Would you mind forking this process?

A DoS attack on Android
(and some countermeasures)

Alessandro Armando
Mauro Migliardi

Alessio Merlo
Luca Verderame

IFIP SEC2012
Android

- OS for mobile devices based on Linux kernel;
- More than 45% of US sales of smartphones in 3Q2011;
- Several works cover Android security mechanism but not stress layer interaction.

Our research covers layer interaction security, leads to the discovery of a new vulnerability and the implementation of two fixes to solve it.
Android architecture

- Android relies on Linux kernel;
“A central design point of the Android security architecture is that no application, by default, has permission to perform any operations that would adversely impact other applications, the operating system, or the user.”

Android Development guide

Achieved through:

- Exploiting Linux process isolation;

- Different Linux user per application;

- Checks applied to IPC invocations.
Inter Process Communication

- **Unix domain socket**: local socket used for exchanging data among processes.

  *Used for message based communication*

- **Binder**: remote procedure call mechanism designed for high performance when performing in-process and cross-process calls.

  *Used for complex data structure exchange*
Each operation in Android involves interaction among layers (**vertical interaction**);

Android proposes three interaction schemes which involve all layers.

**Example:** an application requesting GPS location.
Interaction analysis: Launch of an application

✔ Most common action on Android devices;

✔ Analysis focused on the creation of a new Linux process for the starting application
Process creation: flow

Phase 1.
User taps on application icon.

Phase 2.
System server connects to zygote socket and sends a fork request.

Phase 3.
Zygote process parses the request and executes system call fork creating a process with new Linux ID.

Phase 4.
Zygote process notifies Android about creation.

Phase 5.
ActivityThread attaches starting application to the process.
The Zygote process

✓ Is the only component allowed to request new processes
✓ Runs with root privilege;
✓ Special process with a default Dalvik VM inside;
✓ Several security policies are applied on requests.
zygote socket

```
service zygote /system/bin/app_process -Xzygote /
   system/bin --zygote --start-system-server
   socket zygote stream 666
   onrestart write /sys/android_power/
      request_state wake
   onrestart write /sys/power/state on
```

Linux permission 666 (rw-rw-rw).
Every process can read and write on the socket!
The vulnerability

- Android is not aware of process creation since is no more involved;
- Linux receives legal requests from Zygote (which is root).

Format of request

```bash
--runtime-init [--enable-saf [--enable-checkjni] [--enabl
--nice-name=dummy
com.android.internal.util.WithFramework
```

The dummy process newly created can live even without application attached.
Exploiting the vulnerability

- A non root component can force a root user (e.g. Zygote process) to create an arbitrary number of processes.
- Clearly affects all the other applications.

Fork bomb attack

Denial of service attack.
Implementing the exploit

1. **No Android permissions** are needed.
2. Every Android distribution (even 4.0.3) is affected.
3. Works with both standard and rooted devices.

IFIP SEC 2012  Would you mind forking this process? A DoS attack on Android
Vulnerability testing

Testing with actual devices.

- Devices freeze, crash and reboot.
- Collapse time depends on physical resources.

<table>
<thead>
<tr>
<th>Device Model</th>
<th>Android Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lg Optimus One p550</td>
<td>2.2.3, 2.2.4 (stock LG), 2.2.4 (rooted)</td>
</tr>
<tr>
<td>Samsung Galaxy S</td>
<td>2.3.4 (stock Samsung), 2.3.7 (rooted Cyanogen 7)</td>
</tr>
<tr>
<td>Samsung Next GT-S5570</td>
<td>2.3.4 (stock Samsung), 2.3.4 (rooted)</td>
</tr>
<tr>
<td>Samsung Galaxy Tab 7.1</td>
<td>3.1 (stock Samsung), 3.1 (rooted)</td>
</tr>
<tr>
<td>HTC Desire HD</td>
<td>2.3.2 (stock HTC), 2.3.2 (rooted)</td>
</tr>
</tbody>
</table>

Testing with emulated device.

- Test on 4.0 and 4.0.3 versions of Android;
- Number of dummy processes greatly overcome hardware capability of many available devices.
Vulnerability: fixes

**Requirements:**

1. System server must freely access to zygote socket;
2. Zygote process needs root privileges.

**Fix 1:** Zygote process patch.

**Fix 2:** zygote socket patch.
Zygote process patch

- Changes the Zygote process implementation;
- Zygote process only accepts requests from System server or root;

- Modification at Android layers.

```java
1   boolean runOnce() throws ZygoteInit.
       MethodAndArgsCaller {
2     ...
3     applyUidSecurityPolicy(parsedArgs, peer);
4     applyDebuggerSecurityPolicy(parsedArgs);
5     applyRlimitSecurityPolicy(parsedArgs, peer);
6     applyCapabilitiesSecurityPolicy(parsedArgs, peer);
7     applyForkSecurityPolicy(peer)
```

IFIP SEC 2012 Would you mind forking this process? A DoS attack on Android
**zygote socket patch**

1. Creation and setting of a new owner for the socket;
2. Changing of socket permissions (from 666 to 660);
3. Adding of socket owner group to System server’s group list.

- Involves modifications both at Android layer and Linux layer.

```bash
service zygote /system/bin/app_process -Xzygote /system/bin --zygote --start-system-server
socket zygote stream 660 zygote_socket zygote_socket
...```

IFIP SEC 2012 Would you mind forking this process? A DoS attack on Android
Recognized the vulnerability; ✔️

Opted for the second fix. ✔️

Patch refinement:

change socket owner group from root to system

```
service zygote /system/bin/app_process -Xzygote /
   system/bin --zygote --start-system-server
   socket zygote stream 660 root system
   ...
```

This patch will be included in future versions of Android OS.
Conclusions and future work

✓ Presentation of a previously undisclosed vulnerability in Android OS.

✓ Developing and testing of two possible solutions to solve the problem.

Future work:

➢ Android security model;

➢ Automatic analysis and detection of vulnerabilities.