Android Security & Secure Meta-Markets

Alessandro Armando

(joint work with G. Costa, A. Merlo, and L. Verderame)

DIBRIS, U. of Genova & Security and Trust Research Unit, FBK, Trento

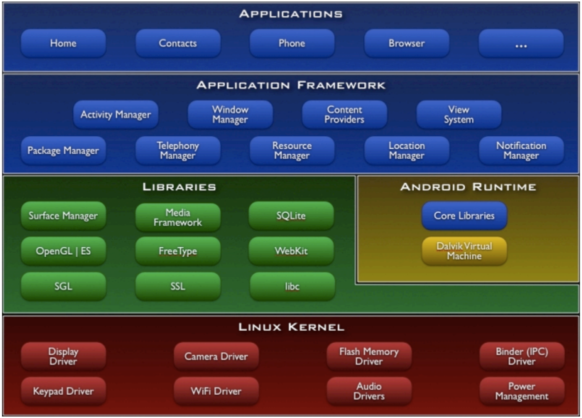
NeSSoS 2013, Sept. 05, 2013

Research Threads on Mobile Security at U. of Genova

1. Security Assessment of Android Cross-layer Architecture

2. BYODroid: a Secure Meta-Market for BYOD Policies

Security Assessment of Android Cross-layer Architecture

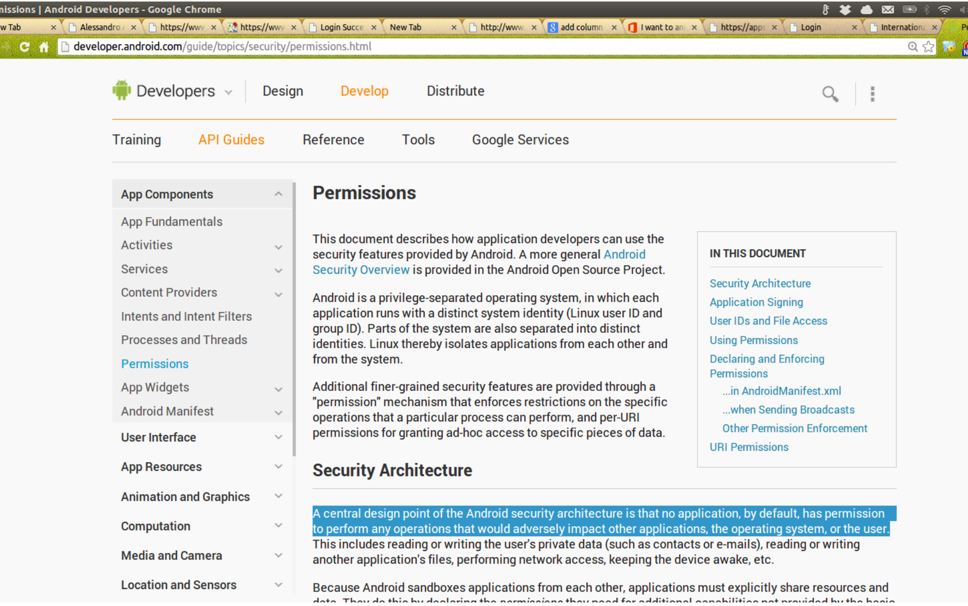


- Java stack built on top of Linux Kernel
- Combination of well-known security solutions (sandboxing + Linux DAC)

Why bothering?

- Android security is a hot topic. Yet,
  - most work has been focusing on the Application Framework (permissions exploitation, IPC, privilege escalation,. . .)
  - little/no work on the Android architecture as a whole.
  - Kernel assumed secure.
- Android stack and Linux Kernel rely on different security models (namely Android Permissions and Linux DAC).
  Are they smoothly integrated?
- Interactions between layers not documented and poorly understood.
- Android sandboxing leads to non-standard use of Linux Kernel.

Android Design Principle



nissions | Android Developers - Google Chrome

developer.android.com/guide/topics/security/permissions.html

Developers | Design | Develop | Distribute

Training | API Guides | Reference | Tools | Google Services

- App Components
  - App Fundamentals
  - Activities
  - Services
  - Content Providers
  - Intents and Intent Filters
  - Processes and Threads
  - Permissions
  - App Widgets
  - Android Manifest
- User Interface
- App Resources
- Animation and Graphics
- Computation
- Media and Camera
- Location and Sensors

Permissions

This document describes how application developers can use the security features provided by Android. A more general Android Security Overview is provided in the Android Open Source Project.

Android is a privilege-separated operating system, in which each application runs with a distinct system identity (Linux user ID and group ID). Parts of the system are also separated into distinct identities. Linux thereby isolates applications from each other and from the system.

Additional finer-grained security features are provided through a "permission" mechanism that enforces restrictions on the specific operations that a particular process can perform, and per-URI permissions for granting ad-hoc access to specific pieces of data.

IN THIS DOCUMENT

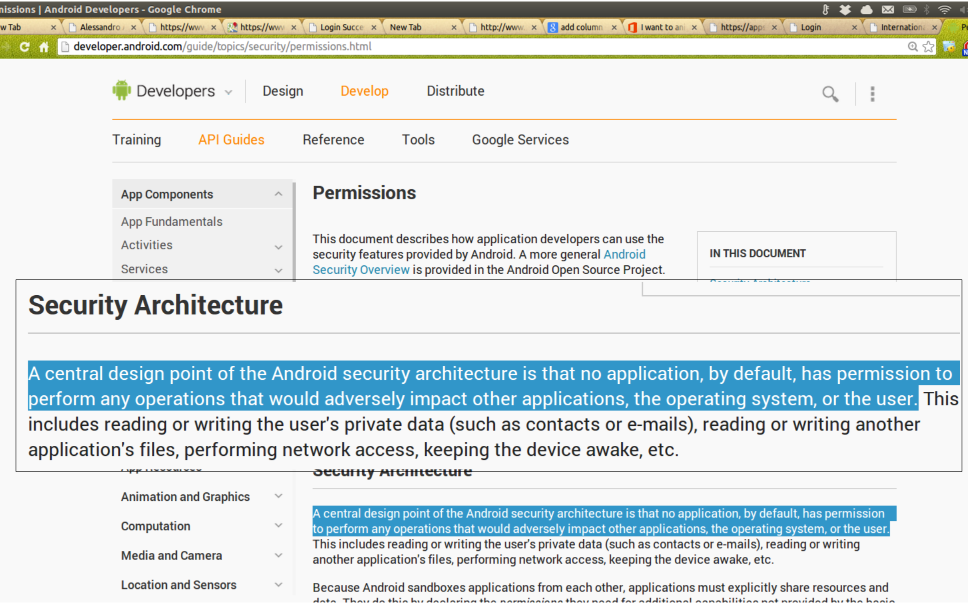
- Security Architecture
- Application Signing
- User IDs and File Access
- Using Permissions
- Declaring and Enforcing Permissions
  - ...in AndroidManifest.xml
  - ...when Sending Broadcasts
  - Other Permission Enforcement
- URI Permissions

Security Architecture

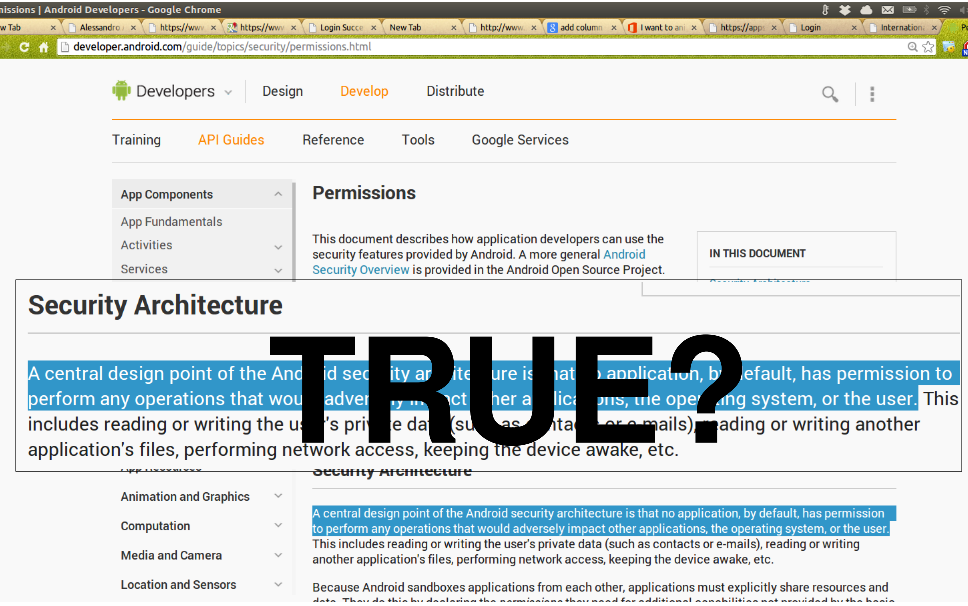
A central design point of the Android security architecture is that no application, by default, has permission to perform any operations that would adversely impact other applications, the operating system, or the user. This includes reading or writing the user's private data (such as contacts or e-mails), reading or writing another application's files, performing network access, keeping the device awake, etc.

Because Android sandboxes applications from each other, applications must explicitly share resources and

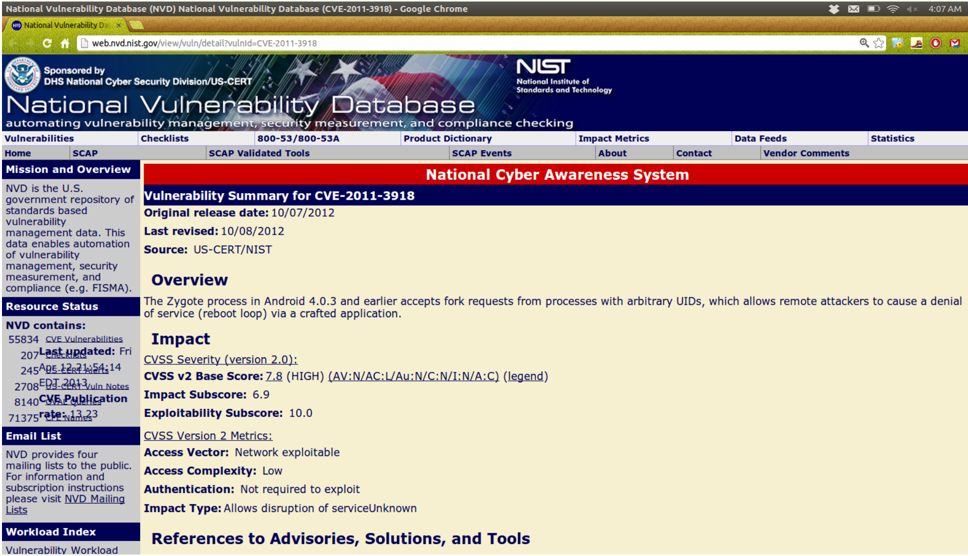
Android Design Principle



Android Design Principle



A Fork Bomb Attack



National Vulnerability Database (NVD) National Vulnerability Database (CVE-2011-3918) - Google Chrome

web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2011-3918

Sponsored by
DHS National Cyber Security Division/US-CERT

NIST National Institute of Standards and Technology

National Vulnerability Database
automating vulnerability management, security measurement, and compliance checking

Vulnerabilities | Checklists | 800-53/800-53A | Product Dictionary | Impact Metrics | Data Feeds | Statistics
Home | SCAP | SCAP Validated Tools | SCAP Events | About | Contact | Vendor Comments

Mission and Overview

NVD is the U.S. government repository of standards based vulnerability management data. This data enables automation of vulnerability management, security measurement, and compliance (e.g. FISMA).

Resource Status

NVD contains:
55834 CVE Vulnerabilities
207 Checklists
245 US-CERT Alerts
2708 US-CERT Vuln Notes
8140 OVAL Queries
71375 CPE Names

Last updated: Fri Apr 12 21:54:14 EDT 2013
CVE Publication rate: 13.23

Email List

NVD provides four mailing lists to the public. For information and subscription instructions please visit NVD Mailing Lists

Workload Index

Vulnerability Workload

National Cyber Awareness System

Vulnerability Summary for CVE-2011-3918

Original release date: 10/07/2012
Last revised: 10/08/2012
Source: US-CERT/NIST

Overview

The Zygote process in Android 4.0.3 and earlier accepts fork requests from processes with arbitrary UIDs, which allows remote attackers to cause a denial of service (reboot loop) via a crafted application.

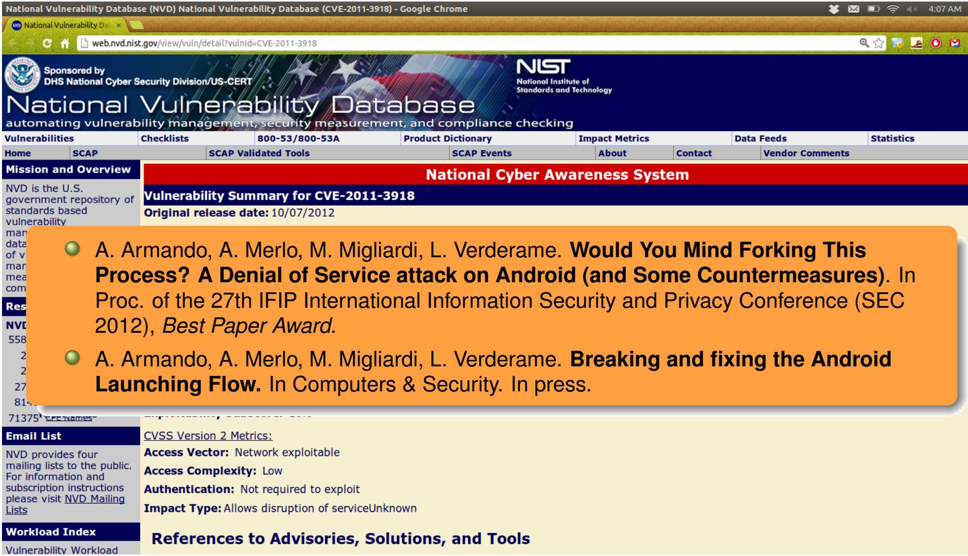
Impact

CVSS Severity (version 2.0):
CVSS v2 Base Score: 7.8 (HIGH) (AV:N/AC:L/Au:N/C:N/I:N/A:C) (legend)
Impact Subscore: 6.9
Exploitability Subscore: 10.0

CVSS Version 2 Metrics:
Access Vector: Network exploitable
Access Complexity: Low
Authentication: Not required to exploit
Impact Type: Allows disruption of serviceUnknown

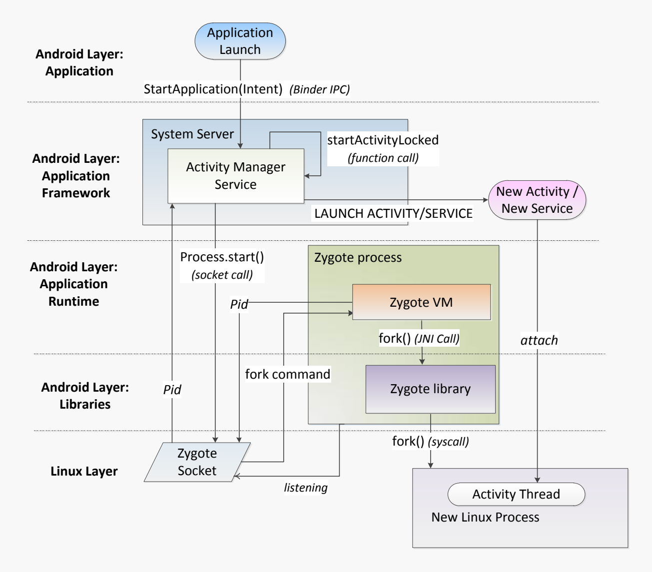
References to Advisories, Solutions, and Tools

A Fork Bomb Attack

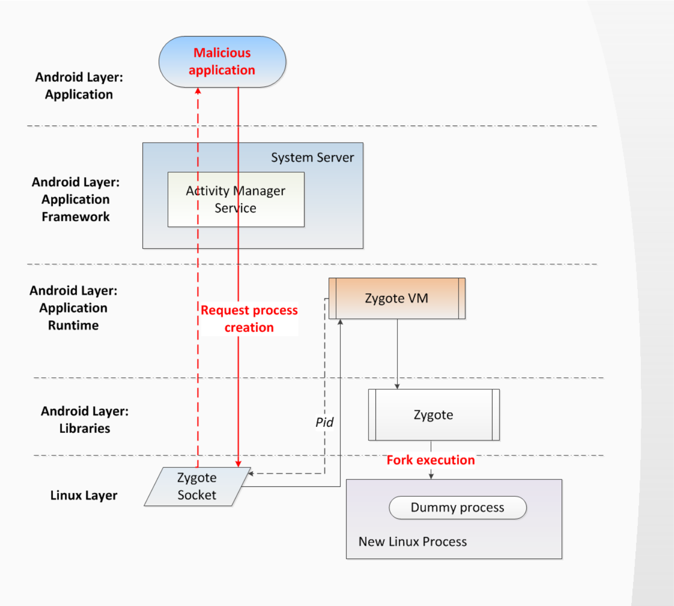


- A. Armando, A. Merlo, M. Migliardi, L. Verderame. **Would You Mind Forking This Process? A Denial of Service attack on Android (and Some Countermeasures)**. In Proc. of the 27th IFIP International Information Security and Privacy Conference (SEC 2012), *Best Paper Award*.
- A. Armando, A. Merlo, M. Migliardi, L. Verderame. **Breaking and fixing the Android Launching Flow.** In Computers & Security. In press.

Forking in Android



Exploiting the vulnerability



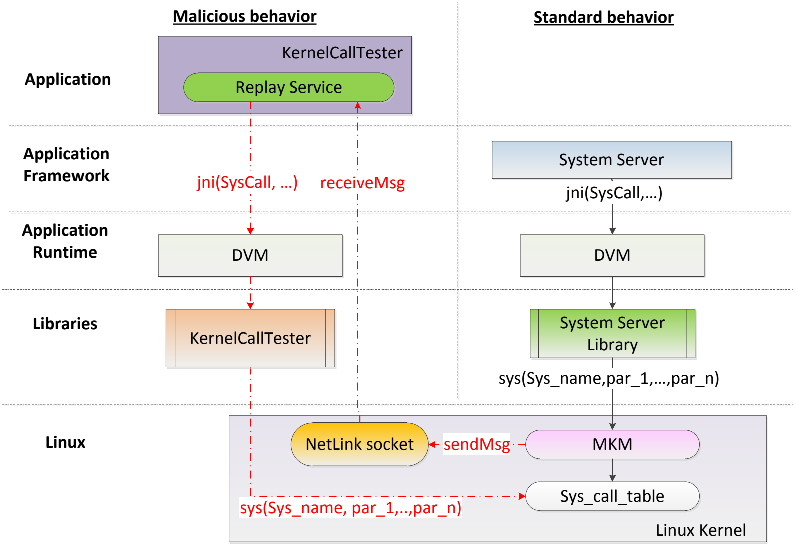
The next step

- **Lesson learned**: ASF does not discriminate the identity of the caller of the fork (i.e. malicious application vs. trusted service in the AF).
- Some questions arise:
  1. Is the problem related to the fork syscall only?
  2. Are applications able to directly execute Kernel calls?
  3. Is it acceptable from a security point of view?
- and, above all,

Are there other cross-layer vulnerabilities?

Empirical Assessment of Kernel Call Invocation

1. Relate kernel calls with trusted services in the AF through experimentation ⇒ Monotoring Kernel Module (MKM)
2. Try to reproduce the very same kernel calls from a malicious unprivileged application ⇒ Kernel Call Tester (KCT)
3. Check whether replicated kernel calls have been executed successfully.
4. Automatically analyze logs to search for vulnerabilities and malicious ”flows” of kernel calls.


Malicious behavior
Standard behavior
Application
KernelCallTester
Replay Service
Application Framework
jni(SysCall, ...)
receiveMsg
System Server
jni(SysCall,...)
Application Runtime
DVM
DVM
Libraries
KernelCallTester
System Server Library
sys(Sys_name,par_1,...,par_n)
Linux
NetLink socket
sendMsg
MKM
Sys_call_table
sys(Sys_name, par_1,..,par_n)
Linux Kernel

Results

- The ASF does not discriminate the caller of any direct kernel call.
- Two new vulnerabilities pave the way to:
  1. **Denial-of-Service** attack that exhausts memory.
  2. **Privacy Leakage** attack of browser data.
- The new vulnerabilities affect *all* Android builds.

A. Armando, A. Merlo and L. Verderame. **An Empirical Evaluation of the Android Security Framework.** In Proc. of the 28th IFIP International Information Security and Privacy Conference (SEC 2013).

Future Work

- Further, finer-grained analysis of MKM logs needed to discover other interplay-related vulnerabilities (if any).
- Extend approach to other cross-layer calls.
- Leverage *profiling technology* (e.g. MKM) for run-time monitoring and/or anomaly detection.

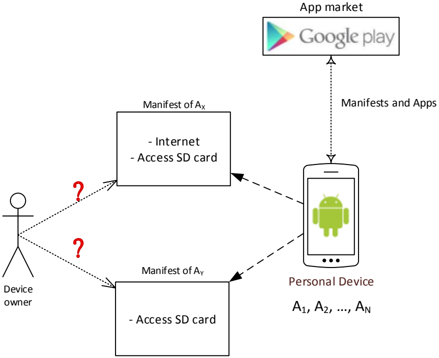
Research Threads on Mobile Security at U. of Genova

1. Security Assessment of Android Cross-layer Architecture
2. BYODroid: a Secure Meta-Market for BYOD Policies

The BYOD Paradigm

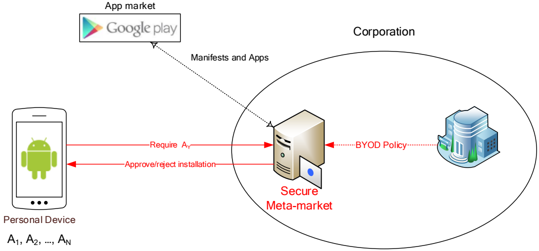
- The Bring Your Own Device paradigm strives to bring usage of personal devices inside organizations.
- BYOD solutions must
  1. allow users to freely personalize devices outside the organization
  2. ensure security of corporate data accessed by personal devices.
- Existing mobile OSes do not support the latter.

Android, Security and Users



- Android applications come up with a manifest file, containing required permissions.
- Users must accept at install time all the required permissions.
- Do users understand both the meaning and the impact of such permissions on their security/privacy?

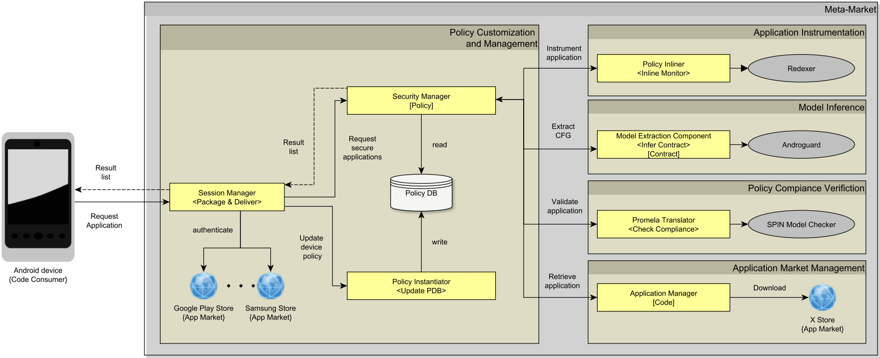
BYODroid: a Secure Meta-Market for BYOD



- BYODroid allows for
  - definition and enforcement of security policies
  - spanning all the applications installed on the device.
- BYODroid supports
  - retrieval and automatic security analysis of applications
  - from different, possibly untrusted, sources,
  - while ensuring that the installed applications collectively comply with a global security policy.
- This is achieved by a fruitful combination of static analysis, model checking, and code instrumentation.

Anatomy of BYODroid

- Model Extraction (Androguard)
- Policy Compliance Verification (SPIN)
- Policy Customization and Storage (Partial Model Checking)
- Application Instrumentation and Monitoring (Redexer)

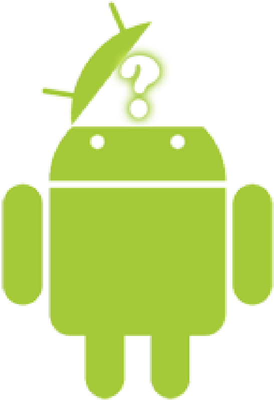


Experimental Assessment (Excerpt)

Application	Size (Mb)	T_{ext}	Nodes	Edges	T_{enc}	T_{mc}	Valid	T_{ins}	Growth
Google Maps	6.6	226569	83	373	3890	390	YES	55647	0.81
Facebook	15.8	24701	26	108	517	367	NO (61)	5653.22	< 0.01
WhatsApp	10.2	388815	200	670	9637	359	YES	68363	< 0.01
Angry Birds	35.5	197718	232	807	13008	63854	Time Out	24627	0.14
Skype	15.5	54827	82	277	1863	381	NO (62)	42974	0.18
Adobe Reader	7.0	14236	44	158	857	405	NO (63)	8985	0.45
FB Messenger	12.6	145436	112	449	4859	439	NO (67)	52979	< 0.01
Gmail	3.7	6.5	98	381	3624	482	YES	32093	1.14
Fruit Ninja	19.2	69343	120	420	3825	989	NO (129)	17655	< 0.01
Google Street View	.3	2875	13	54	214	364	YES	1035	2.01
Tiny Flashlight	1.3	61366	112	374	2927	403	YES	6896	0.94
Instagram	15.6	47917	56	223	1566	482	NO (199)	25834	< 0.01
GO Launcher	.3	189	0	0	3	366	YES	57	1.51
Angry Birds Seasons	44.3	190770	251	837	13220	511	NO (73)	28959	0.11
Angry Birds Rio	34.2	189835	232	807	13066	64503	Time Out	24920	0.14
Dropbox	5.9	.03	79	295	2254	441	YES	15121	0.45
LinkedIn	6.9	.1	170	626	9612	383	YES	54105	0.43
Amazon Kindle	22.3	209889	137	493	5736	1486	YES	236886	< 0.01
Spotify	3.9	0.02	49	186	1061	395	YES	5241	0.56
Firefox	24.2	0.04	63	216	1597	462	YES	28592	0.29

More information

- A. Armando, G. Costa, A. Merlo. **Formal Modeling and Verification of the Android Security Framework.** in Proc. of the 7th International Symposium on Trustworthy Global Computing (TGC 2012).
- A. Armando, G. Costa, A. Merlo, L. Verderame. **Securing the "Bring Your Own Device" Policy.** in the Journal of Internet Services and Information Security(JISIS), Vol.2, N. 3, pp. 3-16, Nov. 2012. Best Paper Award at MIST 2012.
- A. Armando, G. Costa, A. Merlo, L. Verderame. **Bring Your Own Device, Securely.** In Proc. of the 28th ACM Symposium on Applied Computing, Security Track (SAC 2013)



QUESTIONS ?!?