NDK
native API crunch

# Android SDK

## Lecture 3

Android Native Development Kit

11 March 2014

NDK
native API crunch

Linux Kernel

Binder

Android Framework

Managers

Keywords

# Outline

# Linux Kernel

- "Androidized" kernel
- Hundreds of patches over the standard kernel
- Device-specific functionality, fixes, enhancements
- Many features get into the mainline kernel
- "Androidisms"
  - Wakelocks
  - Low-Memory Killer
  - Binder
  - Anonymous Shared Memory
  - Alarm
  - Logger

# Wakelocks

- The Android kernel goes to sleep as often as possible
- Sometimes you want to keep the system from going to sleep
  - Input from the user, critical operations
- Wakelocks keep the system awake
- A wakelock must be obtained by the application when it needs to stay awake
  - Apps communicate with the Power Manager Service when they require a wakelock
  - Device drivers call in-kernel wakelock primitives
- Included in mainline

# Low-Memory Killer

- Linux OOM killer
- Prevents the activation of the OOM killer (system unlikely to run out of memory)
- Kills processes with components unused for a long time
- Based on OOM adjustments mechanism
  - Different OOM kill priorities for different processes
- The userspace may control OOM killing policies
- Policies applied at startup by init
- Modified and enforced by Activity Manager

# Low-Memory Killer

- Levels assigned to processes based on their components
  - Levels from -17 to 15 (high -> killed)
- Threshold (MinFree) for each type of process
  - Foreground_app - application in foreground
  - Visible_app - visible but not in foreground
  - Secondary_server - service
  - Hidden_app - hidden, needed by a running app
  - Content_provider - provide data
  - Empty_app - app not active
- Starts killing when the threshold is reached

# Anonymous Shared Memory (ashmem)

- IPC mechanism
- SysV IPC can lead to resource leakage in the kernel (vulnerability)
- Similar to POSIX SHM, differences:
  - Uses reference counting to destroy the memory regions
  - Shrink mapped regions when the system needs memory
  - To shrink a region it must be unpinned
- First process creates region, uses Binder to share descriptor with other processes
- System services rely on ashmem
  - Surface Flinger, Audio Flinger
- Driver included in the staging tree

# Alarm

- Uses the RTC and HRT fuctionalities
- The *setitimer()* does not work when the system is suspended (HRT)
  - The application receives the signal when the device wakes up
- Using RTC, the alarm will be fired even if the system is suspended
  - RTC hardware device
- Uses HRT by default
- When the system is about to suspend, it uses RTC

# Alarm

- */dev/alarm* character device, *ioctl()*
- SystemClock, AlarmManager class rely on the driver
  - SystemClock - obtain and set time
  - AlarmManager - provide alarms to apps
- The driver and AlarmManager use WakeLocks
  - The app that receives the alarm runs before the system is suspended again
- Included in the staging tree

# Logger

- Uses kernel buffers for logging data
  - Circular buffers in RAM
  - No task switch, no writing in files (compared to syslog)
  - Avoiding write operations in files is critical on Android devices
- Each buffer - separate entry in /dev/log (Events, System, Radio, Main)
  - Logcat displays the Main buffer by default

# Logger

- Log and EventLog classes - public API
- Developers use Log
- EventLog used by the system components
  - Diagnostic events
- Slog - system use (AOSP)

# Logger

- Through liblog library
  - Logging from java classes
  - Used by logcat
  - Formatting and filtering
- Log message
  - Priority, tag and data for each event
  - Priority: verbose, debug, info, warn, error
  - Tag: identifies the component that generated the message
- Staging tree

# Outline

# Binder History

- RPC mechanism
- Initially in BeOS (then bought by Palm)
- OpenBinder project
- OpenBinder developers working in Android team
- Android Binder does not derive from OpenBinder
  - Clean re-write of the same functionality
- OpenBinder documentation for understanding the mechanism
- Binder driver in the staging tree from kernel 3.3

# Binder

- Remote object invocation
  - Remote services as objects
  - Interface definition and reference to it
- Cornerstone of Android architecture
  - Apps talk to System Server
  - Apps talk to other service components
- Developers don't use the Binder directly
- Use interfaces and stubs generated with the *aidl* tool

# Binder Driver

- Part of the Binder implemented in a kernel driver
- Character driver
- */dev/binder*
- Uses *ioctl()* calls
- Transmit parcels of data (serialized) between entities

# Outline

# Android Framework

- On top of the native userspace
- android.* packages, System Services, Android Runtime
- Code in */frameworks* directory in AOSP
- Key building blocks: Service Manager, Dalvik, Zygote

# System Services

- Form an object-oriented OS on top of Linux
- System Server
  - All components run in the *system_server* process
  - Many Java-based services/managers, 2 C-based services
  - Power Manager, Activity Manager, Location Manager, etc.
  - Surface Flinger, Sensor Service (C/C++)
- Media Server
  - *mediaserver* process
  - C/C++ code
  - Audio Flinger, Media Player Service, Camera Service

# Service Manager

- Performs system service handle lookups
- The Yellow pages book of all services
- A service must be registered to the Service Manager to be available
- Started by *init* before any other service
- Opens /dev/binder and becomes the Context Manager of the Binder
- Binder ID 0 = "magic object" = Service Manager

# Service Manager

- System Server registers every service with the Service Manager
- Any app that wants to talk to a system service:
  - Asks the Service Manager for a handle
  - *getSystemService()*
  - Invokes the methods of the service using the handle
- Not used by an app to access its own service
- Used by the dumpsys utility to obtain the status of the system services

# Dalvik

- Dalvik virtual machine
- Java VM optimized for mobile architectures
  - Lower memory footprint
- Works with .dex files instead of .jar files
- Incompatible with Java bytecode
- Register based, not stack based
- 16 bit instructions

# Dalvik

- Includes Just-in-Time (JIT) compiler
  - ARM, x86, MIPS
  - Translates bytecode into binary machine instructions
  - Run directly on the CPU, not one instruction at a time by the VM
  - The conversion is stored and used next time the application runs
  - Apps run much faster

# Zygote

- Daemon used to launch apps
- Listens to connections on its socket for requests to start apps
  - */dev/socket/zygote*
- When it gets a request, it forks itself and launches the app
- Preloads (in RAM) all Java classes and resources needed by an app
- Copy-on-write (COW)
- Classes and resources are not modified, so all apps use them from Zygote (only one copy in RAM)
- The System Server is started explicitly by Zygote
- The PPID of all apps is the PID of Zygote

# Outline

# Activity Manager

- One of the most important services in the System Server
- Handles application lifecycle
- Broadcasts intents
- Starting new components (activities, services)
- Fetching Content Providers
- Responsible with the Application Not Responding (ANR) messages
- Involved in
  - Permission checks
  - OOM adjustments for the Low-Memory Killer
  - Task management

# Activity Manager

- Starts the Launcher (with Intent.CATEGORY_HOME)
- When an app is started from Launcher
  - Launcher's *onClick()* callback is called
  - Launcher calls the startActivity() from ActivityManager (through Binder)
  - ActivityManager calls startViaZigote() method
  - Opens socket to Zygote and asks to start the activity
- *am* command for invoking the functionality of the ActivityManager
- isUserAMonkey()

# Package Manager

- Manages the .apk files in the systems
- API for installing, uninstalling, upgrading .apk files
- *system_server* and *installd* processes
- Ensures that the JIT versions of the dex bytecode is available before the app is started
- Resolves intents
- *pm* command for invoking the functionality of the PackageManager

# Power Manager

- Control the power state of the device
- Handles WakeLocks
- Includes the WakeLock class
  - *aquire()*, *release()*
- Apps request WakeLocks from PowerManager
- All calls to the Power Management (kernel) goes though PowerManager
- Can force device to go to sleep
- Set the brightness of the backlights

# Outline

NDK
native API crunch

- WakeLocks
- Low-Memory killer
- Binder
- Ashmem
- Alarm
- Logger
- System Server
- Service Manager
- Dalvik
- Zygote
- Activity Manager
- Package Manager