Server Virtualization
Bare-Metal Hypervisor

poor device support / sharing

Hypervisor / VMM

Hardware
### Motivation

Poor device performance

Host user space

### Architecture

Host OS Kernel

- kernel module
- emulated devices

Hypervisor / VMM

Hardware
Non-Virtualization
User Space SDK

- no standard apps
- less secure

- custom user space API for isolated apps

OS Kernel

Hardware
## Key Challenges

- **device diversity**
  - Power
  - Cell Radio
  - h.264 accel.
  - camera(s)
  - Touchscreen
  - WiFi
  - pmem
  - speakers
  - Buttons
  - GPU
  - Binder IPC
  - Accelerometer
  - microphone
  - headset
  - GPS
  - Framebuffer
  - Compass
  - RTC / Alarms

- **mobile usage model**
  - graphics-accelerated UI
**Cells**

**Key Observation**

small: one app at a time

large: lots of windows/apps
screen real-estate is limited, and mobile phone users are accustomed to interacting with *one thing* at time
**Cells**

*Usage Model*

foreground / background
Cells
Complete Virtualization

- multiple, isolated virtual phones (VPs) on a single mobile device
- 100% device support in each VP
  - unique phone numbers - single SIM!
  - accelerated 3D graphics!
• less than 2% overhead in runtime tests
• imperceptible switch time among VPs
• each VP sees / uses all devices
• user can run any unmodified apps
• foreground VP switches like an app
Single Kernel: Multiple VPs

isolated collection of processes

virtualize at OS interface

Linux Kernel
**Single Kernel: Device Support**

All VPs access the same device simultaneously:

- Power
- Touchscreen
- WiFi
- hw codec
- pmem
- camera(s)
- speakers
- microphone
- Buttons
- GPU
- Binder IPC
- Accelerometer
- RTC / Alarms
- headset
- GPS
- Framebuffer
- Compass

Kernel

Input
Sensors
Audio/Video
Android...
Device Namespaces

safely, correctly multiplex access to devices
Complete, Efficient, Transparent

Device namespaces

Foreground / background

Cells

Linux Kernel

Cell Radio

Framebuffer

GPU

Power

Input

Sensors

Audio/Video

RTC / Alarms

Android...

Motivation

Overview

Architecture

Graphics

Cell Radio

Evaluation

Demo
efficient basic graphics virtualization

hardware accelerated graphics

proprietary / closed interface
**Approach 1: Single Assignment**

- **screen memory**
- **Framebuffer**
- **GPU**

Virtual addresses

Physical addresses
Approach 2: Emulated Hardware

screen memory

emulated framebuffer

virtual state
mux_fb presents identical device interface to all VPs using device namespaces.

Swap virt addr mappings: point to different phys addr.

Cells: Device Namespaces

Motivation

Overview

Architecture

Graphics

Cell Radio

Evaluation

Demo

mux_fb presents
identical device
interface to all VPs
using device namespaces.

Swap virt addr mappings:
point to different phys addr.

Screen memory

Framebuffer

RAM

Virtual addresses
Physical addresses
Accelerated Graphics

VP: just a set of processes!

OpenGL context

Framebuffer

MMU

GPU

process isolation

graphics virtual addresses

physical addresses

screen memory

Motivation  Overview  Architecture  Graphics  Cell Radio  Evaluation  Demo
**Device Namespace + Graphics Context**

- **VP 1**
  - Foreground
  - OpenGL context

- **VP 2**
  - Background
  - OpenGL context

- **VP 3**
  - Background
  - OpenGL context

**Screen Memory**

- **MMU**
  - Graphics virtual addresses

- **GPU**
  - Physical addresses

**Motivation**

**Overview**

**Graphics**

**Cell Radio**

**Evaluation**

**Demo**
VoIP?

- RilD
- Vendor RIL
- VoIP
- VoIP
- Drivers
- Linux Kernel
- Baseband: GSM / CDMA
DUAL-SIM?

- RilD
  - Vendor RIL
    - Drivers
      - GSM/CDMA
  - RilD
    - Vendor RIL
      - Drivers
        - GSM / CDMA

Linux Kernel
**Motivation**

Proprietary hardware/software requires a well-defined interface.

**Overview**

Vendor API requires a well-defined interface.

**Architecture**

Cell Radio

**Drivers**

Linux Kernel

Baseband: GSM / CDMA

**CellD**

Cell: User-Level Namespace Proxy
Experimental Results

Setup

- Nexus S
- five virtual phones
- overhead vs. stock (Android 2.3)
**Experimental Results**

**Setup**

- CPU *(Linpack)*
- graphics *(Neocore)*
- storage *(Quadrant)*
- web browsing *(Sun Spider)*
- networking *(Custom WiFi Test)*
**EXPERIMENTAL RESULTS**

**Runtime Overhead**

Negligible Overhead In 3D Measurements!

- **Linpack**
- **NeoCore**
- **Quadrant I/O**
- **Sun Spider**
- **Network**

![Bar chart showing runtime overhead across different systems and configurations](chart.png)
**Cells**

**Complete, Efficient, Transparent**

**Mobile Virtualization**

- device namespaces
  - safely and efficiently share devices
- foreground / background
  - designed specifically for mobile devices
- implemented on Android
- less than 2% overhead on Nexus S