

A Content-Aware Kernel IPC Firewall for Android

...

David Wu
Sergey Bratus



Overview

- Understanding Binder and how it's used
- Intercepting and modifying data with BinderFilter
- Demos!

@davidwuuuuuuuuuu

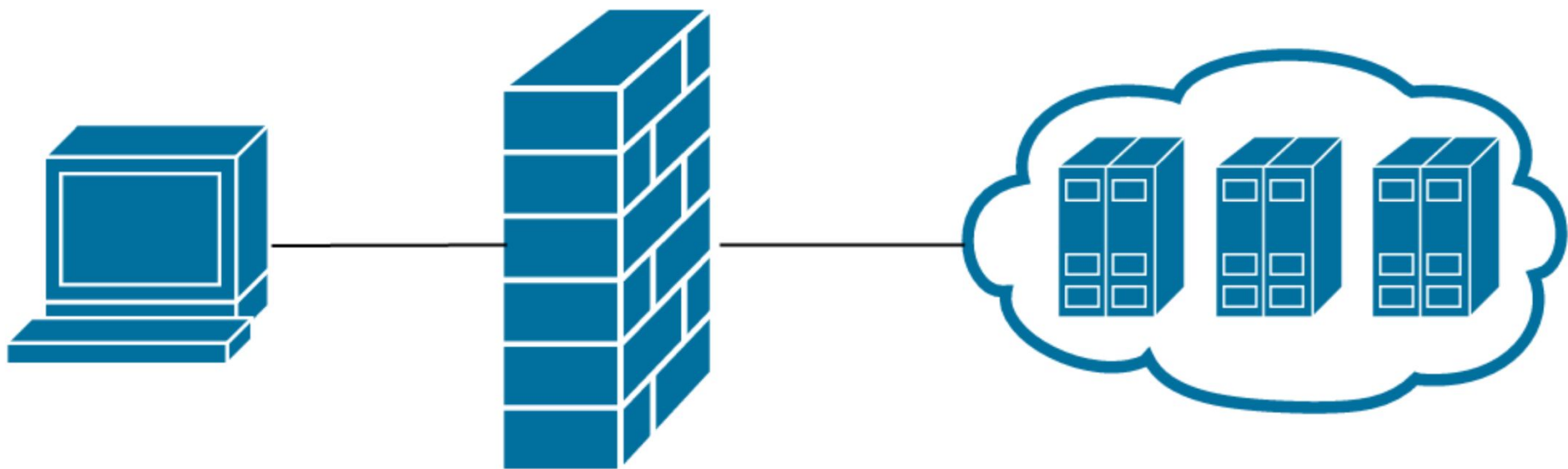
- Undergraduate at Dartmouth
- Software Engineer in Boston
- Network and Linux instrumentation research with Sergey Bratus
- Web analysis automation and Android security research at Ionic Security

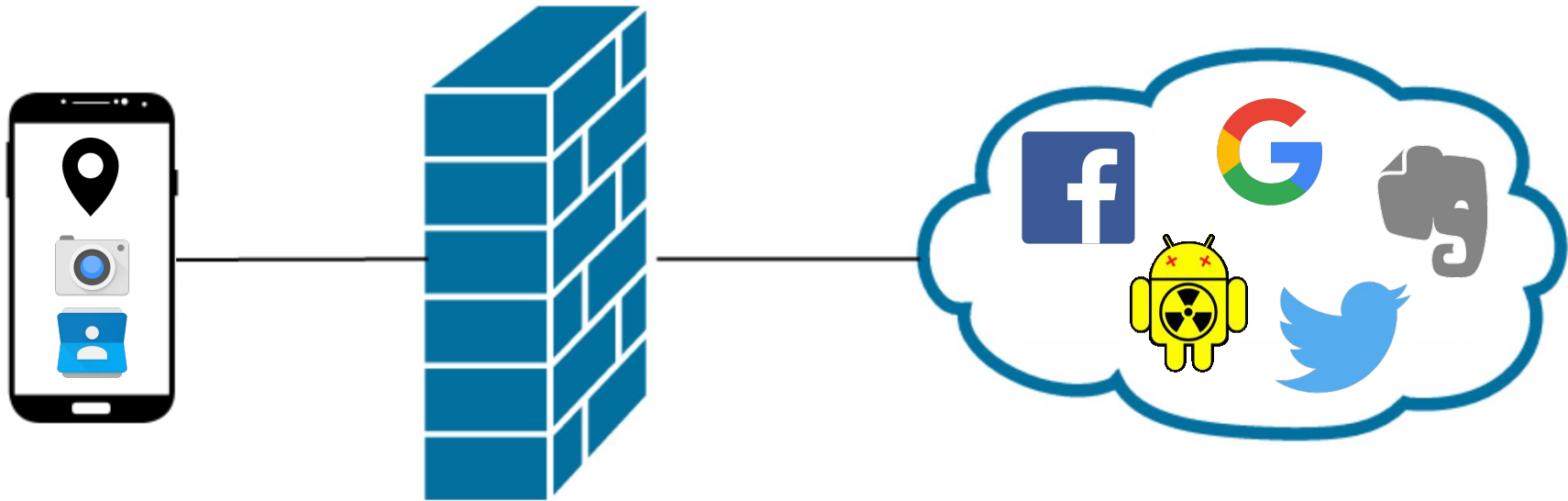
@sergeybratus

- Research Associate Professor at Dartmouth
- LangSec for Penetration Testing
- Weird machines in ELF and DWARF formats
- Hacking 802.15.4/ZigBee digital radio hacking

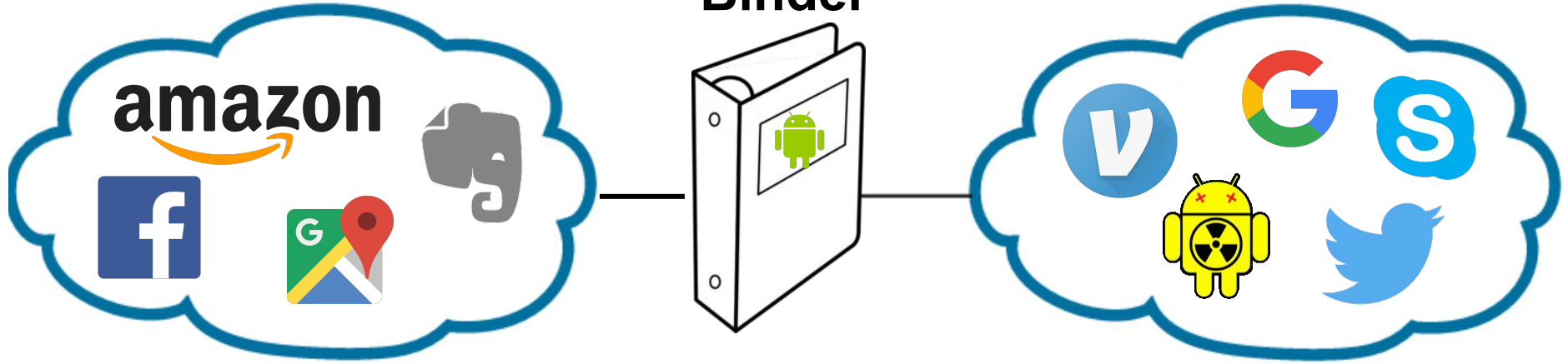
BinderFilter

- Run-time **blocking** and **modification** of all inter-app communication
- Context informed policy decisions
- **Binder** message parser and filter





Binder



Man in the Binder

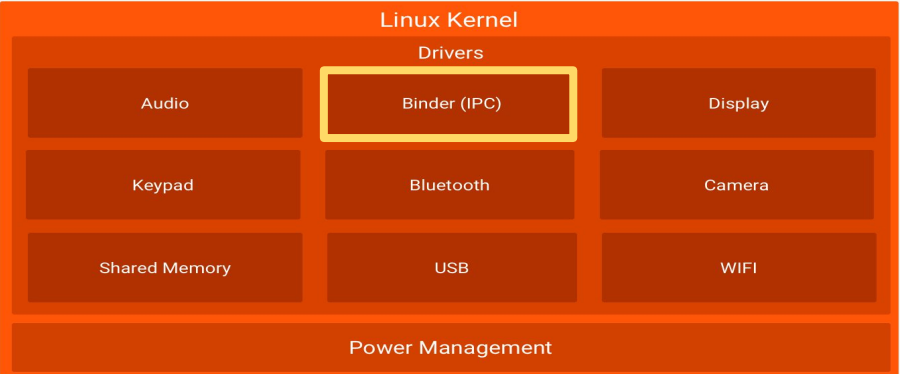
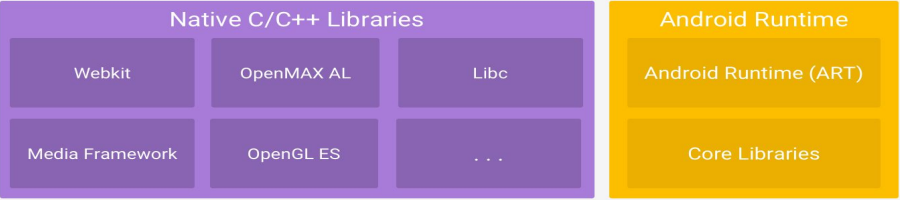
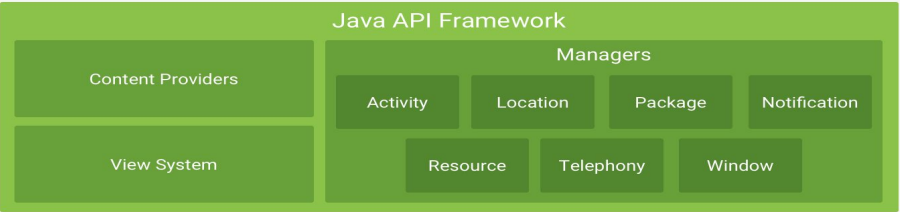
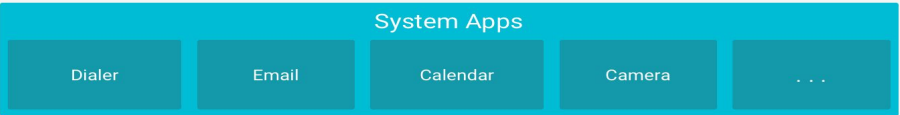
- 2014 Blackhat Europe - Nitay Artenstein, Idan Revivo
- Keylogger, SMS message interceptor
- “Most secure apps protect their data, but don’t bother with data moving between in-app Activities... this data goes through Binder”



Android Security Concepts

- Built on Linux
 - SELinux, file permissions, system calls
 - Sandboxing enforced by UID
 - (each application is a different Linux user)
- Permissions
 - Android 6.0 introduced dynamic permissions for certain messages
 - 30% of users have Android 6.0+ [1]





Application Layer

Application Framework Layer

Core Libraries Layer

Kernel Layer

Binder

- Android's IPC system
- Implemented as a Linux kernel driver
 - `/dev/binder`
 - `/drivers/staging/android/binder.{c,h}`
- Every IPC message goes through Binder driver
- Supports:
 - Security tokens
 - Death notifications
 - (local) RPC
 - Intents
 - ContentProviders



Separate process address spaces enforced by kernel



writeToParcel()

userland
kernel

readFromParcel()

copy_from_user()

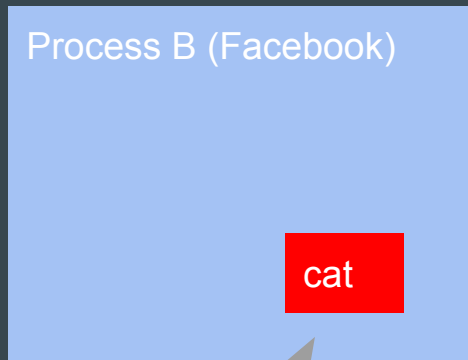
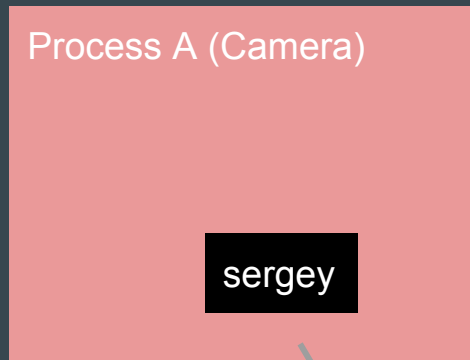
copy_to_user()



Demo

```
binderfilter -s -m "android.permission.CAMERA" -u 10078 -a 3 --modify-data="cat.jpg"
```

Separate process address spaces enforced by kernel



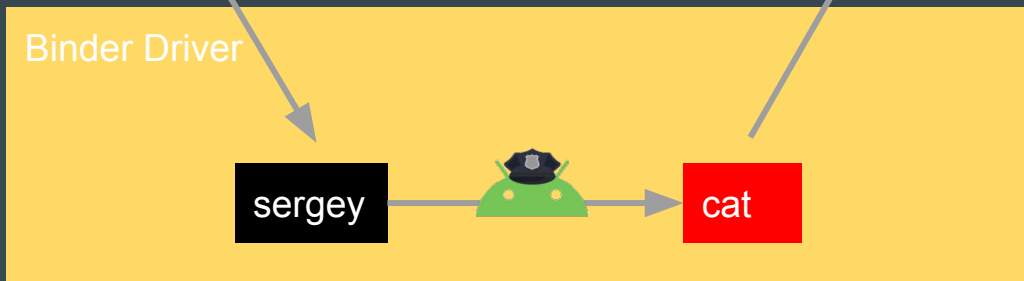
writeToParcel()

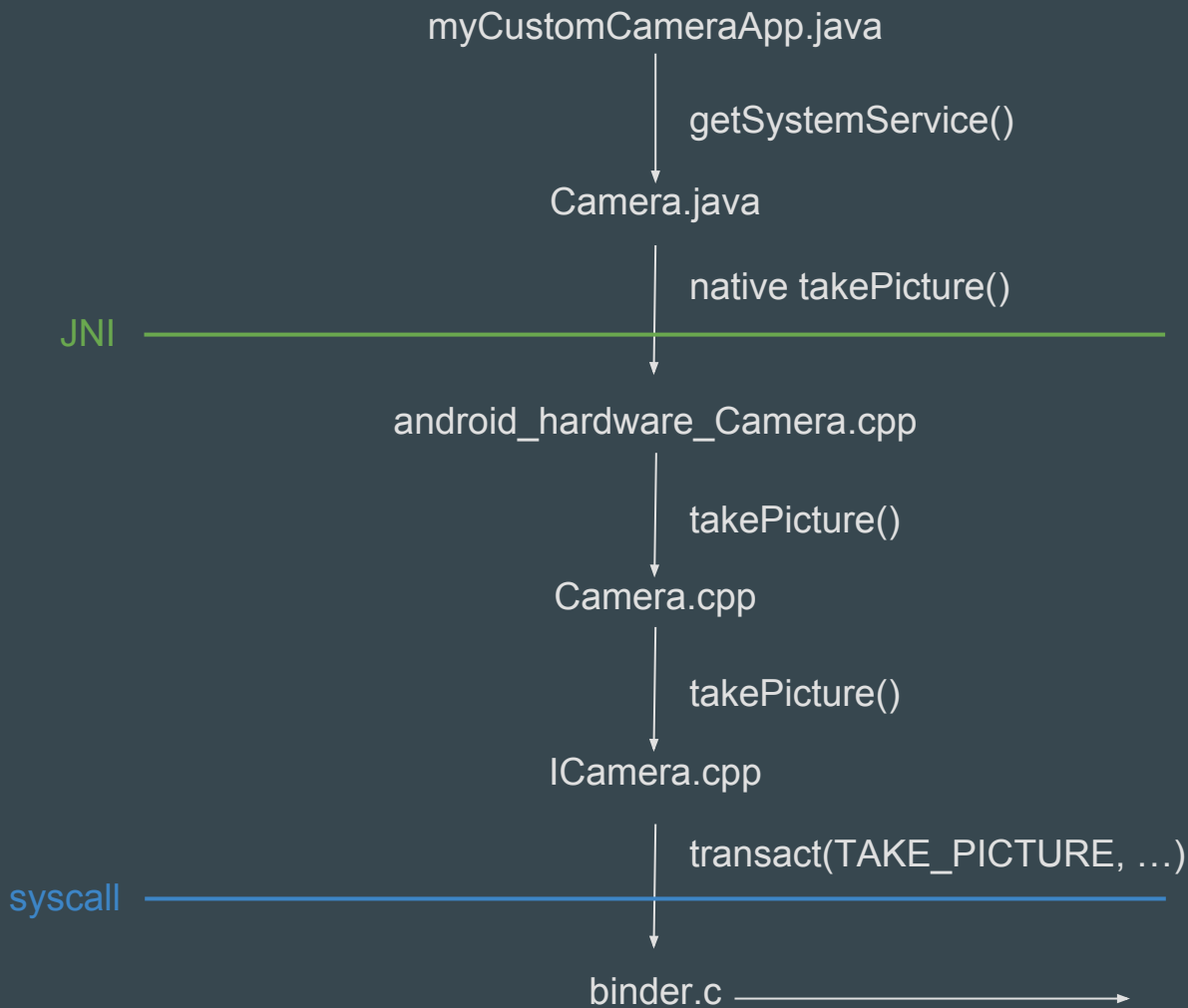
userland
kernel

readFromParcel()

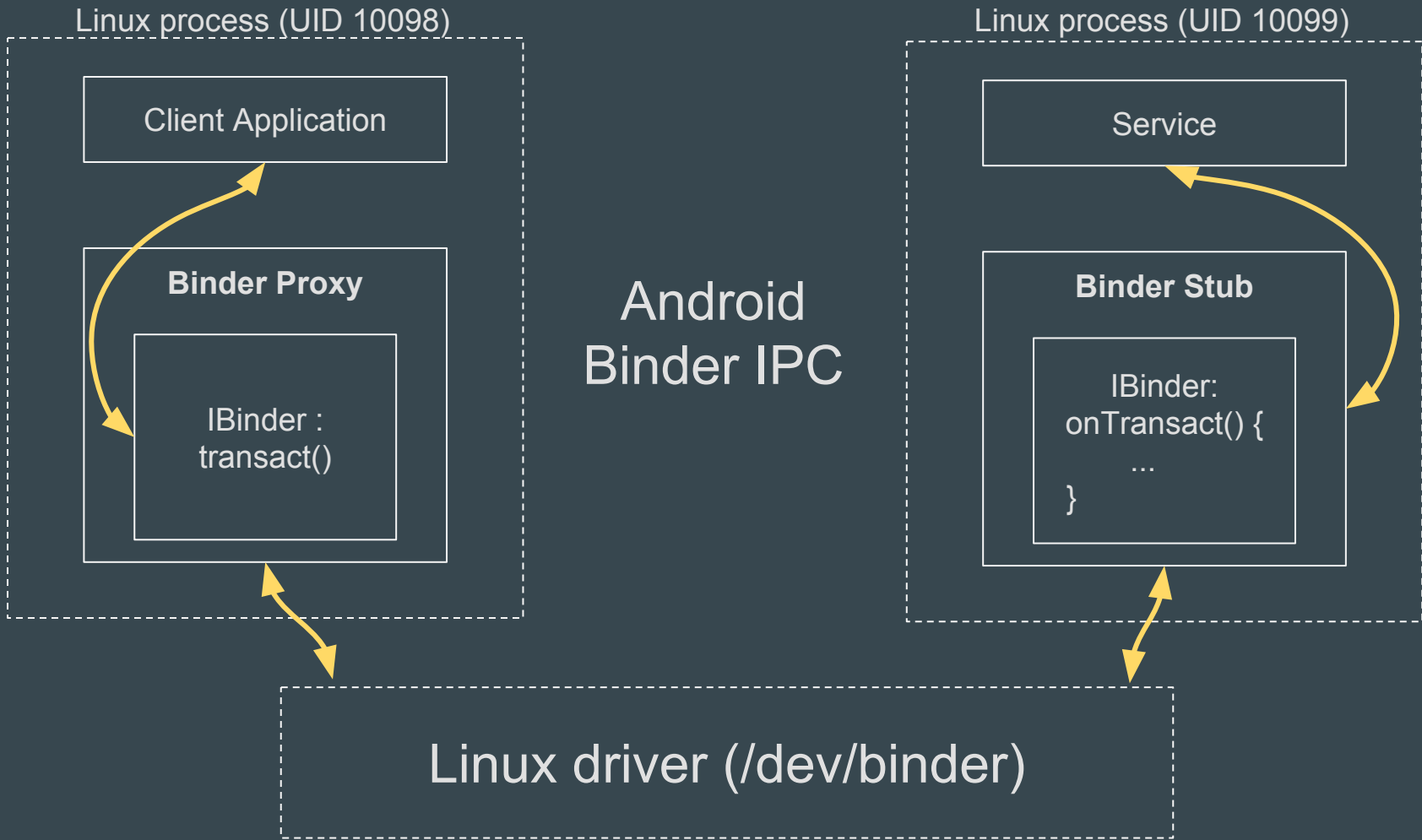
copy_from_user()

copy_to_user()





Visualizations



Service

Binder

Client

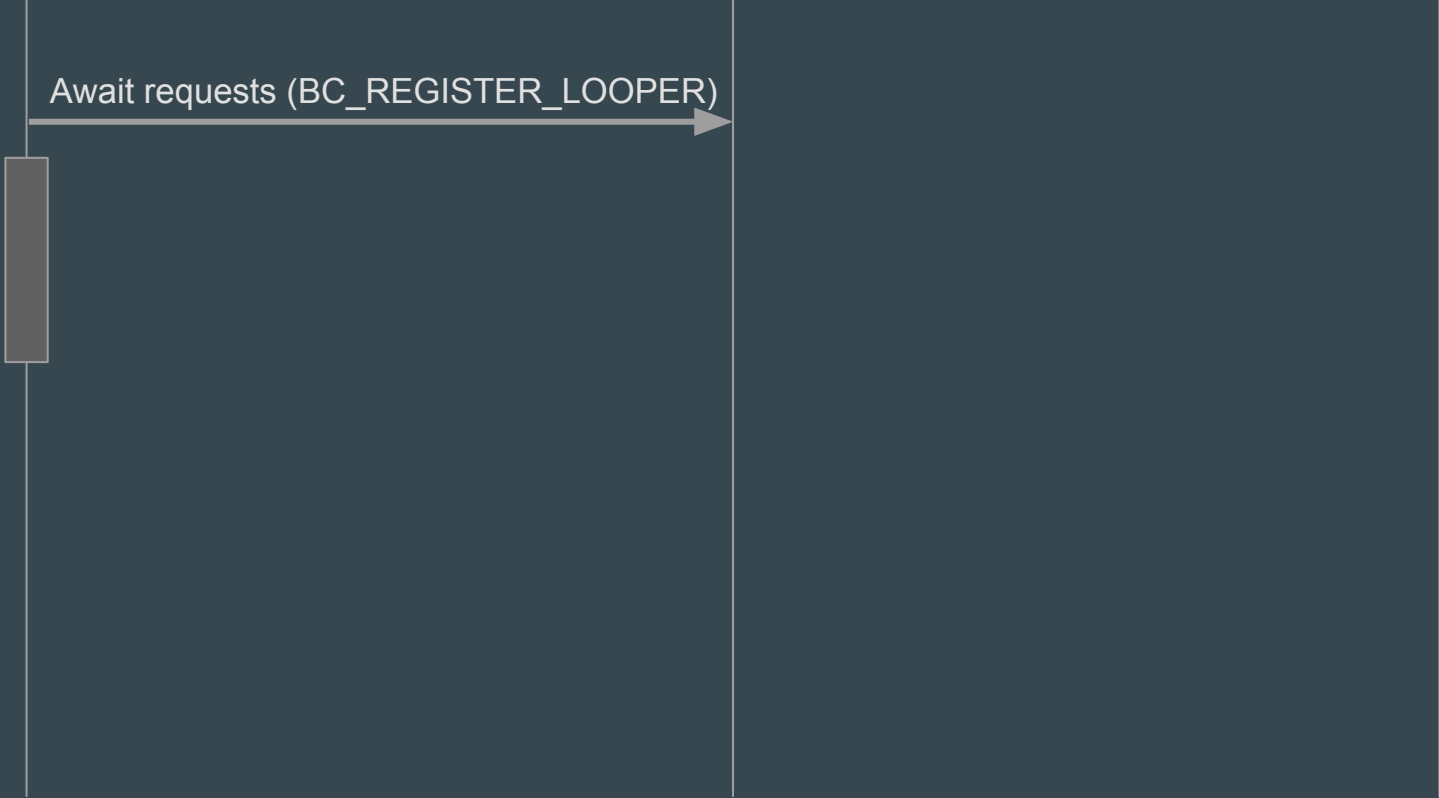
Service

Binder

Client

Await requests (BC_REGISTER_LOOPER)

Service
thread
sleeps



Service

Binder

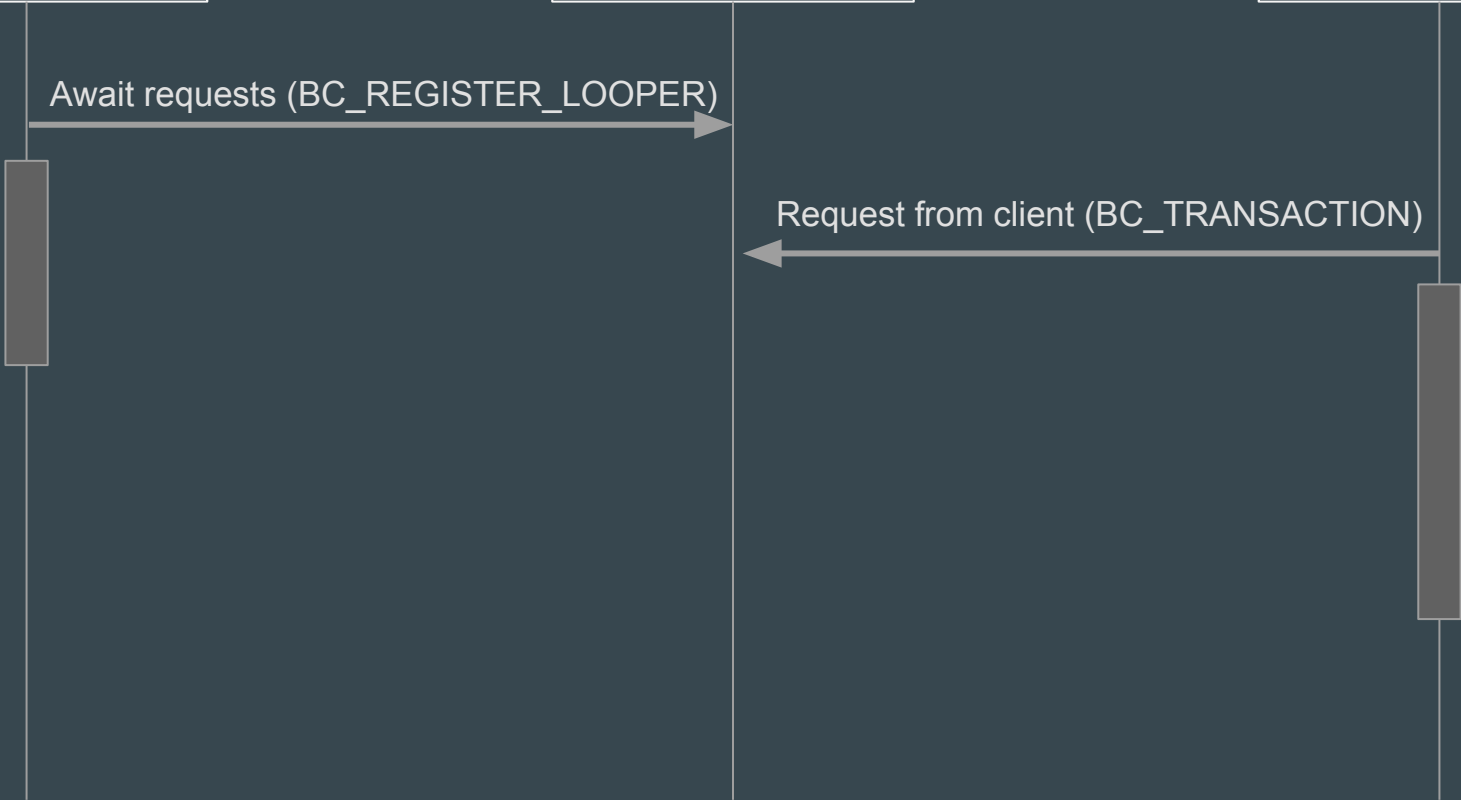
Client

Await requests (BC_REGISTER_LOOPER)

Service
thread
sleeps

Request from client (BC_TRANSACTION)

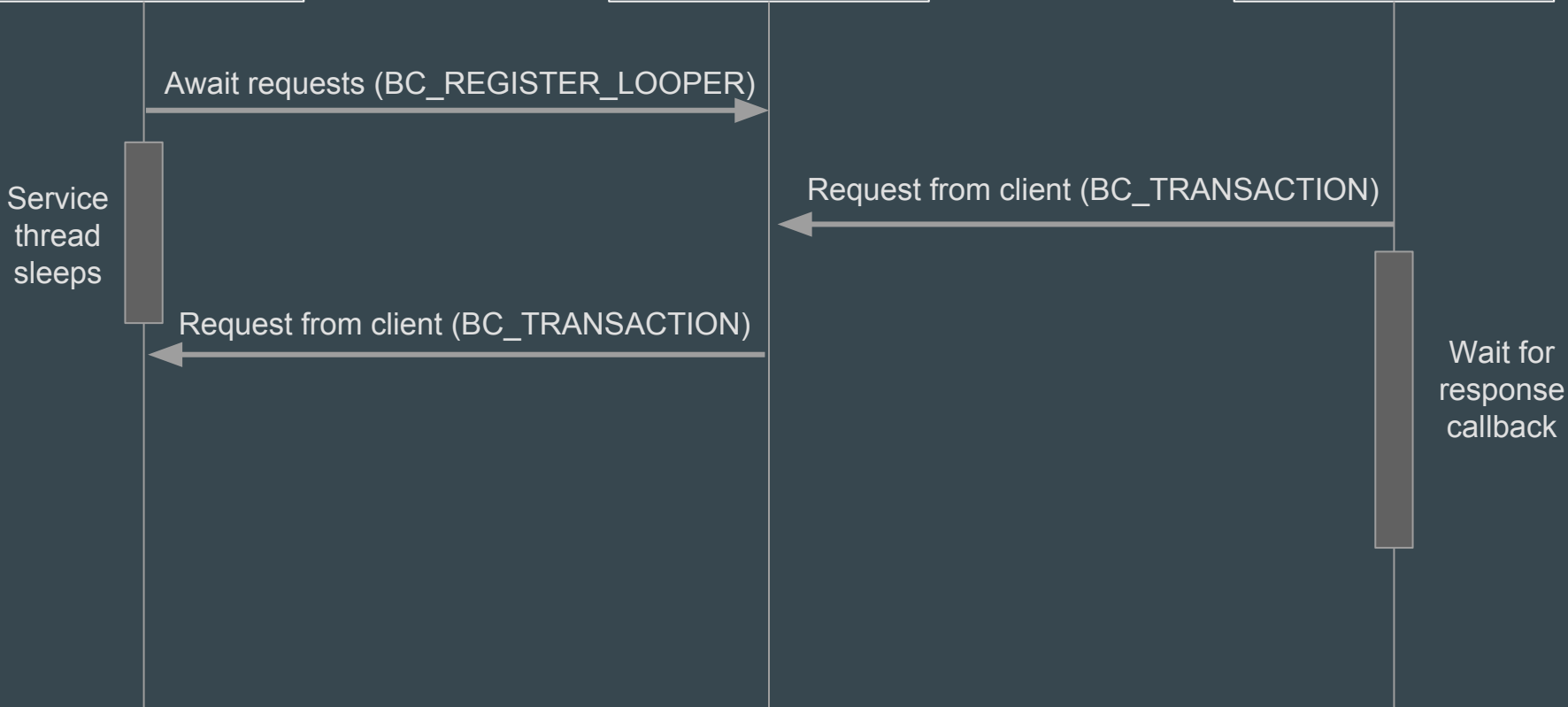
Wait for
response
callback

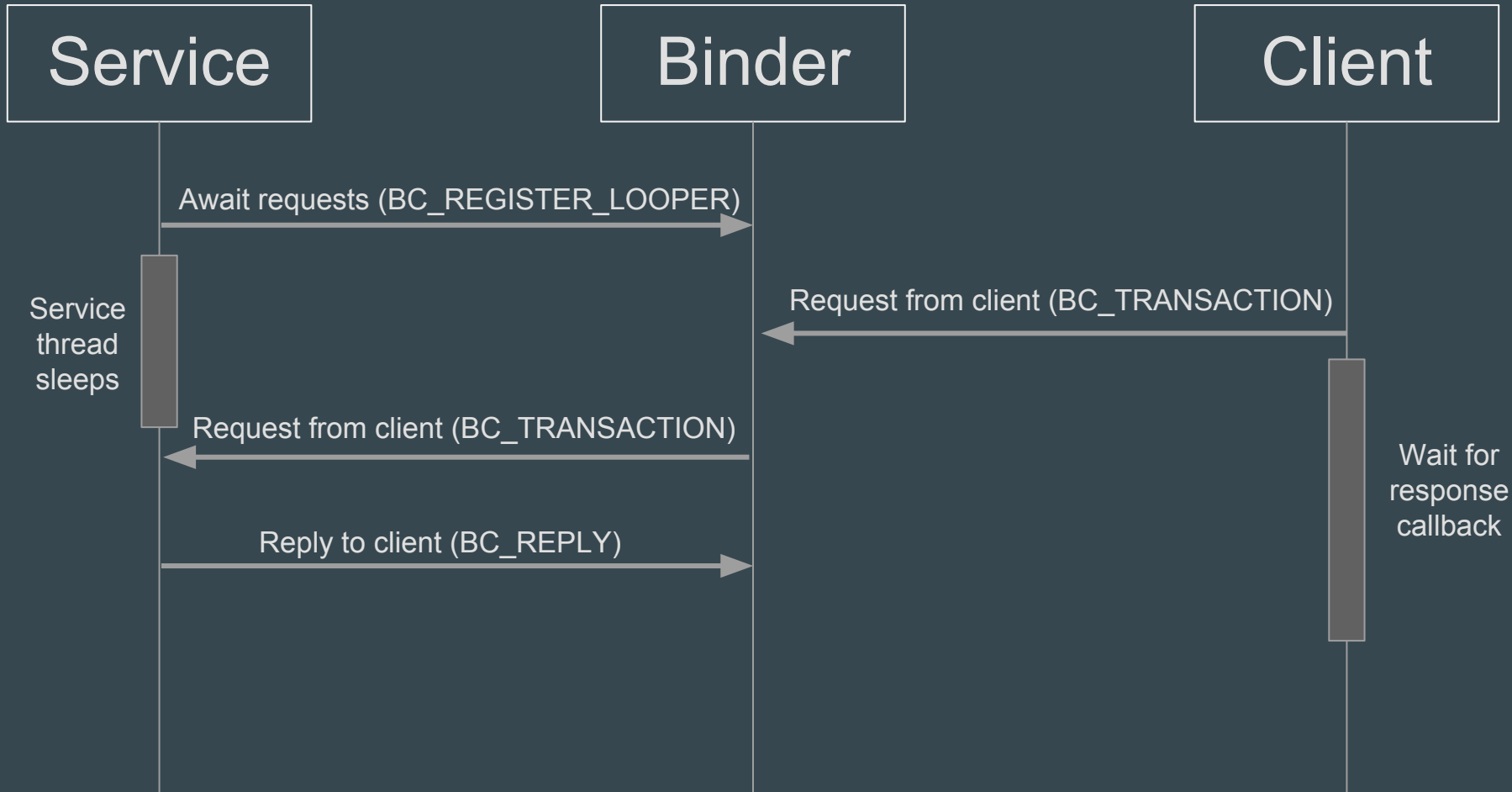


Service

Binder

Client





Service

Binder

Client

Await requests (BC_REGISTER_LOOPER)

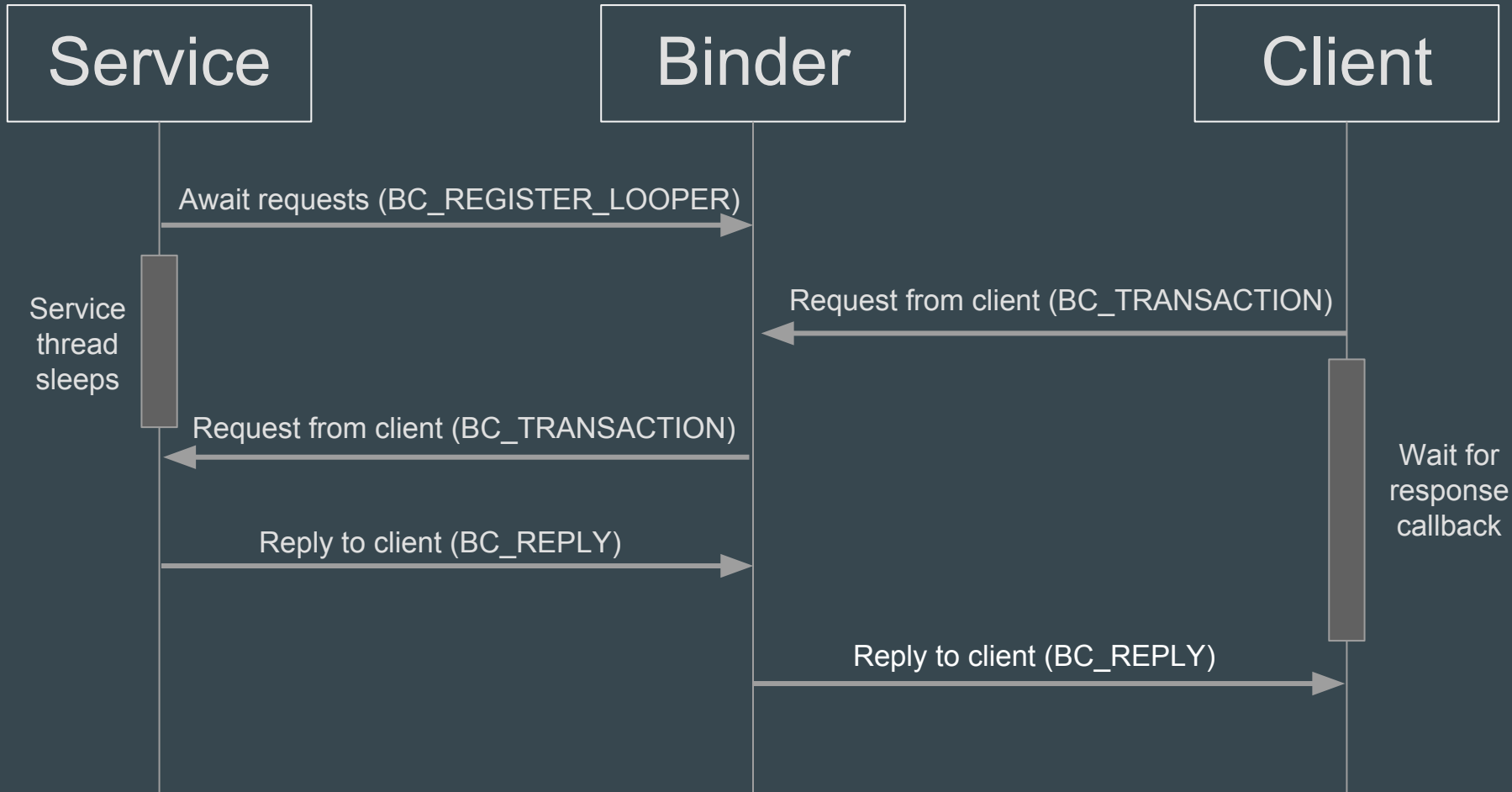
Service thread sleeps

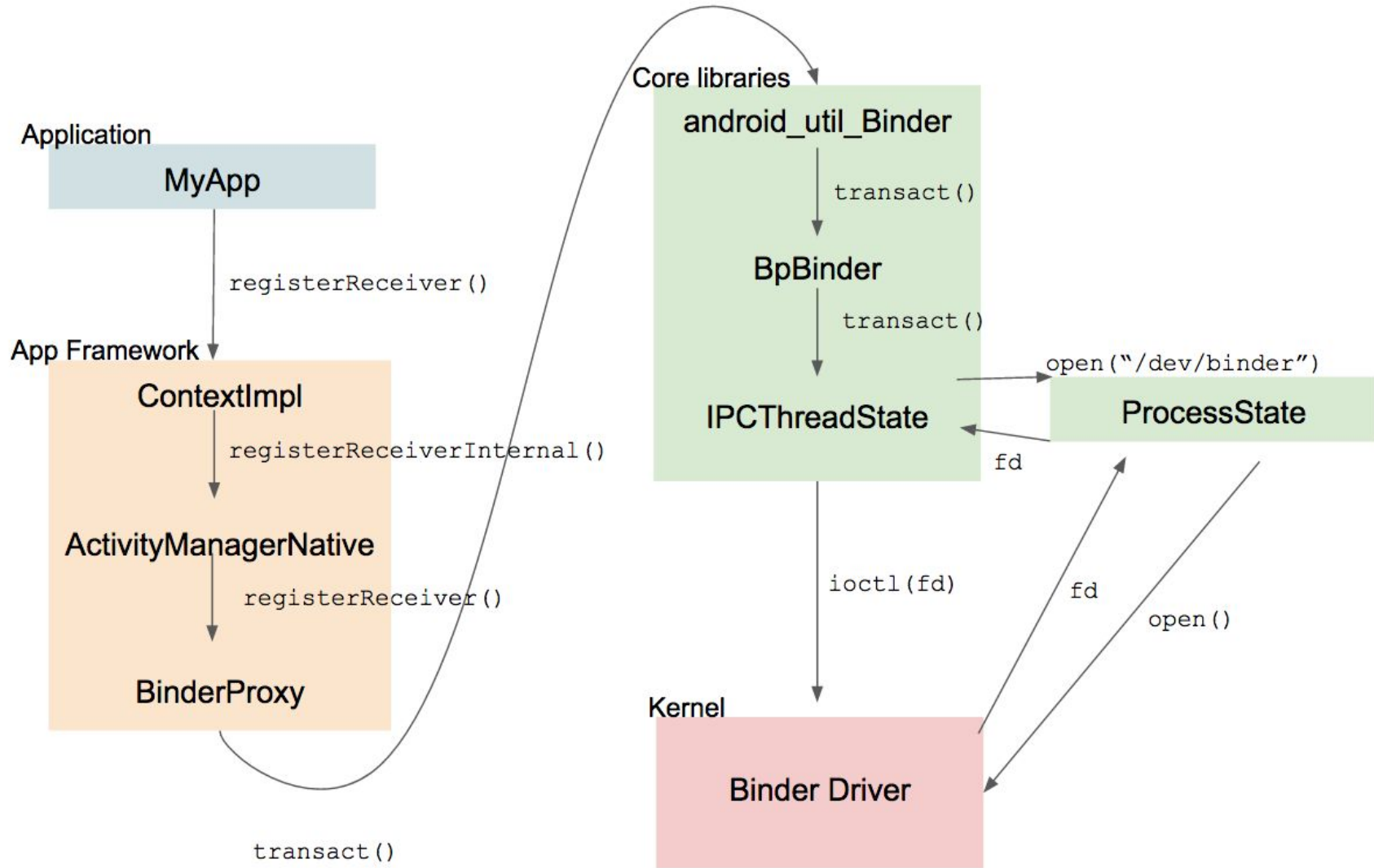
Request from client (BC_TRANSACTION)

Request from client (BC_TRANSACTION)

Reply to client (BC_REPLY)


Wait for response callback





Application Layer

MyApp.java



```
Intent batteryStatus =  
Context.registerReceiver(null, new  
IntentFilter(  
Intent.ACTION_BATTERY_CHANGED);
```

ContextImpl.java

```
registerReceiver() ->  
registerReceiverInternal() ->  
ActivityManagerNative.registerReceiver()
```

ActivityManagerNative.java

```
Parcel data = Parcel.obtain()  
data.writeString(packageName)  
filter.writeToParcel(data)  
IBinder.transact(data, reply)
```

BinderProxy.java (implements IBinder)

```
transact() ->  
native transactNative() // JNI
```

android_util_Binder.cpp

android_os_BinderProxy_transact() ->
IBinder.transact()

BpBinder : IBinder

IPCThreadState::self()->transact()



Core Libraries

```
IPCThreadState.cpp  
  
transact() ->  
waitForResponse() ->  
talkWithDriver()
```

```
open("/dev/binder")
```

```
ProcessState.cpp
```

```
mParcel.write(data)
```

```
Parcel.cpp  
  
// writes Java  
parcel data to  
this process'  
address space
```



```
ioctl(fd, BINDER_WRITE_READ, mParcel)
```

Linux Kernel

```
binder.c
```

IPC “Packets”

```
struct binder_transaction_data {
    /* The first two identify
    the target and contents of
    the transaction.
    */
    union {
        size_t handle;
        void *ptr;
    } target;

    void *cookie;
    unsigned intcode;
    unsigned intflags;

    /* General information about the transaction.*/
    pid_t sender_pid;
    uid_t sender_euid;
    size_t data_size;
    size_t offsets_size;

    union {
        struct {
            /* transaction data */
            const void *buffer;
            const void *offsets;
        } ptr;
        uint8_t buf[8];
    } data;
};

struct binder_write_read {
    signed long write_size;
    signed long write_consumed;
    unsigned long write_buffer;
    signed long read_size;
    signed long read_consumed;
    unsigned long read_buffer;
};

struct flat_binder_object {
    /* 8 bytes for large_flat_header. */
    unsigned long type;
    unsigned long flags;

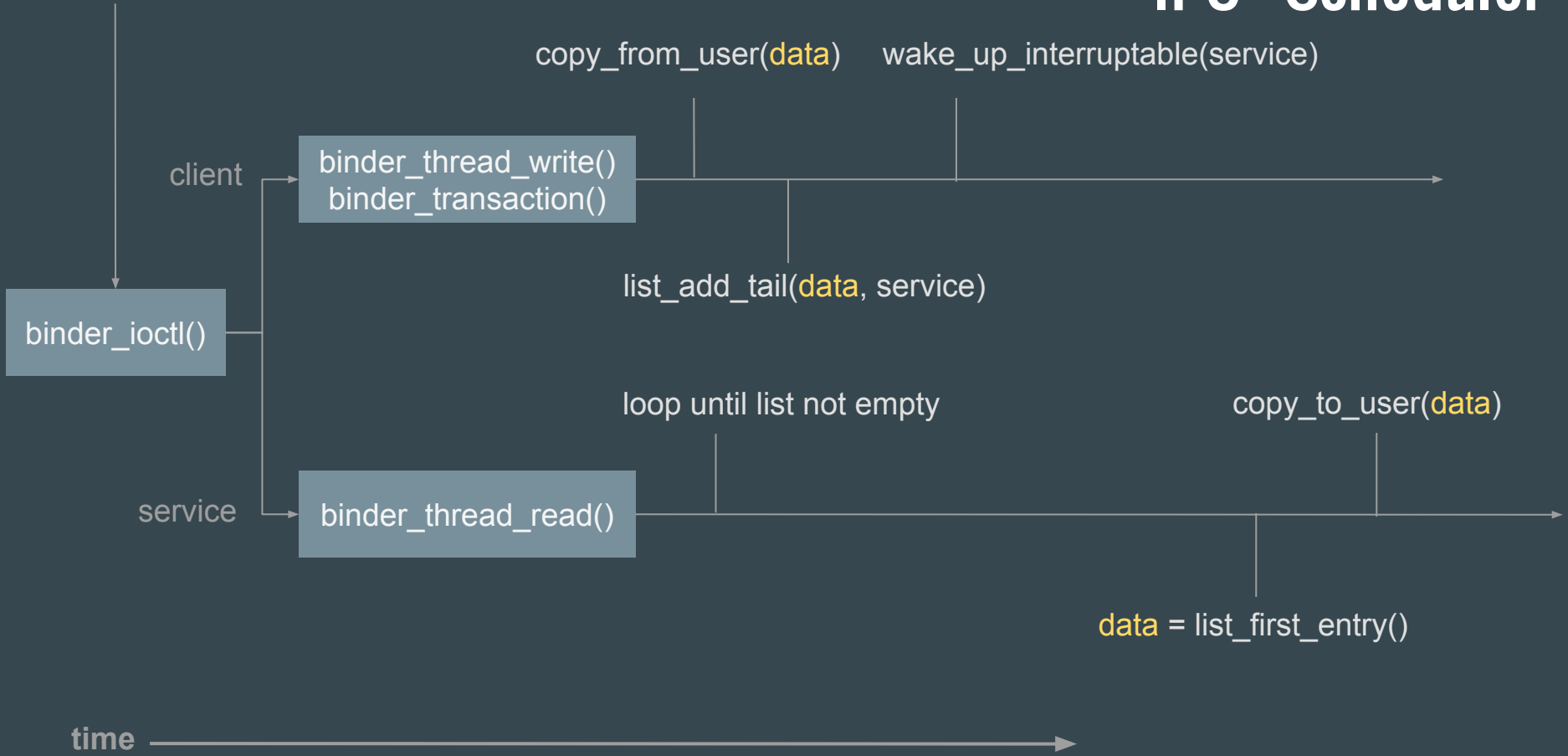
    /* 8 bytes of data. */
    union {
        void *binder; // local obj
        signed long handle; // remote obj
    };

    /* extra data associated with local object */
    void *cookie;
};
```

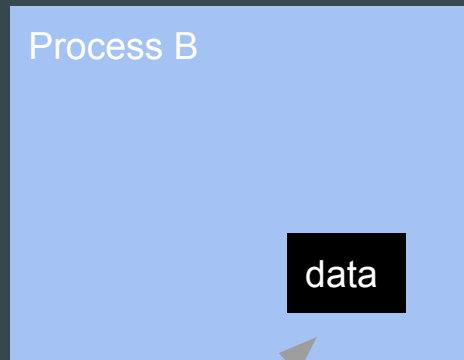
The diagram consists of three arrows. One arrow points from the `struct binder_write_read` definition to the `union` block within `struct binder_transaction_data`. A second arrow points from the `struct flat_binder_object` definition to the `union` block within `struct binder_transaction_data`. A third arrow points from the `const void *buffer` field in the `struct` block of `struct flat_binder_object` to the `const void *buffer` field in the `struct` block of `struct binder_transaction_data`.

IPC "Scheduler"

Blocking ioctl syscall



Separate process address spaces enforced by kernel



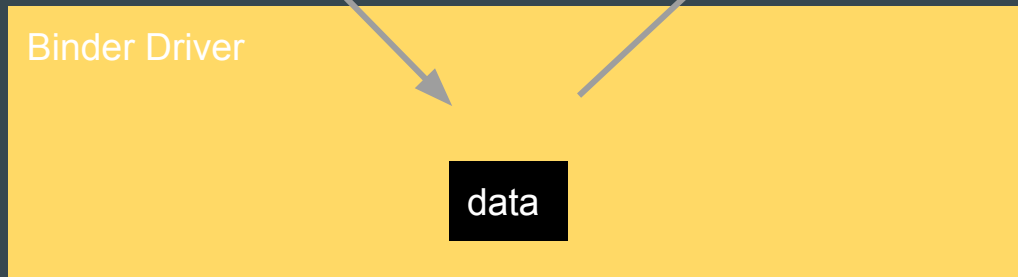
writeToParcel()

readFromParcel()

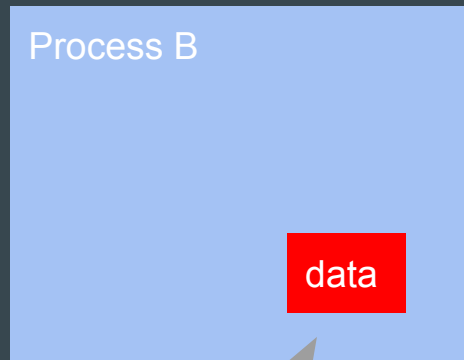
userland
kernel

copy_from_user()

copy_to_user()



Separate process address spaces enforced by kernel



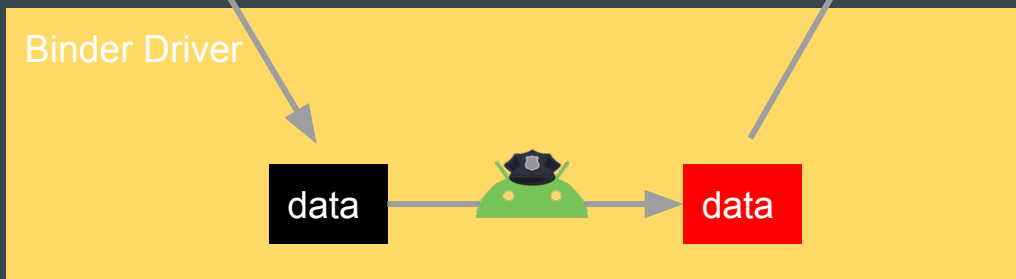
writeToParcel()

userland
kernel

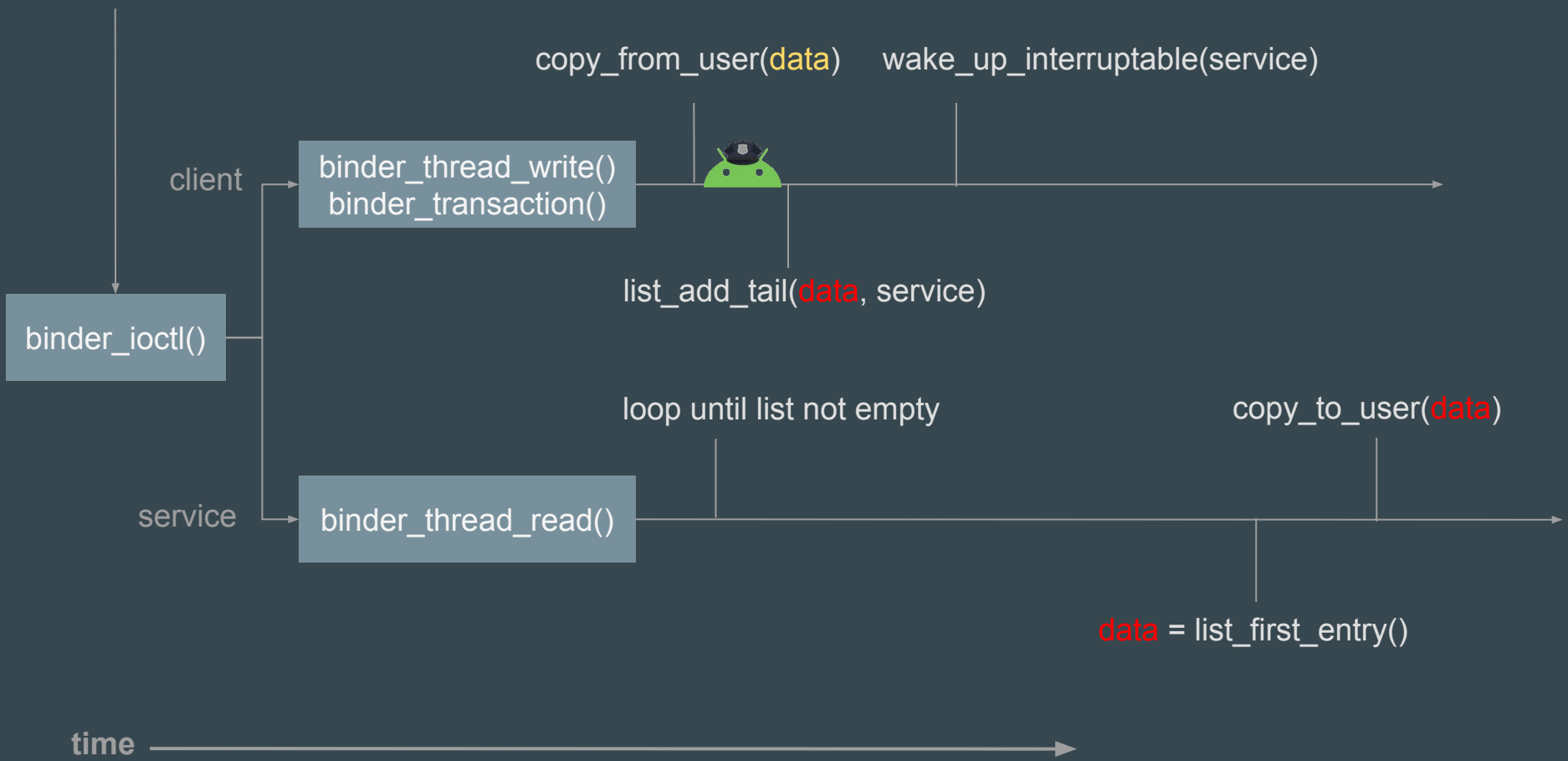
readFromParcel()

copy_from_user()

copy_to_user()



Blocking ioctl syscall



BinderFilter

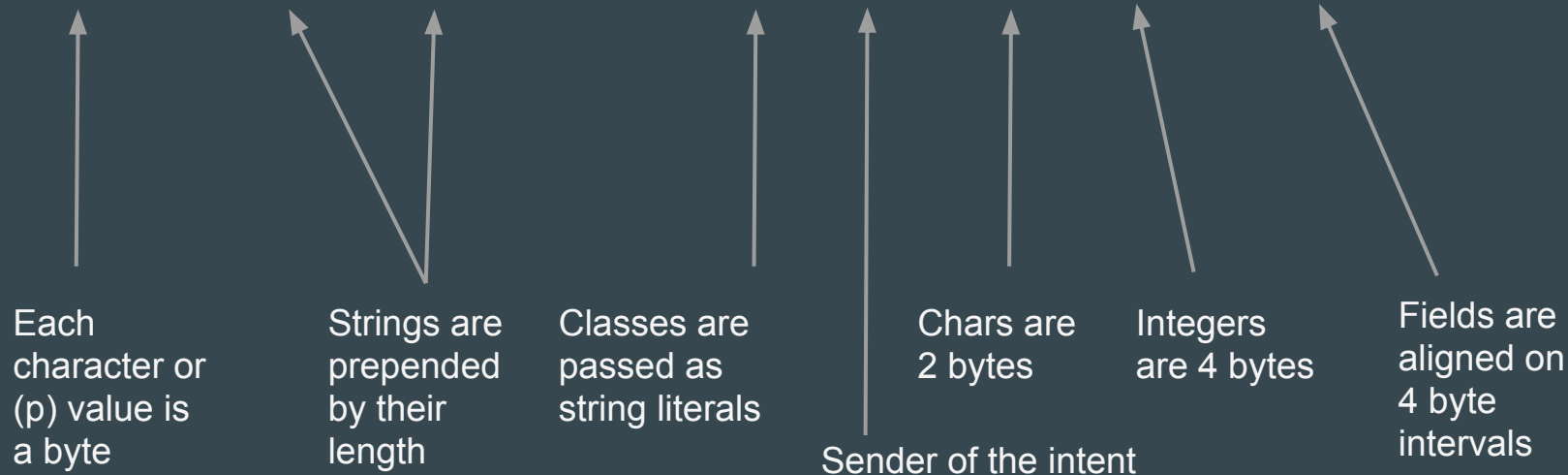
Enhanced logs

- Existing binder.c logs
 - printk(), TRACE_EVENT(), seq_printf()
- Existing: [49431.544219] binder: 9916:9916 BC_TRANSACTION 683674 -> 198
- node 289403, data 8dc12180 (null) size 80-0
- Enhanced: [14:33:56.084452] binder_command BC_TRANSACTION: process pid 9916 (android.picky), thread pid 9916 -> process pid 198 (/system/bin/surfaceflinger), node id 289403, transaction id 683674, data address 8dc12180, data size 80, offsets address null, offsets size 0

Parsing message contents (Man in the Binder)

Binder buffer contents in memory

```
{ (0) (64) (24) (0) android.os.IPowerManager (0) (0) (1) (0) (0) (0) }
```



Modifying Saved Pictures

```
{(4)H(28)(0)android.app.IActivityManager(0)(0)(133)*bs(127)(1)(0)P(196)(180)(174)(224)(145)(181)(172)(19)(0)com.facebook.katana(0)"(0)android.media.action.IMAGE_CAPTURE(0)(0)(0)(0)(255)(255)(255)(255)(3)(0)(255)(255)(255)(255)(255)(255)(255)(255)(255)(0)(0)(0)(0)(0)(0)(1)(0)(1)(0)(0)(0)(0)(0)(1)(0)(13)(0)text/uri-list(0)(0)(0)(1)(0)(1)(0)(255)(255)(255)(255)(255)(255)(255)(0)(0)(1)(0)(3)(0)(4)(0)file(0)(0)(0)(0)(0)(0)(0)(0)(0)(0)(0)(0)(2)(0)(62)(0)/storage/emulated/0/Pictures/Facebook/FB_IMG_1464314001208.jpg}
```

Demo

Evernote: microphone

```
binderfilter -s -m "android.permission.RECORD_AUDIO" -u 10092 -a 1
```

Google Maps: location

```
binderfilter -s -m "android.permission.ACCESS_FINE_LOCATION" -u 10056 -a 1
```

Photos: read storage

```
binderfilter -s -m "android.permission.READ_EXTERNAL_STORAGE" -u 1000 -a 1
```

Camera: ContentProvider (service for saving pictures)

```
binderfilter -s -m "android.permission.READ_EXTERNAL_STORAGE" -u 1000 -a 2
```

```
binderfilter -s -m "ContentProvider" -u 10044 -a 1
```

Blocking messages

- Blocking generic strings in Binder messages is dangerously powerful
- Permissions are passed as string literals in IPC messages
- Check uid, context
- Check binder message content for message with `strstr`
- Clear out the entire message with `memset`

Blocking Permissions

```
{ (0)@ (28) (0) android.app.IActivityManager (0) (0) ) (0) android.permission.ACCESS_COARSE_LOCATION (0) (155) (9) (0) ) ' (0) }
```


Demo

Block play store from installing

```
binderfilter -s -m "com.android.vending.INTENT_PACKAGE_INSTALL_COMMIT" -u  
10018 -a 1
```

```
binderfilter -s -m "com.android.vending.INTENT_PACKAGE_INSTALL_COMMIT" -u  
10018 -a 2
```

Blocking system permissions

Sent before Google Play Store installs the “com.groupme.android” package:

```
{ (0) (64) (28) (0) android.app.IActivityManager (0) (0) (1) (0) (19) (
0) com.android.vending (0) (133) *bs (127) (1) (0) (0) (0) (0) (255)
(255) (255) (255) k (157) +m (1) (0) (1) (0) (1) (0) E (0) com.android.ven
ding.INTENT_PACKAGE_INSTALL_COMMIT.com.groupme.android (0) (0)
(0) (255) (255) (255) (255) (0) (0) (255) (255) (255) (255) (255) (255) (
255) (255) (0) (0) (0) (0) (0) (0) (0) (0) (254) (255) (255) (255) (255) (2
55) (255) (255) (255) (255) (255) (255) (0) (0) H (0) (0) (0) (0) }
```

Demo

Block camera when Wifi SSID matches "ssid"

```
binderfilter -s -m "ContentProvider" -u 10044 -a 2
```

```
binderfilter -s -m "android.permission.CAMERA" -u 10044 -a 1 --context 2 --context-type 2 --context-value "shmoocon-romp"
```

Block Chrome saving to external storage when Blendoku is running

```
binderfilter -s -m "android.permission.WRITE_EXTERNAL_STORAGE" -u 10035 -a 1 --context 9 --context-type 2 --context-value "com.lonelyfew.blendoku"
```

```
binderfilter --print-system-context
```

Wifi SSID context

```
{ (0)@ (30) (0) android.app.IApplicationThread(0) (0) (133)h(127) (07
) (247) (07) (07) (07) $ (07) android.net.conn.CONNECTIVITY_CHANGE (07
) (07) (07) (07) (255) (255) (16) (0) (255) (255) (255) (255) (05) (05) (05)
(05) (05) (05) (05) (05) (254) (255)x (05)BD (45) (05) (11) (0) networkInf
o(0) (4) (0) (23) (0) android.net.NetworkInfo(0) (1) (0) (0) (0) (4) (0) W
IFI (0) (0) (0) (0) (0) (0) (9) (0) CONNECTED (0) (9) (0) CONNECTED
(0) (0) (0) (1) (0) (0) (0) (255) (255) (18) (0) "shmoocon-romp" (0) (0) (11
) (0) networkType(0) (1) (0) (1) (0) (13) (0) inetCond... }
```

Demo

Modifying arbitrary strings

```
binderfilter -s -m "binderfilter.arbitrary.shmoocon-wpa" -u 1000 -a 3 --modify-data "CatsRule-wpa"
```

Modify GPS location

```
binderfilter -s -m "android.permission.ACCESS_FINE_LOCATION" -u 10056 -a 2
```

```
binderfilter --get-gps-bytes --latitude "43.7069062" --longitude "-72.2870538"
```

```
binderfilter -s -m "android.permission.ACCESS_FINE_LOCATION" -u 1000 -a 3  
--modify-data="121.43.1.231.123.218.69.64.82.204.230.22.95.18.82.192."
```

GPS location context

```
{ (0)@(29) (0) android.content.IIntentSender (0) (0) (0) (1) (0) (255)
) (255) (255) (255) (0) (0) (255) (255) (255) (255) (0) (0) (255) (255) (2
55) (255) (255) (255) (255) (255) (0) (0) (0) (0) (0) (0) (0) (0) (254) (25
5) (255) (255) (224) (4) (0) BNDL(3) (0) 8 (0) com.google.android.location
tion.internal.EXTRA_LOCATION_LIST(0) (0) (11) (0) (1) (0) (4) (0)
(25) (0) android.location.Location(0) (7) (0) network(0)
(192) (191) (187) (145) T(1) (0)@(165) R(132) \ (0) (177) (237) (254) (1
94) (60) (218) (69) (64) (121) (189) (234) (183) (101) (18) (82) (192) (0
) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (1) (0) u(19) (...}
```

$* (\text{double}^*) (\{177, 237, 254, 194, 60, 218, 69, 64\}) = 43.704979$

$* (\text{double}^*) (\{121, 189, 234, 183, 101, 18, 82, 192\}) = -72.287458$

Summary

- All IPC messages go through Binder
- Logging, blocking, and modification for any message at runtime
- Contextual blocking
 - WiFi state, Bluetooth state, Apps running
- Modification of message data
 - Camera, Location

Future work

- More contexts
- Message modification library
- Kernel patch

References

- [1] <https://developer.android.com/about/dashboards/index.html>
- [2] https://developer.android.com/reference/android/Manifest.permission.html#BLUETOOTH_PRIVILEGED
- [3] http://androidxref.com/6.0.1_r10/xref/frameworks/base/services/core/java/com/android/server/pm/PackageManagerService.java#9557
- [4] <https://developer.android.com/guide/components/bound-services.html#Creating>
- [5] http://androidxref.com/6.0.1_r10/xref/frameworks/base/core/java/android/hardware/Camera.java#1412
- [6] <https://developer.android.com/reference/android/hardware/Camera.html>
- [7] <https://developer.android.com/guide/topics/location/strategies.html>
- [8] <http://tools.android.com/tech-docs/android-ndk-preview>

<https://github.com/dxwu/BinderFilter>

<http://binderfilter.org/>

Installation methods

- Cross-compile kernel sources with our Binder hook
- Flash new kernel image onto Android with fastboot
 - This preserves user information, apps, and state!
- Requirements:
 - Linux build env (Include headers don't work on OSX)
 - adb, fastboot, abootimg
 - Unlocked bootloader, root access
- <https://github.com/dxwu/BinderFilter/wiki/Setup>