Contents

1 IPC

Inter-process communication. Each process has its own address space. This provides data isolation and prevents direct interaction between different processes. How can we communicate with a Service, or send an Intent? Use **Binder**.

- Underpins most Android communication, i.e. when we use various system capabilities
- Kernel driver: provides lightweight RPC (remote procedure calls), data passing. C.f. Linux/Unix signals / pipes / sockets etc. Reading and writing Parcels between processes. Process, user ID authority / trust
- Per-process thread pool for handling requests
- Synchronous calls between processes

2 Binder

2.1 Facilities

- Calls: Simple inter-process messaging system. One-way, two-way
- Identifying: PID, UID
- Notification: Link to death, Leaked Service connections
- Managing: Reference counting, object mapping across processes, sleeping and waking worker threads
- Indirect functionality: As a token, sharing fd (file descriptor) to shared memory area

2.2 Implementation

API for apps

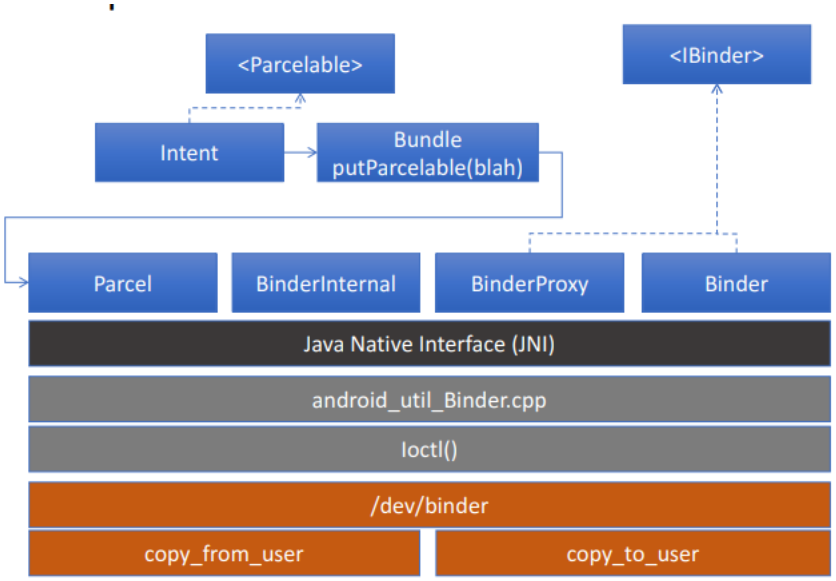
- AIDL
- Java API wrapper
  - Exposes the IBinder interface
  - Wraps the middleware layer
  - Parcelable object marshalling interface

Native Middleware

- Implements the user space (i.e. within a process) facilities of the Binder framework
- Marshalling and unmarshalling of specific data to primitives
- Provides interaction with the Binder kernel driver

Kernel driver

- Supports ioctl system calls from the middleware
- Supports cross-process file operations, memory mapping
- Thread pool for each service application for IPC
- Mapping of objects between processes via `copy_from_user`, `copy_to_user`



placeholder