Восьмая независимая
научно-практическая конференция
«Разработка ПО 2012»
1 - 2 ноября, Москва

Технология контейнерной виртуализации для платформы Android

Евгений Баталов, Александр Карташов, **Кирилл Кринкин**

Санкт-Петербургский Академический университет РАН

Рынок мобильных устройств

- В 2011:
  - Пользователей сотовой связи -- 5,981 млн
  - Подписчиков мобильного интернет -- 1,186 млн
  - Продано мобильных телефонов – 1,775 млн
  - Продано смартфонов -- 491 млн
    - (Прогноз 2012 – 686 млн)

- Нежелательное ПО:
  - Официально зарегистрировано 2,500 mobile malware (2010, Bullguard)
  - Двукратный рост числа mobile malware в 2011 по отношению к 2010 (IBM X-force)

http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats

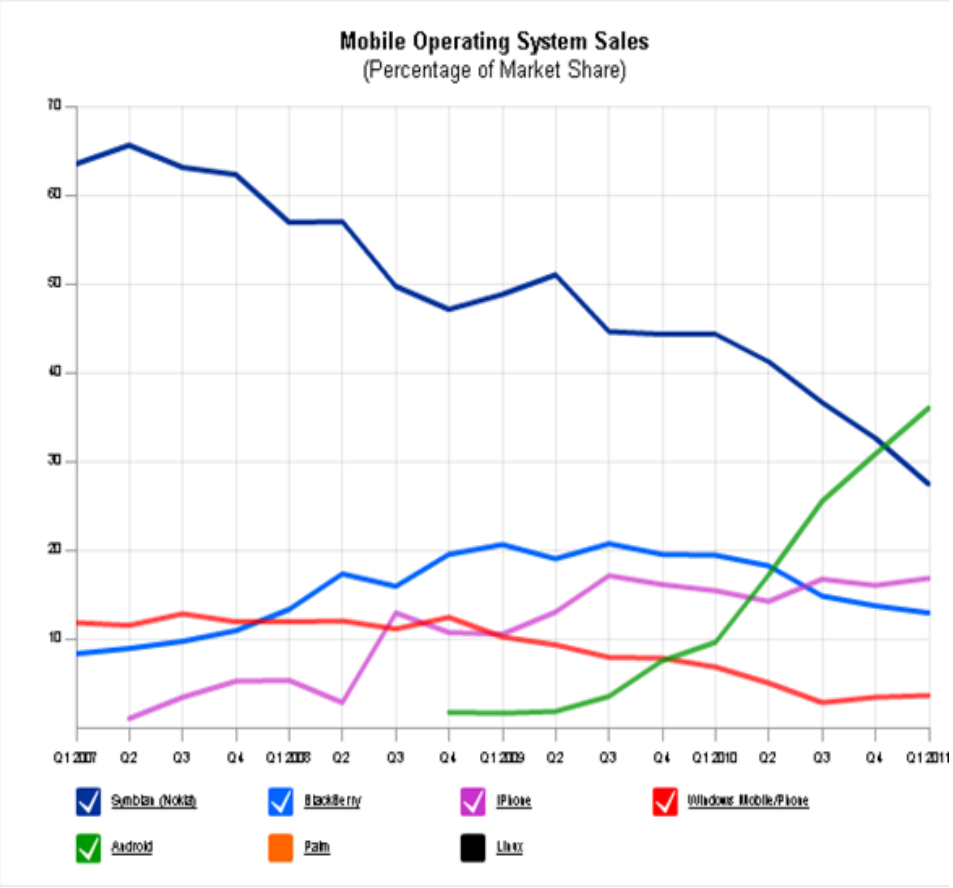
Ожидаемые технологии (Gartner)

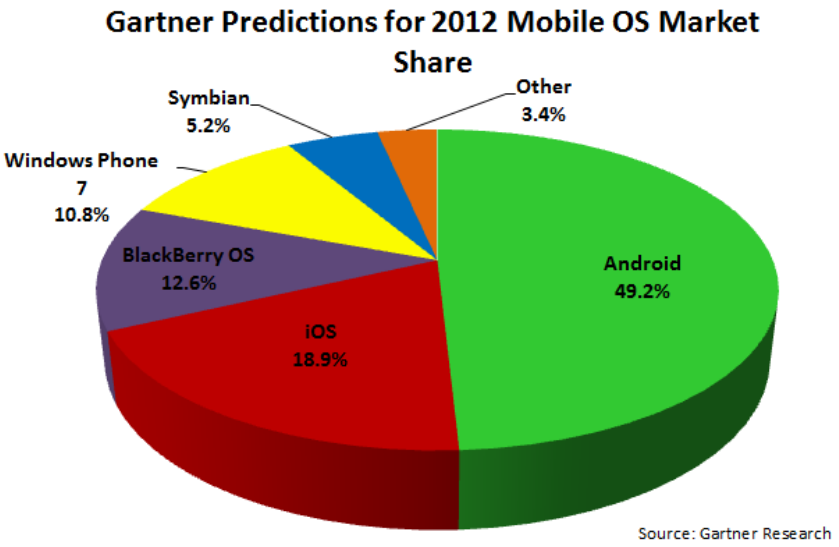
http://www.gartner.com/it/page.jsp?id=2124315

Задачи "мобильной виртуализации"

- Защита личных данных;
- Изоляция подозрительных программ;
- Ограничение доступа приложений к системным сервисам (SMS, WiFi, телефония,…);
- Защита корпоративных приложений на смартфонах сотрудников;
- Эффективное использование аппаратных средств;
- Создание окружения для разработки и отладки мобильных приложений.

Почему Android?





- Open source
- Linux based

Gartner Research

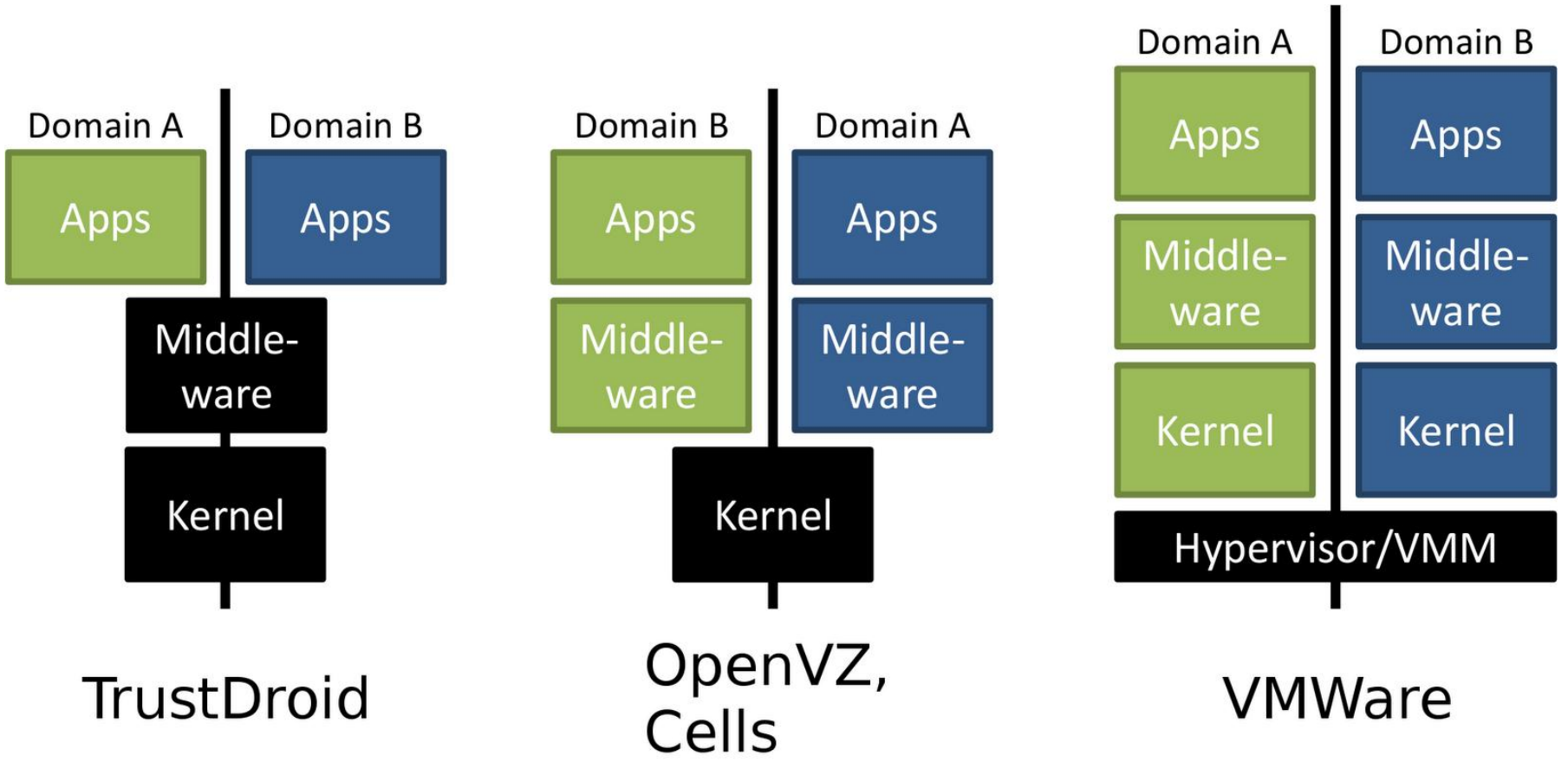
AndroidVM. Цели и задачи

- Независимое исполнение нескольких Android на одном устройстве;
- Защита взаимного влияния приложений и системных служб, принадлежащих разным Android;
- Управление доступом к физическим и виртуальным устройствам;
- Совместное использование устройств;
- Эффективное использование системных ресурсов (память, CPU, GPU, аккумулятор,…)

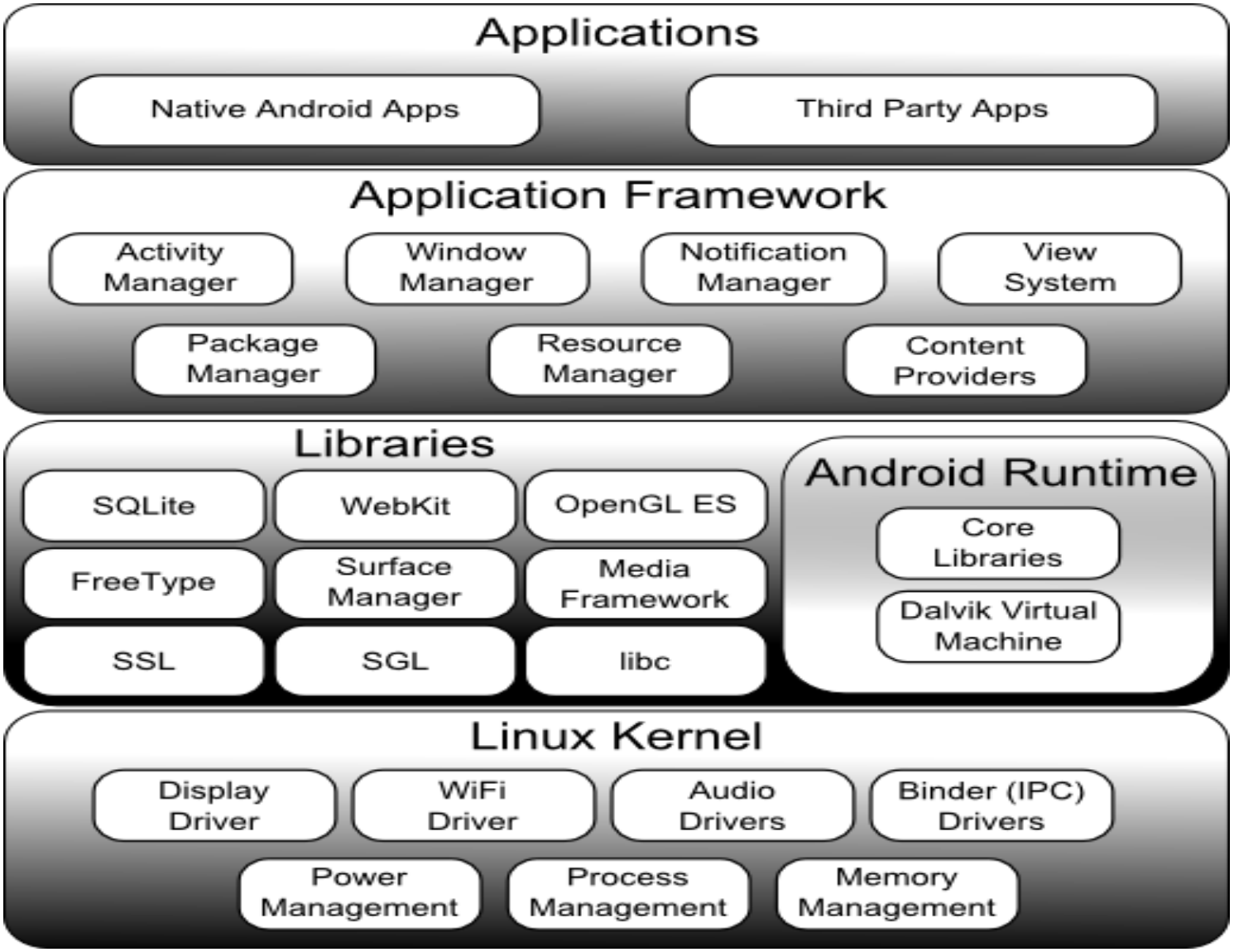
Основные варианты использования

- Запуск нескольких независимых контейнеров с Android на одном устройстве;
- Интерактивная работа пользователя с приложениями в одном (активном) Android-контейнере;
- Фоновая работа приложений в неактивных (фоновых) контейнерах;
- Возможность смены активного контейнера.

Изоляция программ на мобильных платформах

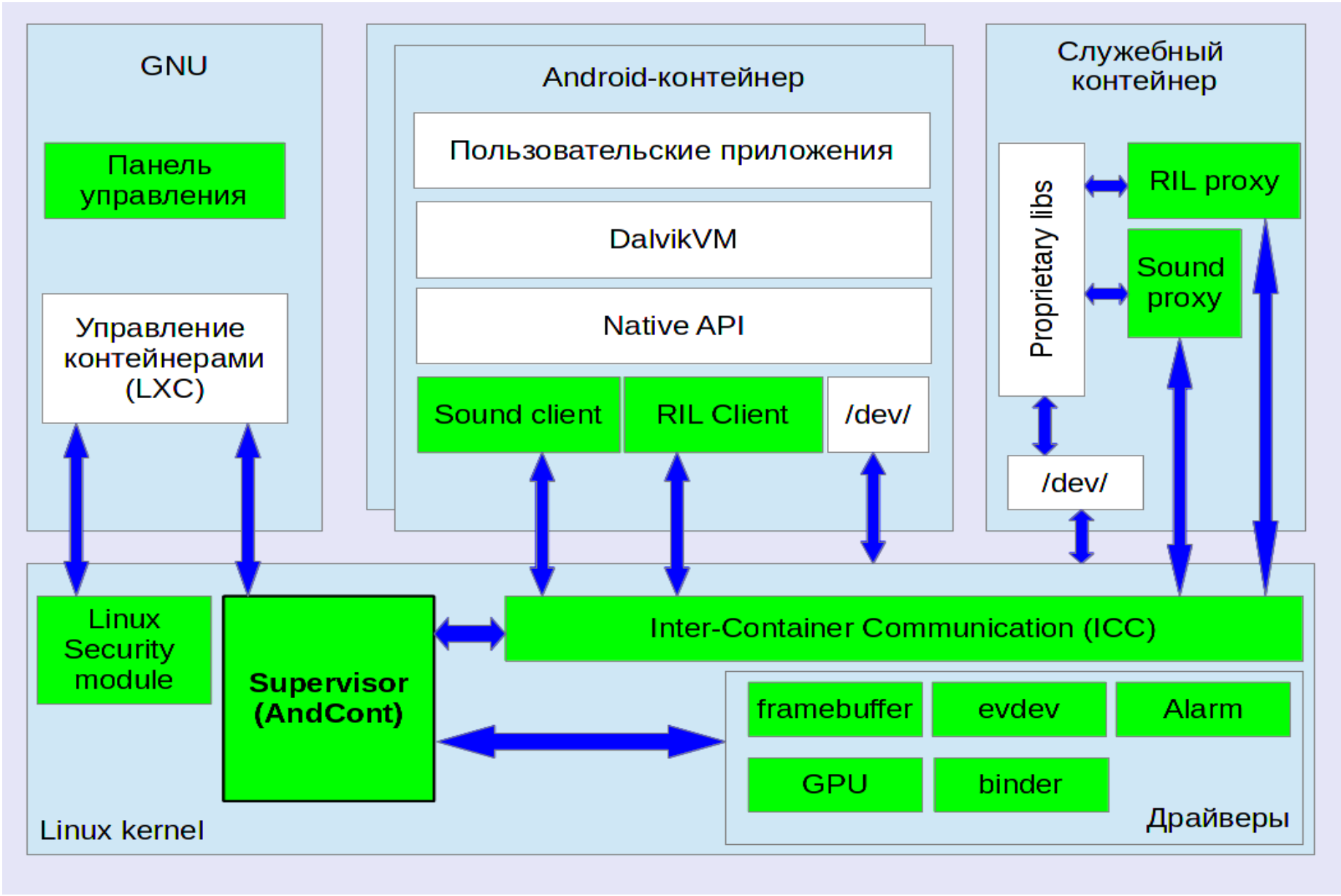


Архитектура Android



http://www.techotopia.com

Архитектура решения



Доработанные компоненты

- Ядро
  - Supervisor (AndCont)
  - Механизм межконтейнерного взаимодействия (ICC)
  - LXC
- Периферия
  - Мультиплексор RIL
  - Мультиплексор звука
  - Framebuffer
  - GPU
- Системные сервисы
  - Binder,
  - Alarm
- Панель управления

Доработки LXC

- Запрет инициации перезагрузки виртуальной машиной Dalvik.
- Уведомления супервизора уровня ядра о запуске нового контейнера перед запуском процесса `init`.
- Расширен синтаксис конфигурационных файлов LXC: добавлена возможность описания заблокированных путей `sysfs` и перенаправления блочных устройств.
- Устранена утечка файловых дескрипторов из демона `adbd`, который используется для доступа к смартфону через USB.

Механизм межконтейнерного взаимодействия (ICC)

Задача: обеспечить возможность передачи сообщений между контейнерами

Ограничения:

- сокеты не работают, т. к. привязаны к сетевому пространству имен;
- именованные каналы требуют агент в корневом пользовательском окружении – неэффективно.

Решение:

Собственный механизм и протокол IPC на базе Netlink

Контроль доступа к `sysfs`

Задача: обеспечить возможность запрета управления физическими устройствами через интерфейс `sysfs`.

Решение

- Операции блокировки `sysfs` в фреймворке Linux Security Modules.
- Конфигурационные параметры конфигурационном файле контейнера со списком запрещенных путей в `sysfs`.

Виртуализация блочных устройств

Задача: предоставить контейнерам персональный раздел на MMC накопителе.

Решение:

- Образ раздела MMC-карты разместить в устройстве loop.
- Перехват запроса блочного устройства (`bdget`), подстановка ссылки на подготовленный loop.

Виртуализация системных сервисов

Задача: обеспечить непротиворечивый доступ к системным сервисам, создать у Android иллюзию монопольного использования.

Примеры сервисов:

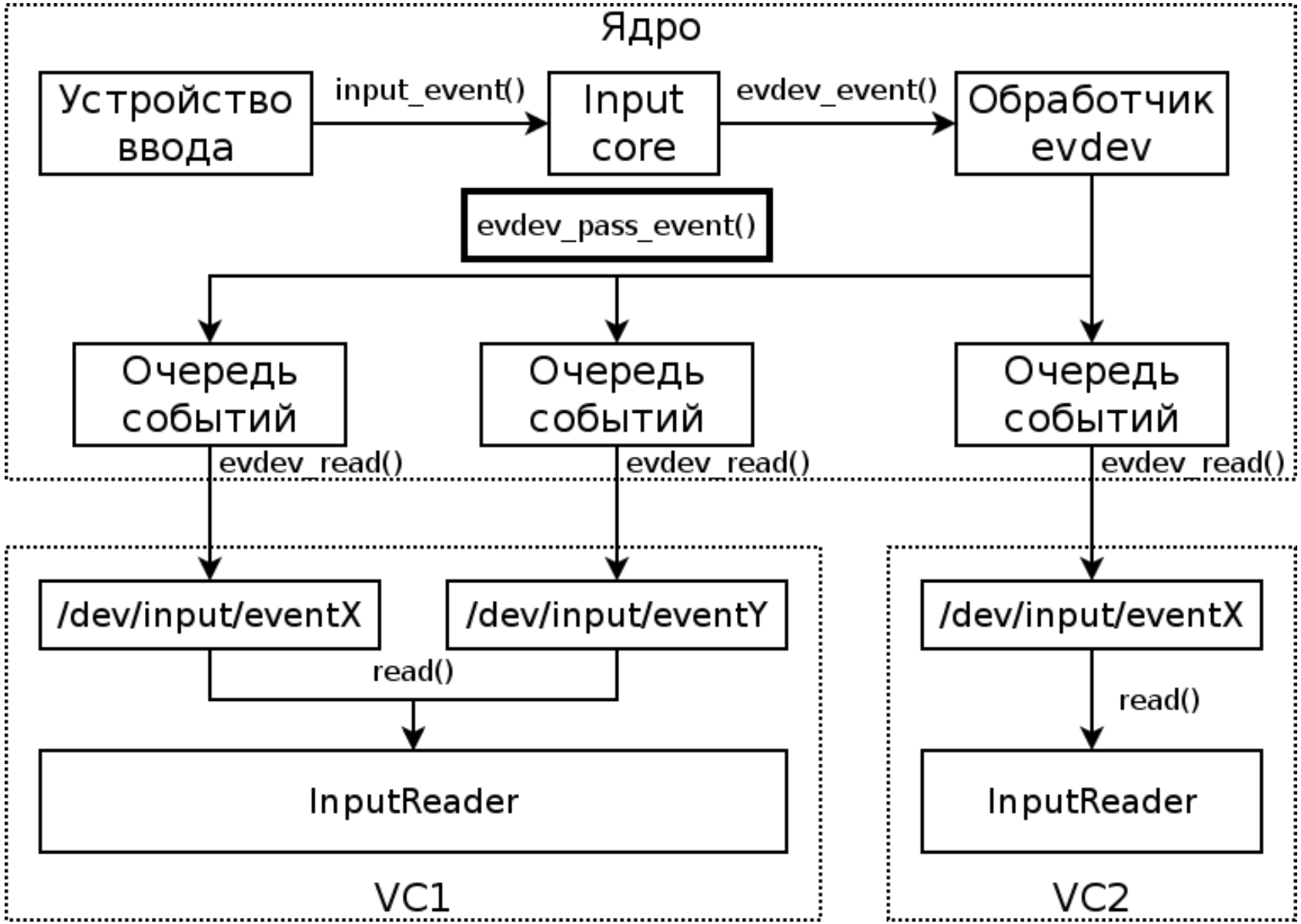
- Binder — система межпроцессного обмена;
- Alarm — интерфейс к часам реального времени.

Решение:

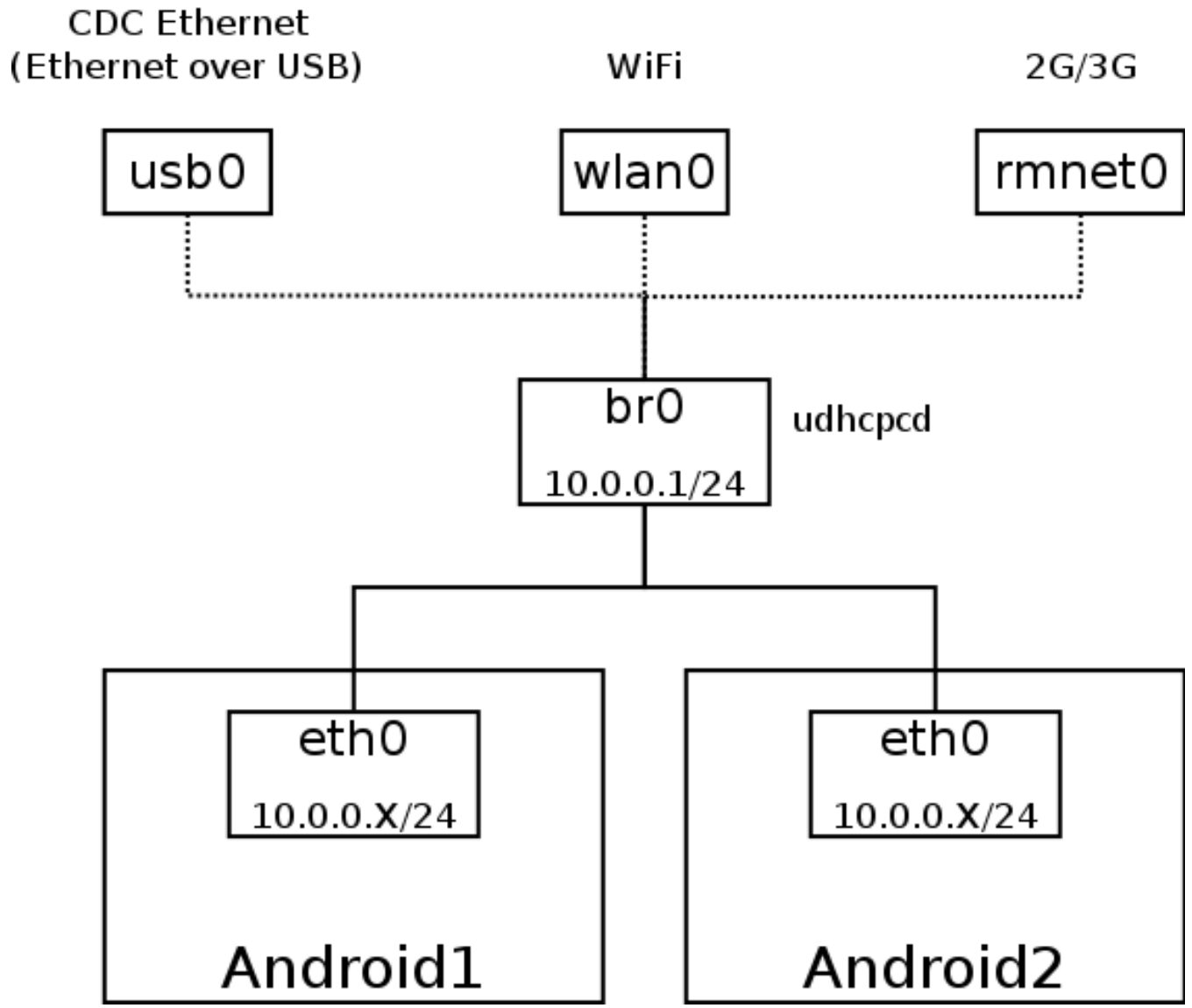
Виртуализация устройств:

- поддержка состояния устройства для каждого контейнера
- механизм мультиплексирования

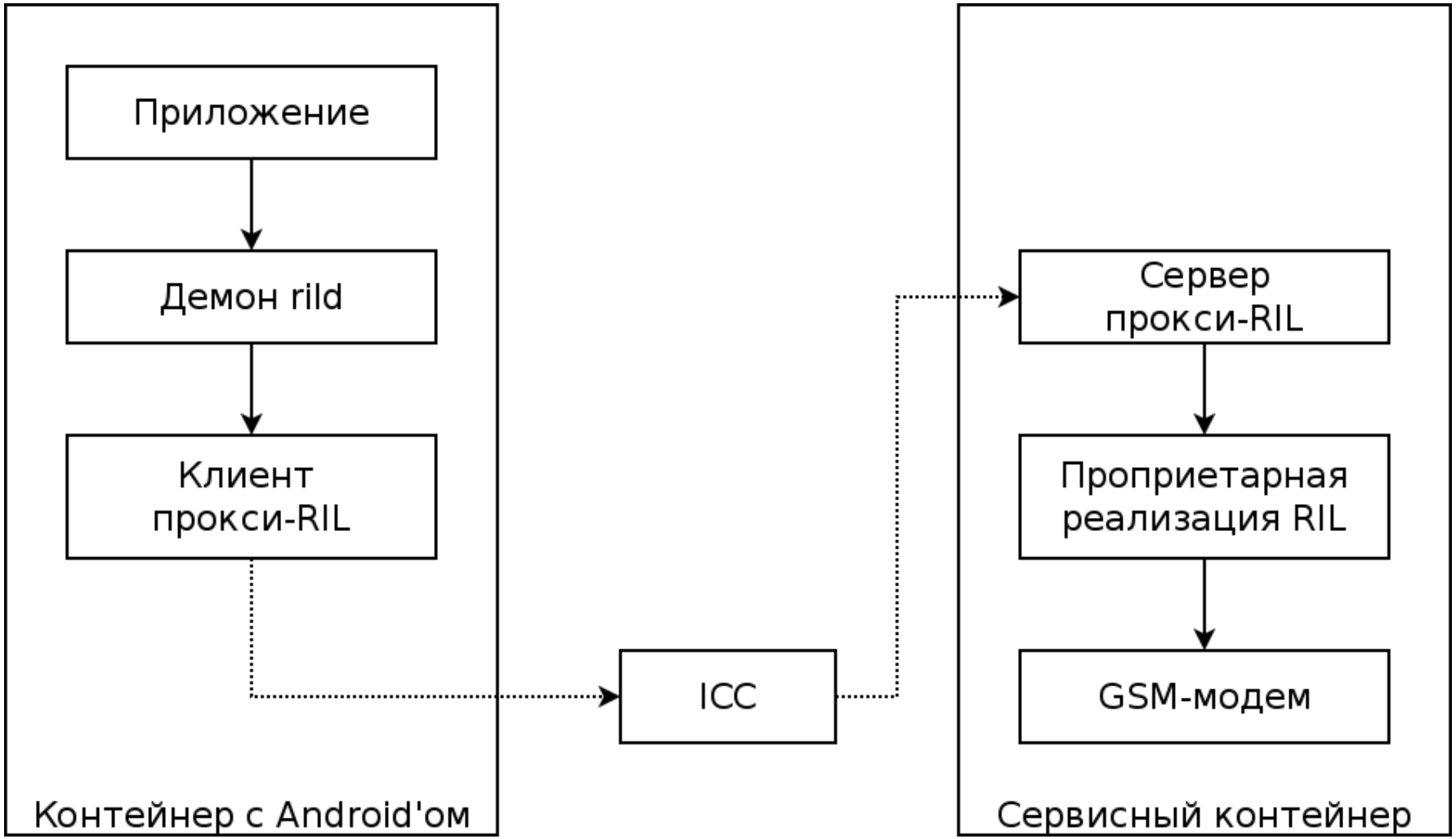
Виртуализация подсистемы ввода



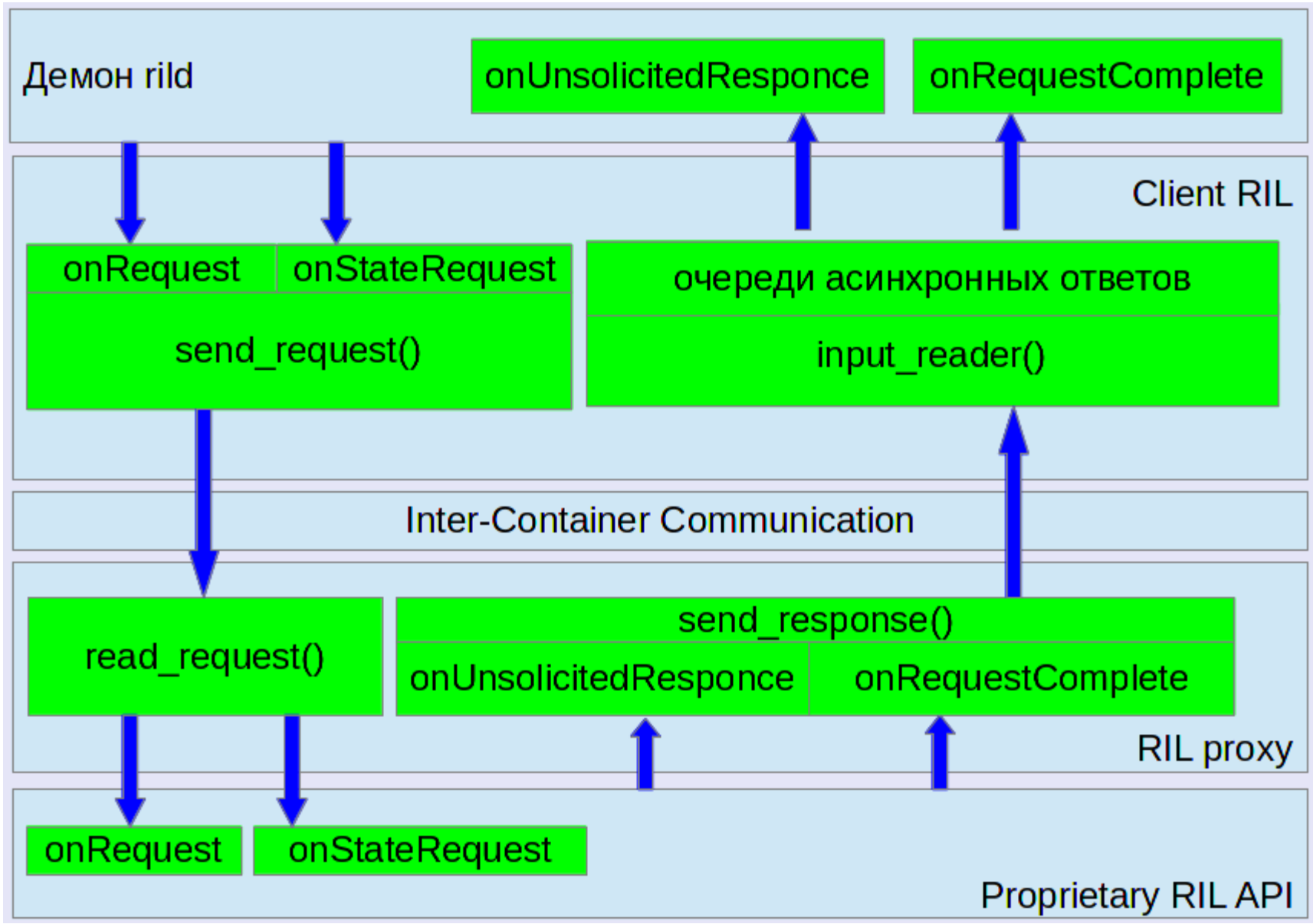
Виртуализация сети



Телефония (Radio-Interface Layer)



Мультиплексирование стека RIL



Маршрутизация запросов RIL

- Б*езусловная передача запроса* проприетарной библиотеке для всех запросов, не изменяющих состояние соединения и GSM-модема.
- *Передача запроса от активного контейнера проприетарной библиотеке* для запросов, связанных с отправкой SMS и совершением звонков.
- *Безусловное блокирование* — для всех неизвестных запросов;

Маршрутизация уведомлений

- *Маршрутизация в активный контейнер* — входящие SMS и звонки.
- *Маршрутизация во все контейнеры* — информационных уведомлений (например, уведомлений об изменении уровня сигнала сотовой сети)

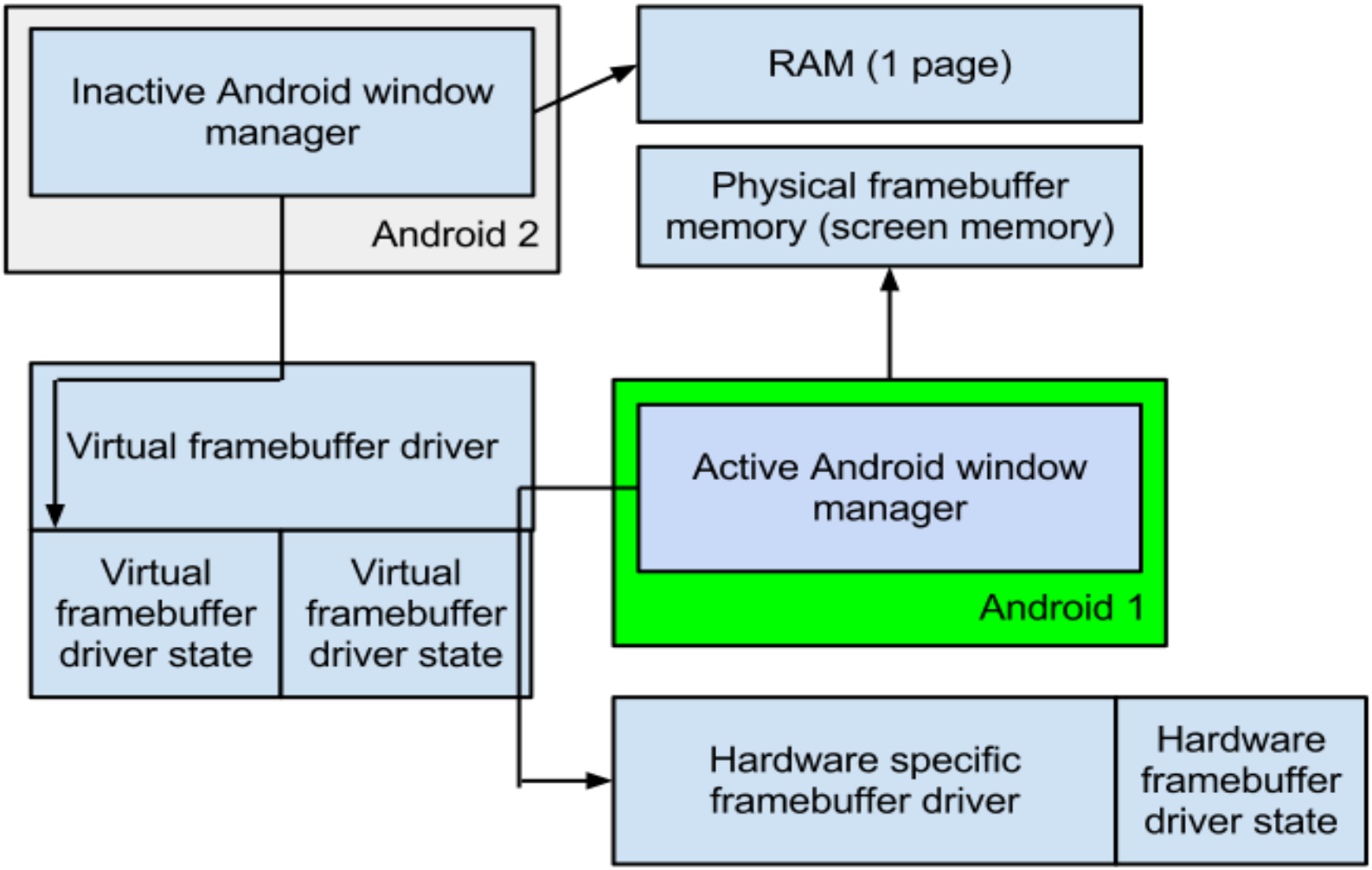
Виртуализация графики

Задача: Отображать экране только активный Android.

Решение:

- Изменить схему отображения памяти видеоконтроллера
- Отображать на память физического экрана только видео-память активного контейнера
- Использовать MMU для реализации схемы отображения видеопамяти

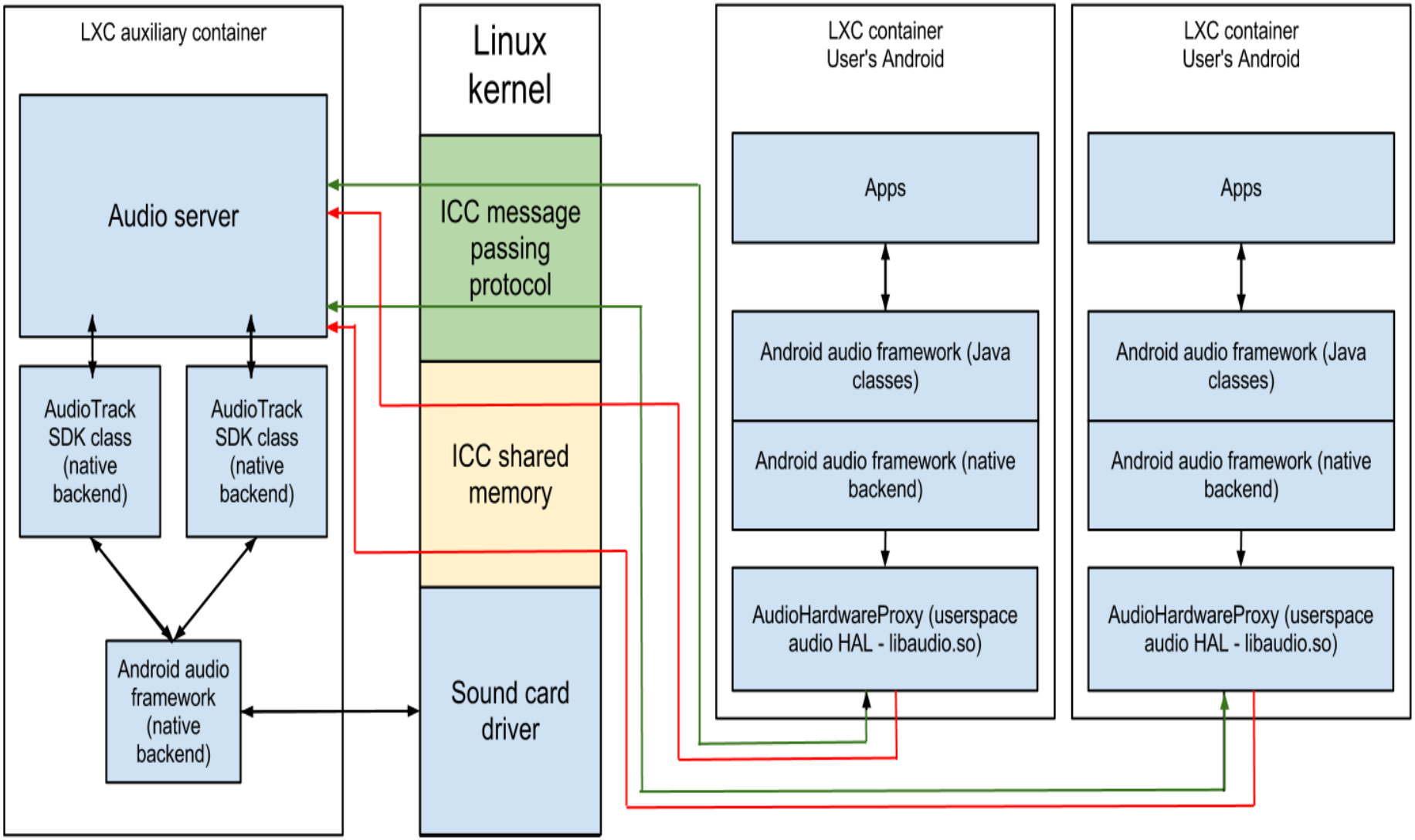
Отображение видеопамяти



Компоненты Android audio framework

- Java API для работы со звуком
- AudioFlinger (C++)
- Программный микшер
- Effects (возможно DSP)
- Android audio HAL

Архитектура виртуализации звука



Приоритезация аудио потоков

- Прокси сервер получает от каждого контейнера звуковой поток, точку назначения, режим.

- Приоритезация точек назначения звукового потока:
  - Headset: 10
  - Small speaker: 5
  - Big speaker: 0

Интеграция Audio и телефонии

1. Звонок поступает в контейнер
2. Контейнер вызывает у прокси клиента AudioHardwareInterface::setMode(IN_CALL)
3. Прокси клиент уведомляет об этом сервер
4. Сервер инициирует эмуляцию воспроизведения звука контейнеров
5. Сервер вызывает у оригинального HAL AudioHardwareInterface::setMode(IN_CALL)
6. Оригинальный HAL ответственен за интеграцию аудио и телефонии

Управления питанием

- Framebuffer early suspend - уведомление userspace о состоянии экрана (on/off)
- Без виртуализации fbearlysuspend следует непосредственно за событием нажатия кнопки power (on/off)
- При fbearlysuspend(off) Android выключает экран
- Многие приложения останавливаются после нажатия power.

Поддерживаемое системное ПО

Ядро Linux:

- 2.6.35 — смартфоны
- 3.0.30 — эмулятор.

Userspace:

- Android 2.3 (CyanogenMod 7) — смартфоны,
- Android 4 (официальный) — эмулятор.

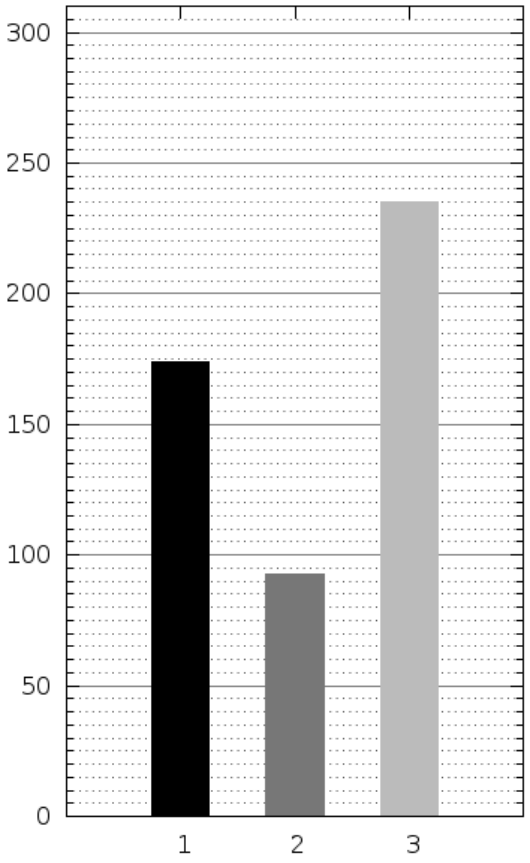
Опорные сценарии для тестирования

1. Простой.
2. Проигрывание музыки с выключенной подсветкой экрана.
3. Интерактивная игра с одновременным проигрывнием музыки.

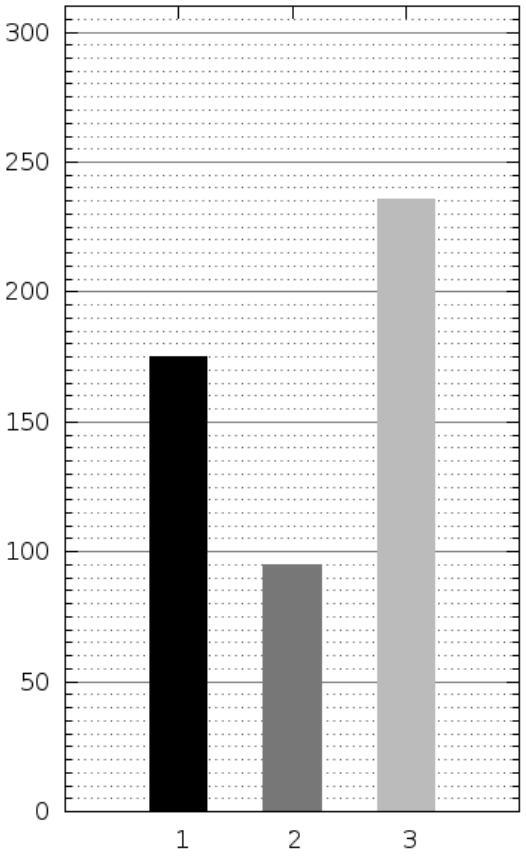
Тестируемые конфигурации

1. Оригинальное окружение CyanogenMod7 для Samsung Galaxy S II.
2. Один контейнер с CyanogenMod 7.
3. Два контейнера с CyanogenMod 7: в одном контейнере проигрывалась музыка, в другом запущена игра.

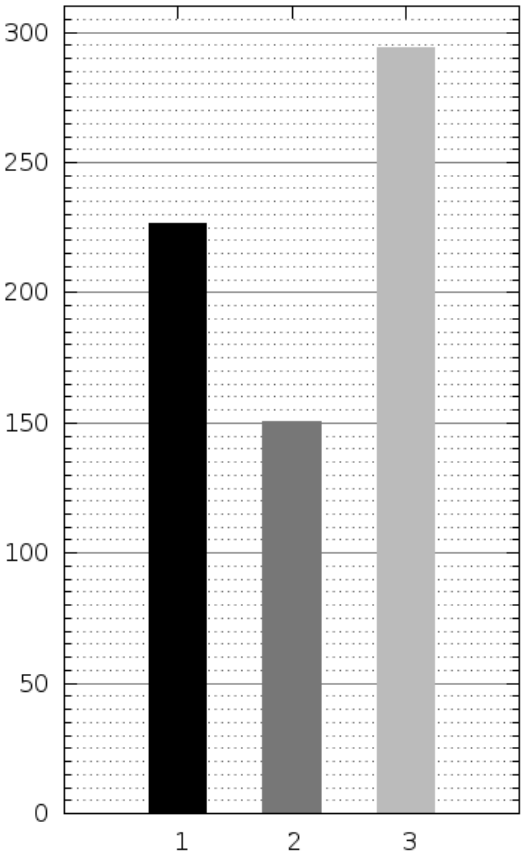
Потребление памяти



Сценарий 1

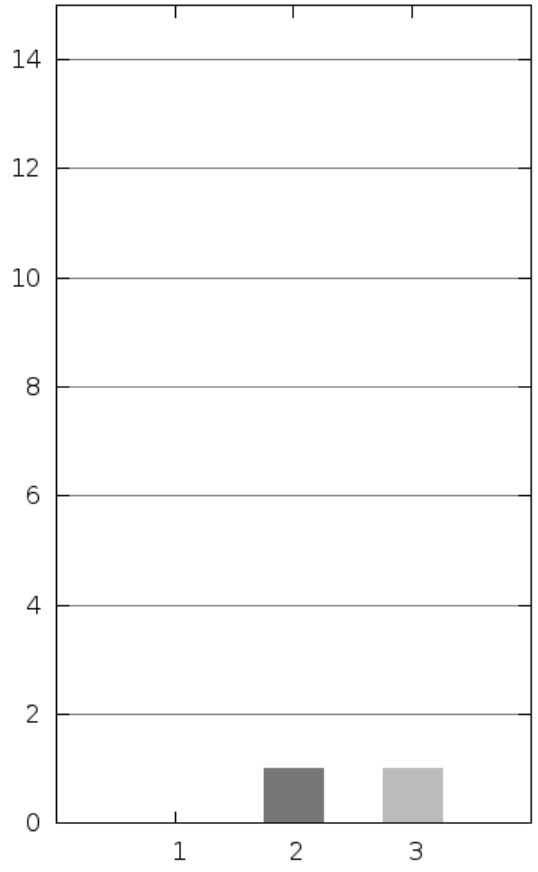


Сценарий 2

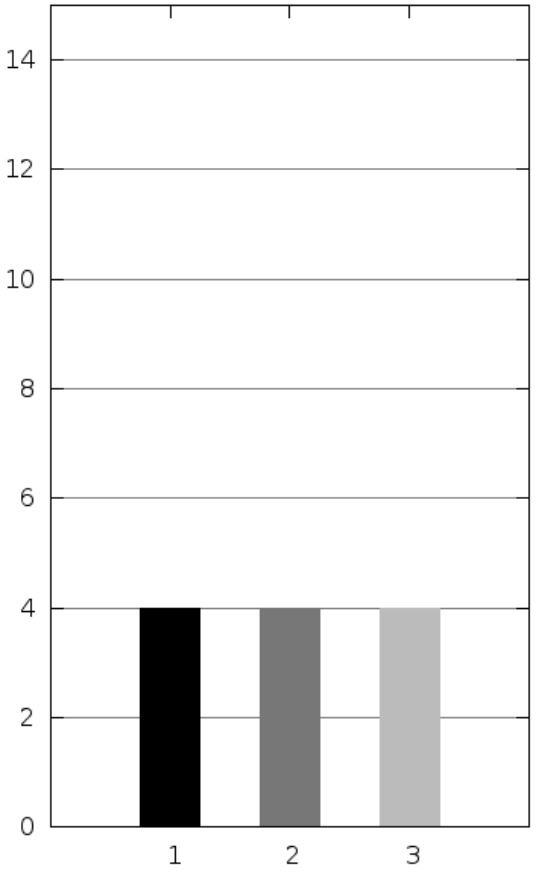


Сценарий 3

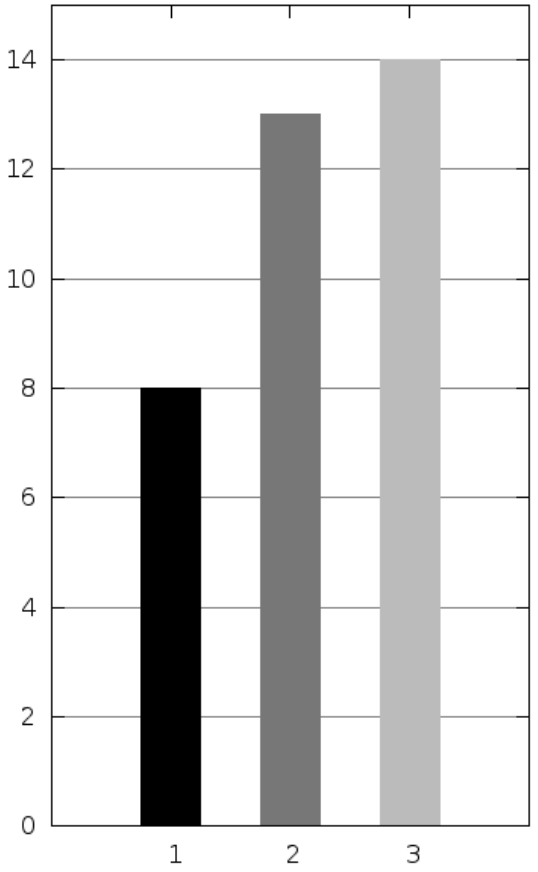
Потребление аккумулятора



Сценарий 1



Сценарий 2



Сценарий 3

Направления развития проекта

- Запрет доступа к устройствам ввода
- Автоматическая генерация начальной конфигурации контейнера
- Управление Android Low Memory Killer
- Совместное использование разделов MMC-карты
- Оптимизация использования памяти для системных библиотек и Dalvik
- Поддержка нескольких абонентских номеров в RIL
- Маршрутизация и совместное использование GPS, Bluetooth, камеры, микрофона

Вопросы

Демо

- http://www.youtube.com/watch?v=0_PsQb4WnT8&feature=youtu.be
- http://16918.selcdn.ru/androidvm/androidvm-demo.mkv
- http://16918.selcdn.ru/androidvm/demo.m4v