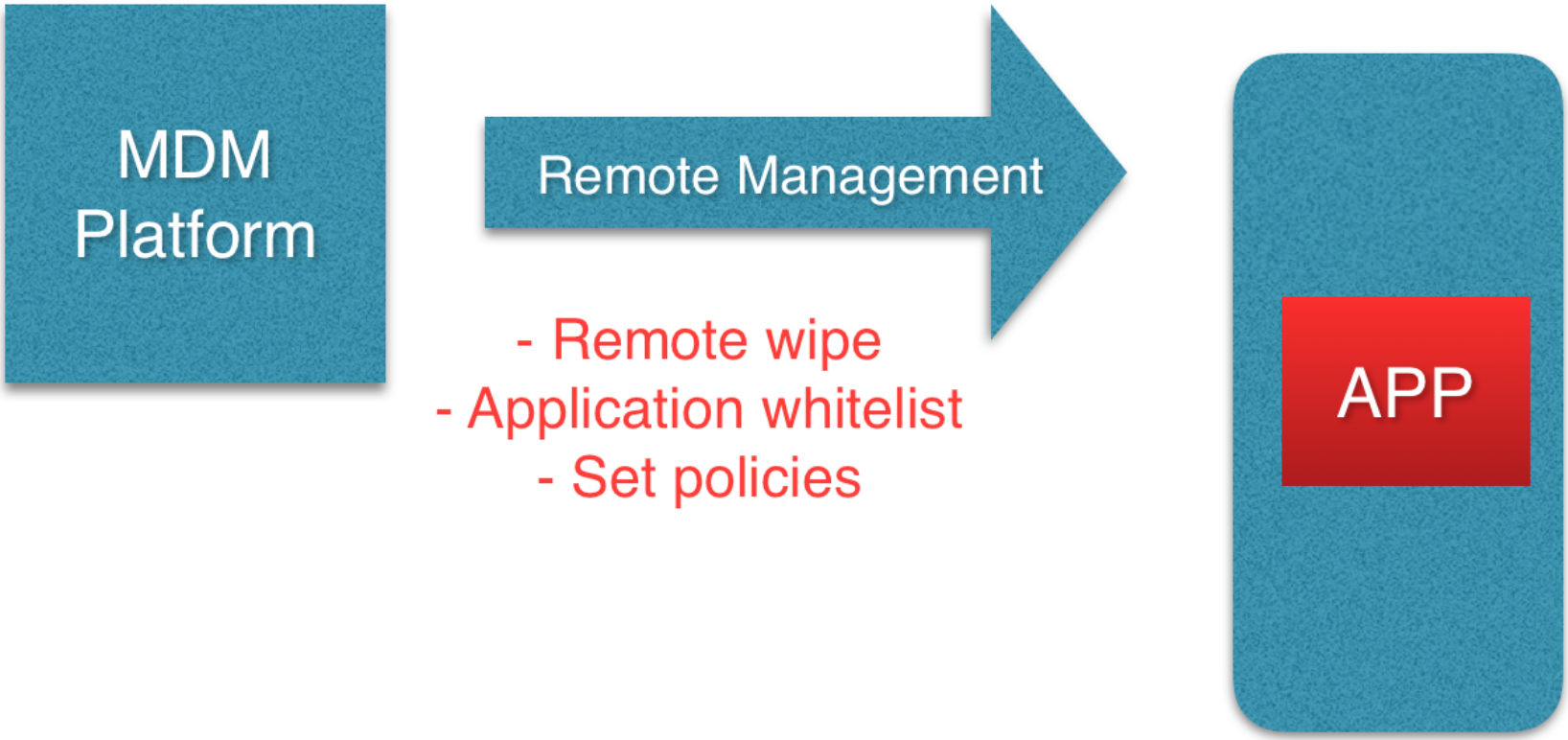
Security Analysis of Android for Work

Research Project #1

Tom Curran & Ruben de Vries

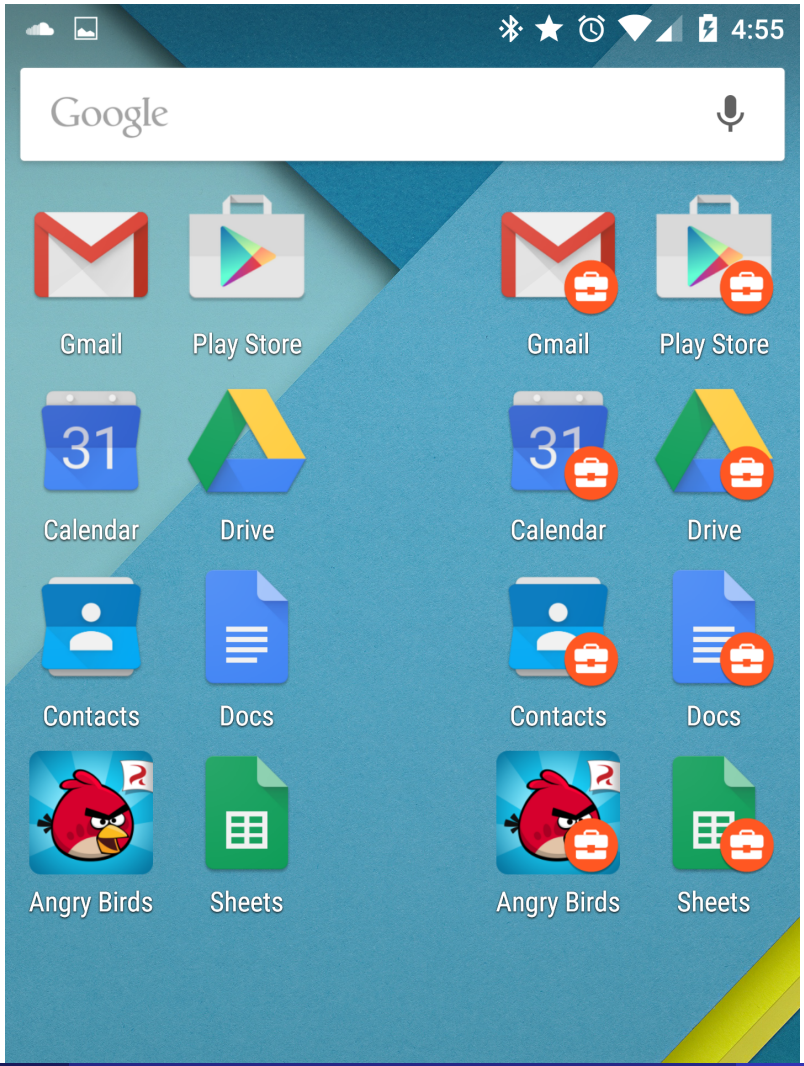
RP1 project presentation, 2016

What is Android for Work



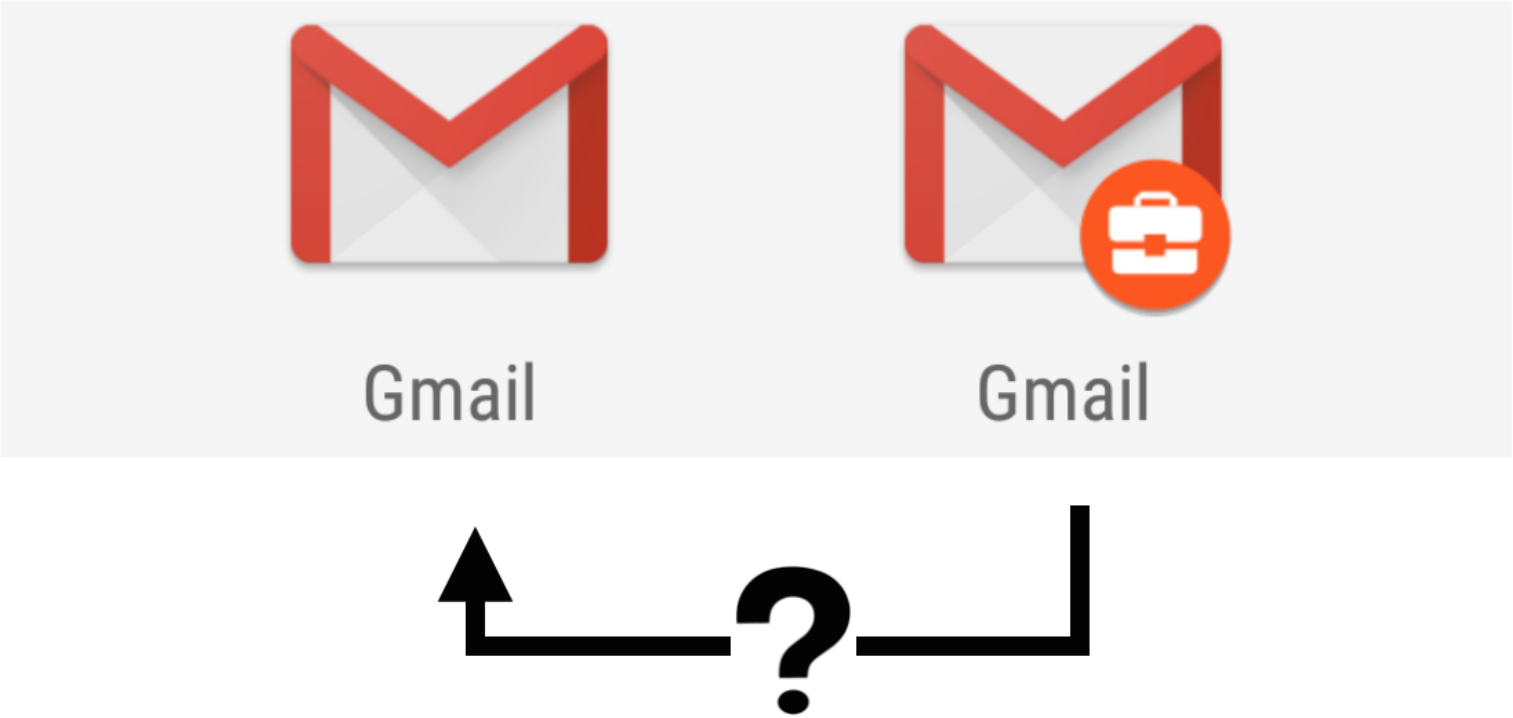
Why is it interesting?

- Data separation achieved using separate user profiles
- Profiles run concurrently



Research Question

Is it possible to read data from the work profile using a process started by the personal profile?



Research Question; narrowed down

- Is it possible to read data from a managed profile from the user profile using the binder?
- How does Android for Work handle encryption of data?

Findings

- Data can be read via the Binder
- Data is encrypted when device is switched off, but not once it is running.

Encryption; Demo

[...] Once a device is encrypted, all user-created data is automatically encrypted before committing it to disk and all reads automatically decrypt data before returning it to the calling process.

- Android for Work Security White Paper

Root?

- Root exploits uncovered in the past
  - Towel Root, affecting up to KitKat 4.4.2 (2014)
  - Stagefright 2.0, affects up to Lollipop 5.1 (2015)
- Rooting Marshmallow 6.0+ Harder but possible
  - SELinux
  - Exploits in Linux kernel e.g. CVE-2016-0728 (2016)
  - *Fuzzing Android System Services by Binder, Blackhat 2015*
- Once you have root, lie about having it
  - *All Your Root Checks Are Belong to Us, Blackhat 2015*

Android Version Distribution

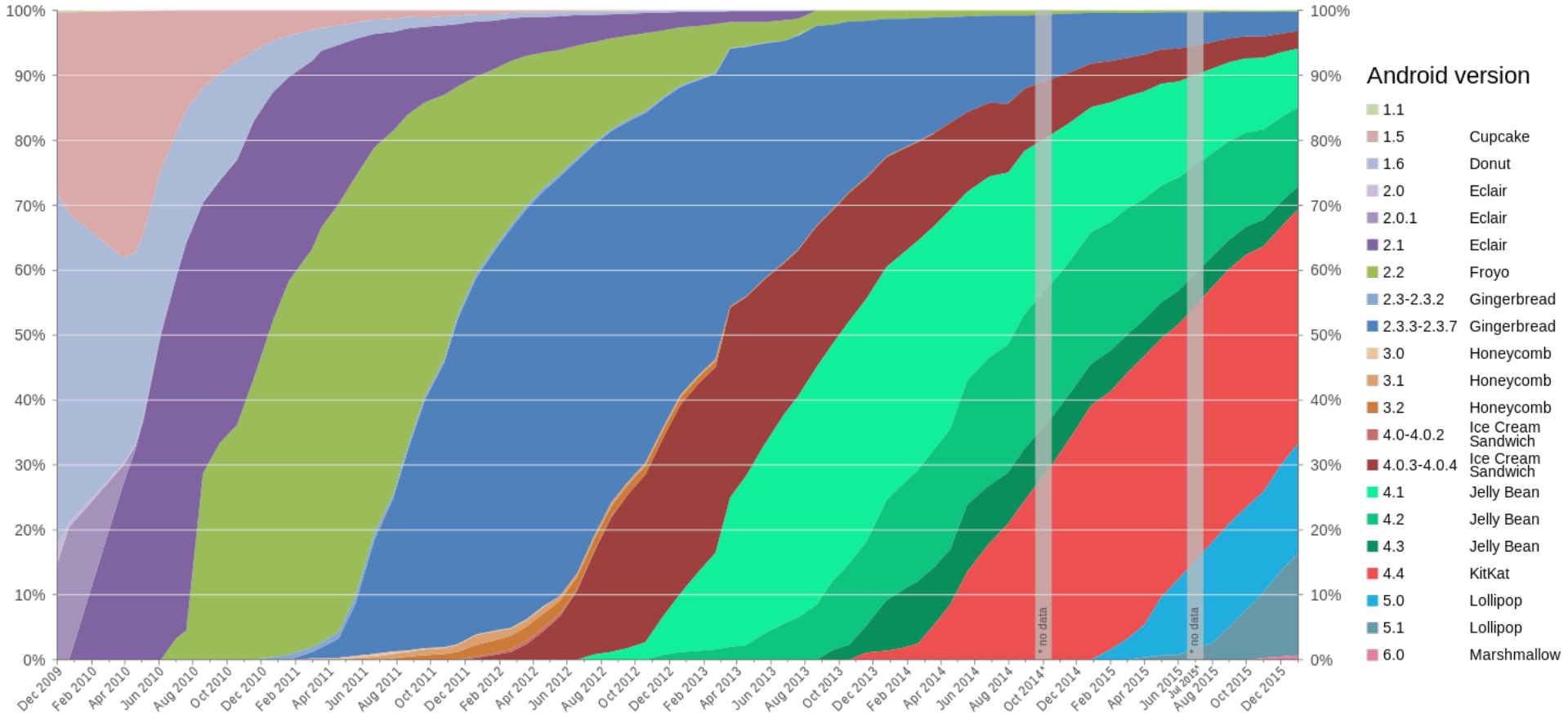
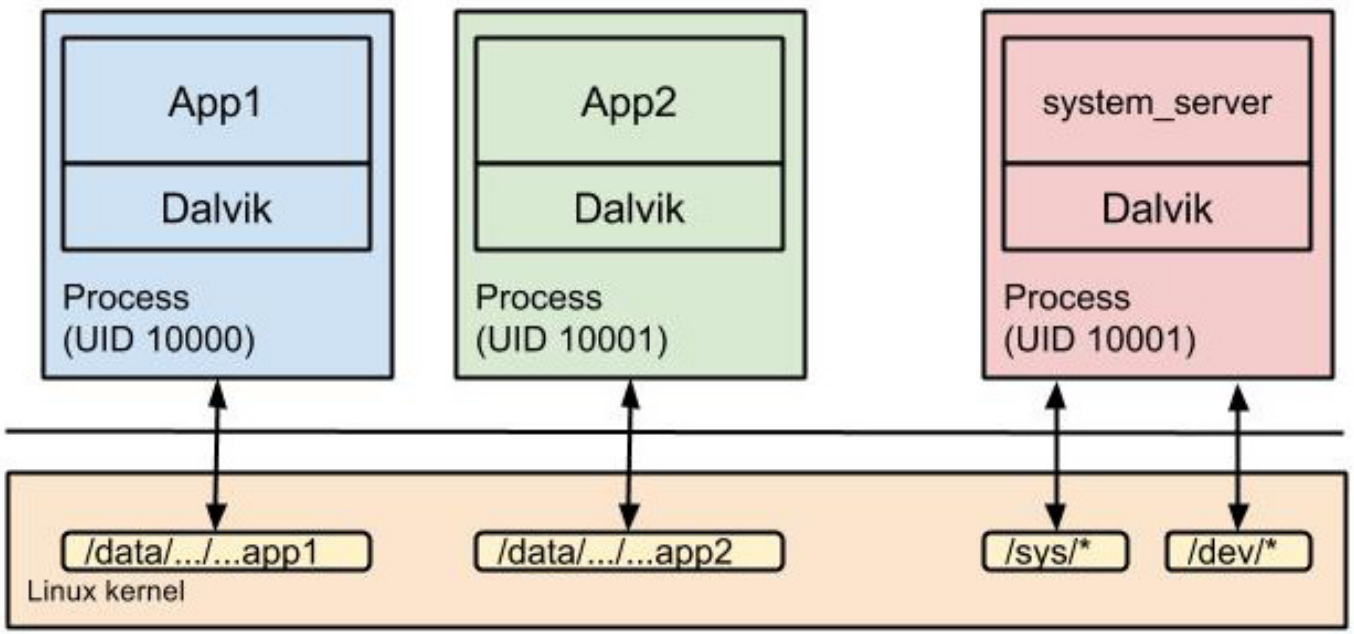
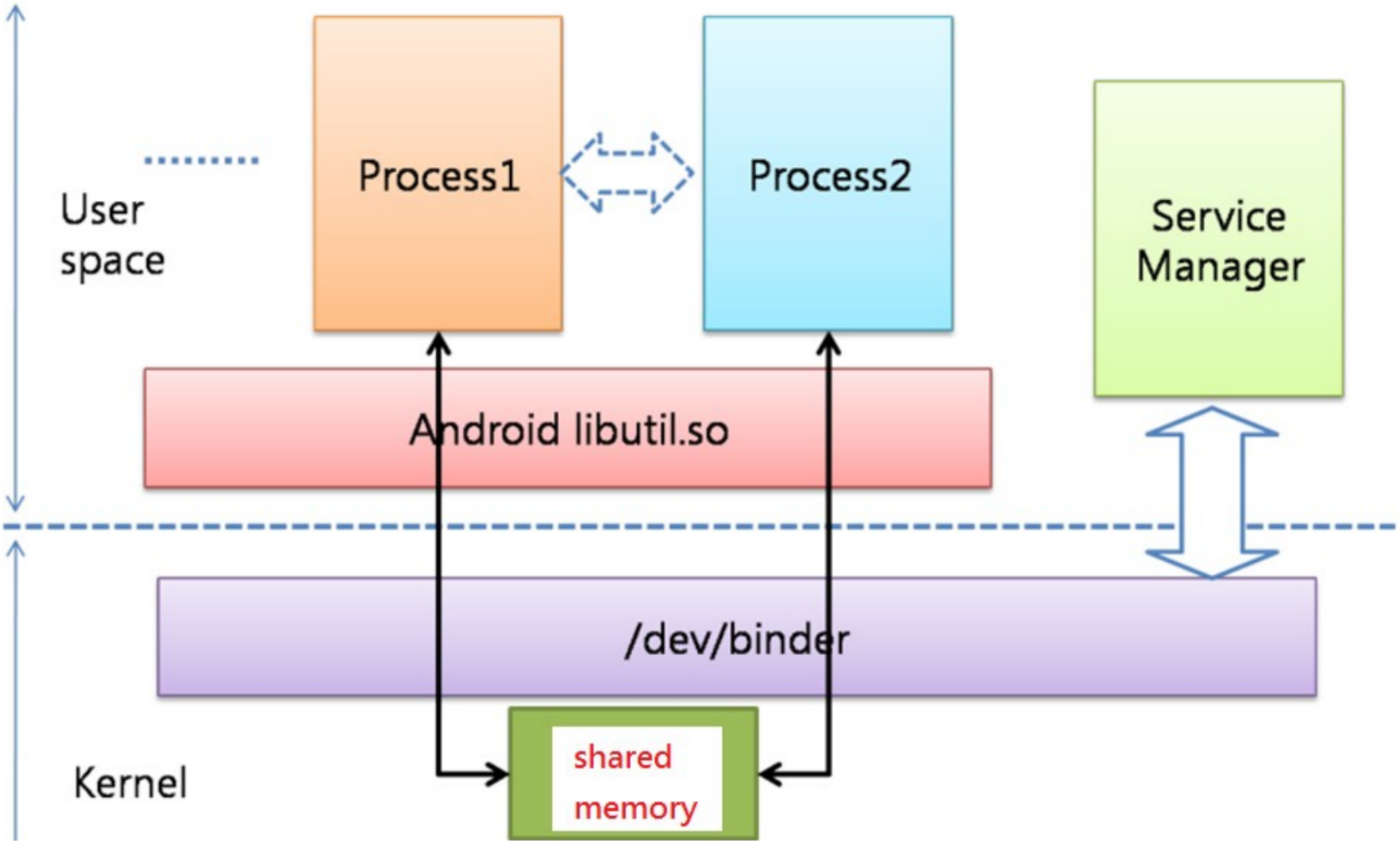


Figure: Collected over 7-day period ending on 4th January 2016, Google.

Application Sandboxing



Binder IPC


User space
Process1
Process2
Service Manager
Android libutil.so
/dev/binder
shared memory
Kernel

Binder IPC

- Isolate kernel from user apps
- All communication between processes passes via the Binder
- Any data type can be sent
- Two components: kernel driver and library loaded in applications

Attacking the Binder?

1. Inject code into target service
2. Hook the function writing data to the driver
3. Listen on target service

Attacking Android for Work?

- Services shared between users
  - Keyboard
  - Phone calls
  - ...
- Flexible
- Nothing displayed on UI
- Subvert file-based encryption from Enterprise apps (e.g. Sophos Mobile Encryption)?

Is it really practical?

- Number of obstacles to first overcome
  - Gaining root access
  - Bypassing SELinux
  - Avoiding root detection
- Will never achieve 100% security
  - Layered security
  - Encrypt the traffic
  - Minimize data travelling acrosss Binder

Conclusion

- Data is not encrypted while device is running
- Bypassing root detection from MDMs is possible
- Data flowing through the Binder can be read by other *rooted* users

Questions?