
    f_{\text{MTD}, \text{WRITE}}(x) &= 160 + 0.24 \times x \\
    f_{\text{MTD}, \text{WRITEOOB}}(x) &= 583 \\
    f_{\text{MTD}, \text{ERASE}}(x) &= 153.33 + 0.02 \times x \\
    f_{\text{MTD}, \text{ISBADBLOCK}}(x) &= 58 \\
    f_{\text{MTD}, \text{MARKBAD}}(x) &= 60
\end{align*}
\]

x: bytes, f(x): usec
## Test Cases

<table>
<thead>
<tr>
<th>Network I/O Test Cases</th>
<th>Storage I/O Test Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Attack</td>
<td>TcS0</td>
</tr>
<tr>
<td>Under Attack</td>
<td>TcN1</td>
</tr>
<tr>
<td>(No I/O ACM)</td>
<td>TcS1</td>
</tr>
<tr>
<td>Under Attack</td>
<td>TcN2</td>
</tr>
<tr>
<td>(20% I/O ACM Policy)</td>
<td>TcS2</td>
</tr>
<tr>
<td>Under Attack</td>
<td>TcN3</td>
</tr>
<tr>
<td>(10% I/O ACM Policy)</td>
<td>TcS3</td>
</tr>
</tbody>
</table>

**net_atk:** UDP packet flooding (sending out UDP packets with the size of 44,160 bytes every 1msec)

**mtd_atk:** excessive NAND READ operations (scanning every directory in the filesystem and reading file contents)
Access Control

- **Throughput increase and power consumption decrease even under malware attack**

**Effectiveness**

**Throughput: Network**

![Bar chart showing throughput for Network test cases (TcN0, TcN1, TcN2, TcN3) under No attack and Under attack conditions.](chart)

**Throughput: Storage**

![Bar chart showing throughput for Storage test cases (TcS0, TcS1, TcS2, TcS3) under No attack and Under attack conditions.](chart)

**Power Consumption**

![Bar chart showing power consumption for Network and Storage test cases (TcN0/TcS0, TcN1/TcS1, TcN2/TcS2, TcN3/TcS3) under No attack and Under attack conditions.](chart)
Xen ARM Application For Real-time
Xen ARM: Pre-emption

### Status of real-time support
- The jitter of timer interrupt latency by the hypervisor is bounded within 10% compared with native real-time OS.

### Technical Issue
- DI and NP sections should be minimized.

#### Latency caused by interrupt-disabled (DI) section
- Delay

#### Latency caused by non-preemptible (NP) section
- Delay

#### Urgent Interrupt
- Wake up next VM

- Hypervisor should support RT Domain via priority-based scheduling, VMM pre-emption and so on.

### Guest Domain
- Non RT Domain (Running)

### Xen-ARM
- Send IRQ to Real-time Domain (As Non RT Domain has credits, RT Domain is not scheduled. Furthermore Xen cannot tell difference between Non RT Domain and RT Domain.)

### Scheduling

### Preemption
- Voluntary CPU release
- Non RT domain
Real-time Performance

- **Evaluation Environment**

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H/W (Tegra250)</td>
<td>CPU</td>
</tr>
<tr>
<td></td>
<td>RAM</td>
</tr>
<tr>
<td>S/W</td>
<td>Hypervisor</td>
</tr>
<tr>
<td>Guest OS (DOM0)</td>
<td></td>
</tr>
<tr>
<td>Guest OS (DOM1)</td>
<td>uC/OS-II</td>
</tr>
</tbody>
</table>

- **Cyclic test benchmark repeats**
  1. RT task sleeps for 10ms
  2. Timer interrupt will occur after 10ms
  3. Timer interrupt wakes up the RT domain (uC/OS-II)
  4. uC/OS-II preempts Xen-ARM
  5. RT task is scheduled
  6. RT task logs timestamp

**CDF of Responsiveness (Periodic Timer Interrupt : 10ms)**

- **Response Overhead (3us)**

**Native (uC/OS-II)**

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>Avg</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>9995</td>
<td>9996</td>
<td>810169</td>
<td>10000</td>
</tr>
</tbody>
</table>

**Xen-ARM (uC/OS-II)**

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>Avg</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>9996</td>
<td>9999</td>
<td>327119</td>
<td>10001</td>
</tr>
</tbody>
</table>

**Unit: usec**