MAN IN THE BINDER: HE WHO CONTROLS IPC, CONTROLS THE DROID

Nitay Artenstein - nitaya@checkpoint.com
Idan Revivo - idanr@checkpoint.com
Who Are We?

Nitay Artenstein
- Researcher at Check Point
- Used to do pentesting in Africa (with a machete)
- Now does more risky stuff, such as kernel exploits

Idan Revivo
- Researcher at Check Point
- When he’s not breaking Android, he breaks his trainees at the gym
- Contributor to Cuckoo Project
Overview
ev·o·lu·tion  (ēvəˌlōəˈshən, əˈvə-)  

*n.* A gradual process in which something changes into a different and usually more complex or better form
Malware on Windows
Malware on Android
Why the Big Difference?

- The sandbox
- Android is a complicated environment
- Do we work in Java? JNI? C? Native ARM?
How to Write Malware in this Mess?
Welcome to Binder
• Android Malware Today
• Developer Point-of-View
• What is Binder?
• Man In The Binder Attacks
• Possible Solutions
Android Malware Attacks
What Do Mobile Malware Authors Want?

- Sending SMS to premium numbers
- Location tracking
- Secondary APK installation
- Link clicking
- Bank fraud
- Stealing personal information
- Etc..
Android Malware Evolution

- **Android Was Born**
  - 9/2008

- **Fake Player**
  - 8/2010
  - First SMS Trojan
  - Just asks for SEND_SMS permission

- **DroidDream**
  - 3/2011
  - Uses root exploits
  - Installs secondary APK
  - 50 variants in app store

- **Spy Eye**
  - 3/2011
  - Banking malware

- **Spitmo – Zeus goes mobile**
  - 3/2011

- **Obad – The most sophisticated Android trojan**
  - 6/2013
  - 3 exploits
  - 1 backdoor
  - SMS Trojan

- **Dendroid – Android RAT**
  - 5/2014

---

*blackhat®
EUROPE 2014*
Keylogging – Swapping the Keyboard
Intercepting SMS – Just Ask Politely

```java
if (messageBody.equals("somedata")) {
    Intent i = new Intent(context, Webservice.class); // web service validation
    i.putExtra("messageBody", messageBody)
    startService(i);
}
// Stop it being passed to the main Messaging inbox
abortBroadcast();
```

```
<uses-permission android:name="android.permission.SEND_SMS" android:required="true"/>
<uses-permission android:name="android.permission.WRITE_SETTINGS" android:required="false"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE" android:required="false"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" android:required="true"/>
<uses-permission android:name="android.permission.CAMERA" android:required="true"/>
<uses-permission android:name="android.permission.RECORD_AUDIO" android:required="false"/>
<uses-permission android:name="android.permission.PROCESS_OUTGOGING_CALLS" android:required="true"/>
<uses-permission android:name="android.permission.RECEIVE_SMS" android:required="true"/>
<uses-feature android:name="android.hardware.camera" android:required="false"/>
```
Location Tracking – Again Just Ask Politely

```java
@Override
public void onLocationChanged(Location location) {
    // TODO Auto-generated method stub
    int latitude = (int) (location.getLatitude());
    int longitude = (int) (location.getLongitude());
    Log.i("Geo_Location", "Latitude: " + latitude + ", Longitude: " + longitude);
}
```

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" android:required="true" />
<uses-permission android:name="android.permission.READ_CONTACTS" android:required="true" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" android:required="true" />
<uses-permission android:name="android.permission.GET_TASKS" android:required="true" />
<uses-permission android:name="android.permission.WAKE_LOCK" android:required="false" />
```

Easy to Detect
Developer Point-of-View
Android Architecture Basics

• Android is built on top of the Linux kernel
• An application doesn’t talk to hardware
• Talking to the system – only via IPC
The Sandbox

• Each app runs with its own uid

• Privileges are given upon app installation

• Each privilege translates into a gid
What is Binder?
Return of the Microkernel

• Minimalist kernel, less attack surface
• Monolithic kernels won the war
• How to get the benefits of a microkernel anyway?
IPC is the Key

- Isolate the kernel from user apps
- Implement system servers in userland
- Control all communication via Binder
A Sample Transaction

write_buffer

binder_write_read

code

binder_transaction_data

protocol tag*

strlen

Parcel

26 android.media.IMedia
Player 1.0 1.0 interface descriptor

Function code 16: setVolume

1 float leftVolume

2 float rightVolume
Why Target Binder?

• Stealthy, difficult to detect

• Portable data interception

• Integration with the system architecture
Ready for Some Fun?
First Attack: Keylogger
Keyloggers, the Binder Way

- A thread in an app sets up a listener
- It is contacted by the InputContext interface when the user hits a key
- All communication is done via Binder
Keylogging Demo
Second Attack: Data Grabbing
The Secret About Activities

• Most secure applications protect their data

• However, developers don’t bother to encrypt data moving between in-app Activities

• Surprise: This data goes through Binder
Yes, in-app data goes through Binder
...and we got the hex dump to prove it
Third Attack: Intercepting SMS
What Happens When You Get An SMS?

• The Telephony Manager notifies the SMS app

• The app queries the TM’s database

• The response is sent back as a Cursor object

• …but that’s just a file descriptor!
Let's Grab It!
SMS Interception Demo
How Do I Protect Myself?

• Do as much as you can in-app
• Audit your app to see what goes to IPC
• If it goes through Binder, encrypt it
Questions?