# Android Kernel Modifications

Karthik Dantu and Steve Ko

# Today: Android Kernel Extensions

- Resources:
  - AOSP
  - Linux code
  - “Embedded Android”
  - http://elinux.org
  - http://source.android.com
  - http://developer.android.com

# Kernel Changes

- ashmem
- binder
- Logger
- Paranoid Networking
- pmem
- Viking killer
- wakelocks
- drivers, ports to various platforms

# Wakelocks

- A mechanism to prevent the system from entering into the suspend or other low-power states
- Telling the system that there's something important going on, i.e., the system shouldn't go to low-power states.
- Can be used from the user space as well as the kernel space

# Android App Wakelock Types

| Flag Value | CPU | Screen | Keyboard |
|---|---|---|---|
| PARTIAL_WAKE_LOCK | On | Off | Off |
| SCREEN_DIM_WAKE_LOCK | On | Dim | Off |
| SCREEN_BRIGHT_WAKE_LOCK | On | Bright | Off |
| FULL_WAKE_LOCK | On | Bright | Bright |

# User Space Wakelock

- Any user space process can lock/unlock by writing to
  - /sys/power/wake_lock
  - /sys/power/wake_unlock
- PowerManager does this

```
PowerManager pm = (PowerManager) getSystemService(Context.POWER_SERVICE);
 PowerManager.WakeLock wl = pm.newWakeLock(PowerManager.SCREEN_DIM_WAKE_LOCK, "My Tag");
 wl.acquire();
   ..screen will stay on during this section..
 wl.release();
```

# Kernel Space Wakelock

#include <linux/wakelock.h>

void wake_lock_init(struct wake_lock *lock, int type, const char *name)

void wake_lock_destroy(struct wake_lock *lock)

void wake_lock(struct wake_lock *lock)

void wake_unlock(struct wake_lock *lock)

```
struct state {
        struct wakelock wakelock;
}

init() {
        wake_lock_init(&state->wakelock, WAKE_LOCK_SUSPEND, "wakelockname");
}

Before freeing the memory, wake_lock_destroy must be called:

uninit() {
        wake_lock_destroy(&state->wakelock);
}
```

# Acceptance to Mainstream Kernel

- Code mostly developed behind closed doors
- Shipped first before getting into the mainline kernel
- Much duplication with an existing API (pm_qos)
- No way to recover if a user process exits while holding a wakelock
- Etc.

# Memory Management

- Android wants to keep apps around as much as possible
  - Multi-tasking, fast context switch
  - Memory size is limited, so we can't keep everything around
  - No swap space
- Enable faster IPC

# ashmem

- Shared memory allocator
- Like POSIX SHM, but with simpler file-based API
- Reference counting for better memory management
- Pin/upin for better memory usage
- libcutils (system/core/include/cutils/ashmem.h)

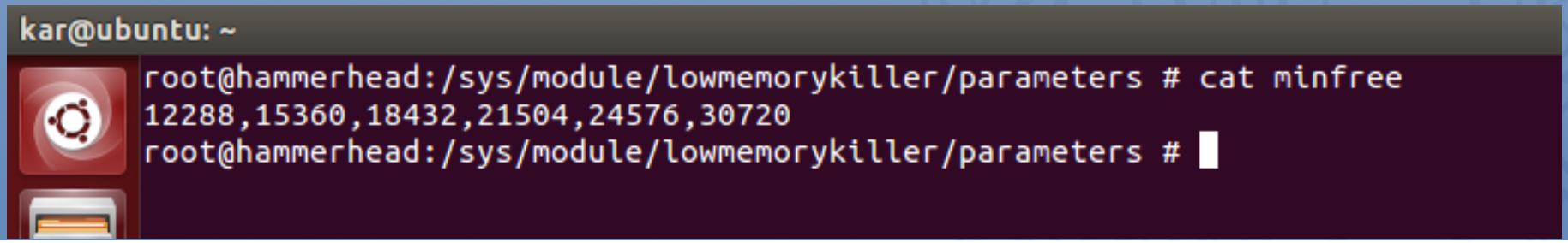
```
fd = ashmem_create_region("my_shm_region", size);
if(fd < 0)
    return -1;
data = mmap(NULL, size, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
if(data == MAP_FAILED)
    goto out;
```

# pmem

- Allow kernel/user-space to share large (1-16+MB) physically contiguous memory
- Similar to ashmem
  - No reference counting
  - pmem = physical memory; ashmem = virtual memory
- Used mainly for kernel drivers (dsp, gpu)
- G1 → hardware 2D engine uses it for scaling, rotation, color conversion
- drivers/misc/pmem.c

# Viking Killer

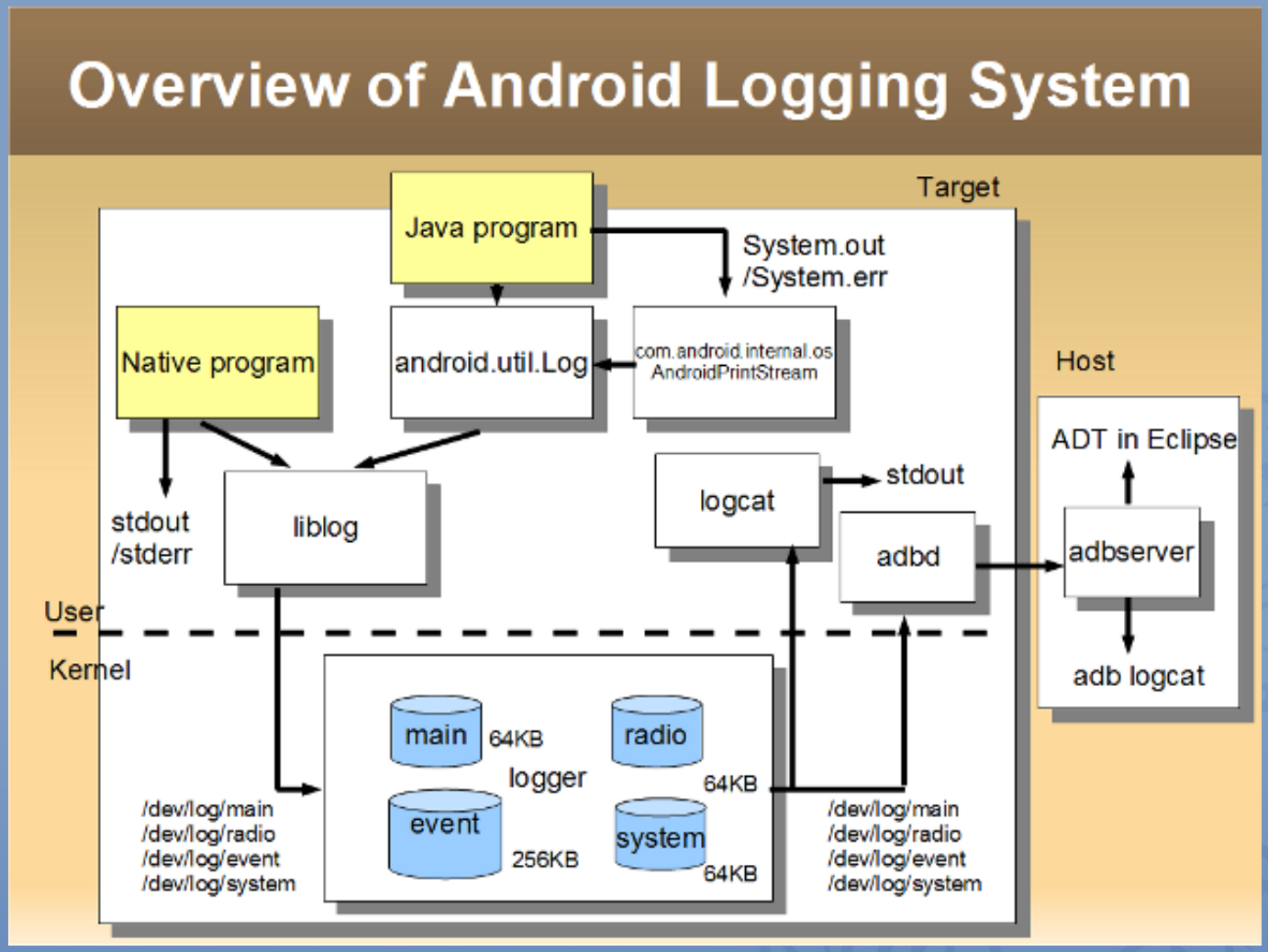
- Custom implementation of OOM killer from Linux
- Allows user-space input into OOM kill policy
- Six categories: foreground app, visible app, secondary server, hidden app, content provider, empty app
- Uses LRU policy
- Memory limit in terms of pages
- drivers/misc/lowmemorykiller.c
- security/lowmem.c



# Android Logging System

- Android has a separate logging system different from Linux's (e.g., dmesg & /proc/kmsg).
- A typical user's point of view
  - android.util.Log API
  - Logcat
- Kernel-side
  - /drivers/staging/android/logger{.c, .h}

# Architecture




http://elinux.org/Android_Logging_System (image by Tetsuyuki Kobabayshi, of Kyoto Microcomputer Co.)

# Log Buffers

- Main: main app log
  - android.util.Log
- Event: system event information
  - android.util.EventLog
- System: system components
  - android.util.Slog
- Radio: radio and phone-related events
- /dev/log

# Paranoid Networking

- Another potentially interesting change
- GID-based network restrictions

| #define | GID | Capability |
|---|---|---|
| AID_NET_BT_ADMIN | 3001 | Can create an RFCOMM, SCO, or L2CAPP Bluetooth socket |
| AID_NET_BT | 3002 | Can create a Bluetooth socket |
| AID_INET | 3003 | Can create IPv4 or IPv6 socket |
| AID_NET_RAW | 3004 | Can create raw sockets |
| AID_NET_ADMIN | 3005 | Allow CAP_NET_ADMIN permissions for process |

# Summary

- Wakelock is a mechanism to tell the kernel that it shouldn’t go to sleep because there’s some interesting stuff going on
- Low memory killer manages app processes memory usage
- Logger has a high-speed mechanism to log different types of events