A Computational Forensic Methodology for Malicious Application Detection on Android OS

Svetlana Voronkova

BunnyTN3 Workshop, 2012

Outline

- Motivations and objectives
- Suspicious Application Detection Methodology
- The evaluation
- The proposed system prototype
- Conclusions

Digital Forensics

Focuses on:

- extraction
- recovery
- analysis

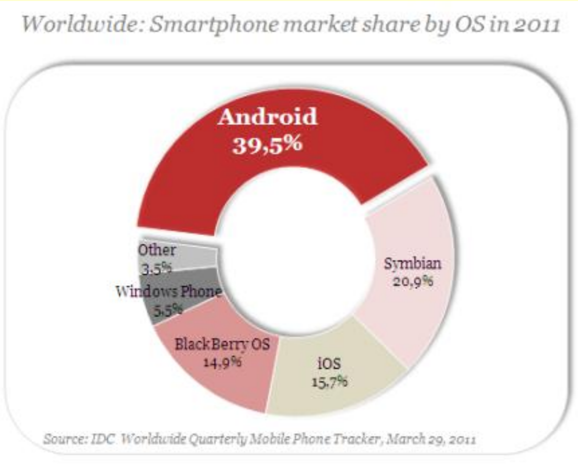
of **information** stored on computers or any other electronic media.

Digital Forensics is used:

to find out exactly what happened on a digital system, and who was responsible for it.

Android OS

- Relatively new and rapidly evolving **OS for smartphones** and tablets created by Google;
- Provides an open development platform;
- Has big and significant community
- Distributes apps via Android Market that supports a **lightweight security policy**;
- Due its popularity it is becoming **very attractive** to malicious software developers.



Worldwide: Smartphone market share by OS in 2011

Source: IDC Worldwide Quarterly Mobile Phone Tracker, March 29, 2011

The Motivation (1)

The **small range** of existing forensic tools that support the analysis of devices running Android OS.

The importance to be able to **reproduce the scene** that happened on the device in the case of it's involvement in the **illegal activities**.

Facts:

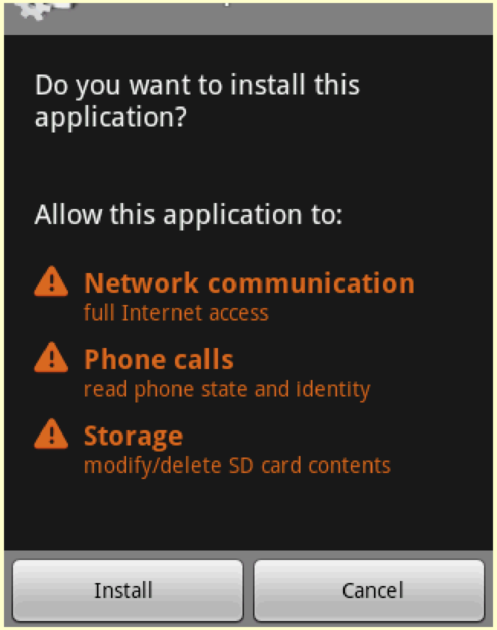
- **2010 -** very small number of forensics tools that explicitly deal with Android OS.
- **March, 2011-** Google confirms that **58** malicious apps were uploaded to Market, with the download onto **260,000** devices.
- **March, 2011-** "DroidDream " detected. It is able install any applications and **execute code with root privileges** .

The Motivation (2)

- Forensics activities on Android were complex for a number of reasons:
  - **Dynamic application analysis** is (at the moment) impossible on Android OS:
    - Applications are executed in a sandbox environment;
    - Any modification to the original state of a device would invalidate any evidence collected.
  - **Static analysis** of Android Application is very difficult:
    - It is impossible to download applications from the Android Market for testing-analysis purposes ;
    - It is necessary to use specific debugging tools and techniques (e.g., ADB).

The Methodology

- Classifies applications based on Android security permissions.



Do you want to install this application?

Allow this application to:

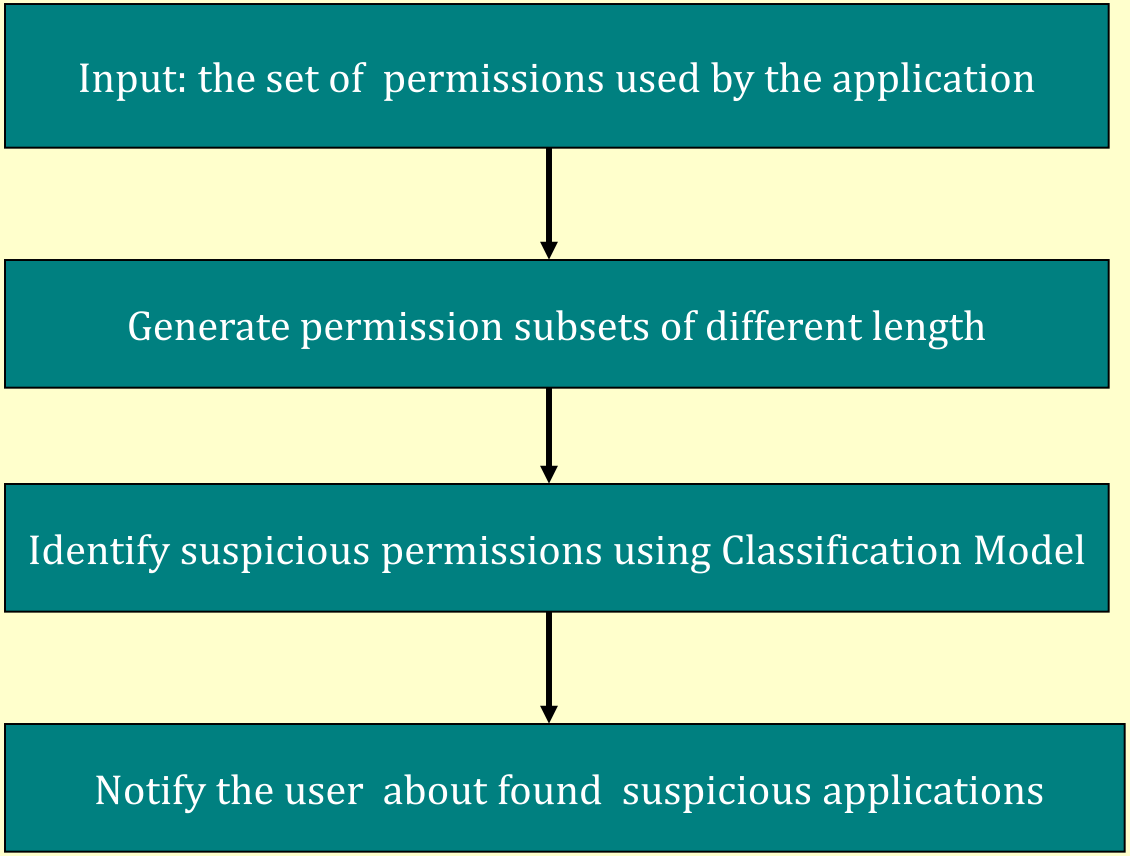
⚠ **Network communication**
full Internet access

⚠ **Phone calls**
read phone state and identity

⚠ **Storage**
modify/delete SD card contents

- The classification model:
- is built from 13000 apps hosted on Android Market and collected by AppAware; all of them are prompted to use user's personal data ;
- classifies Android applications into: *suspicious, possibly suspicious, not suspicious;*

Suspicious App Detection Algorithm



Input: the set of permissions used by the application

Generate permission subsets of different length

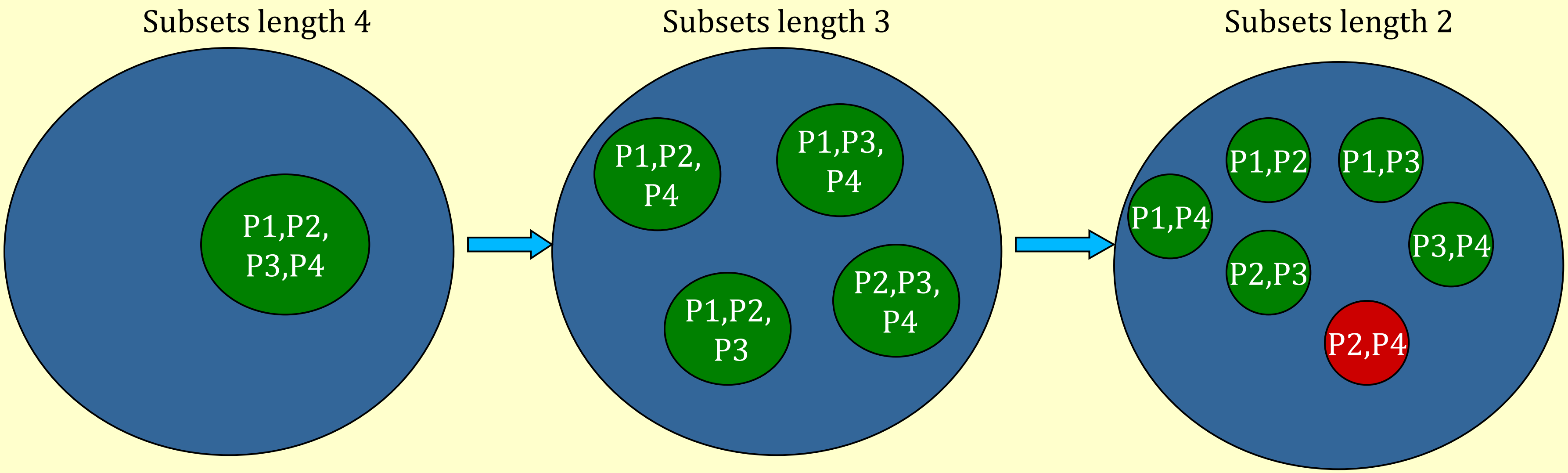
Identify suspicious permissions using Classification Model

Notify the user about found suspicious applications

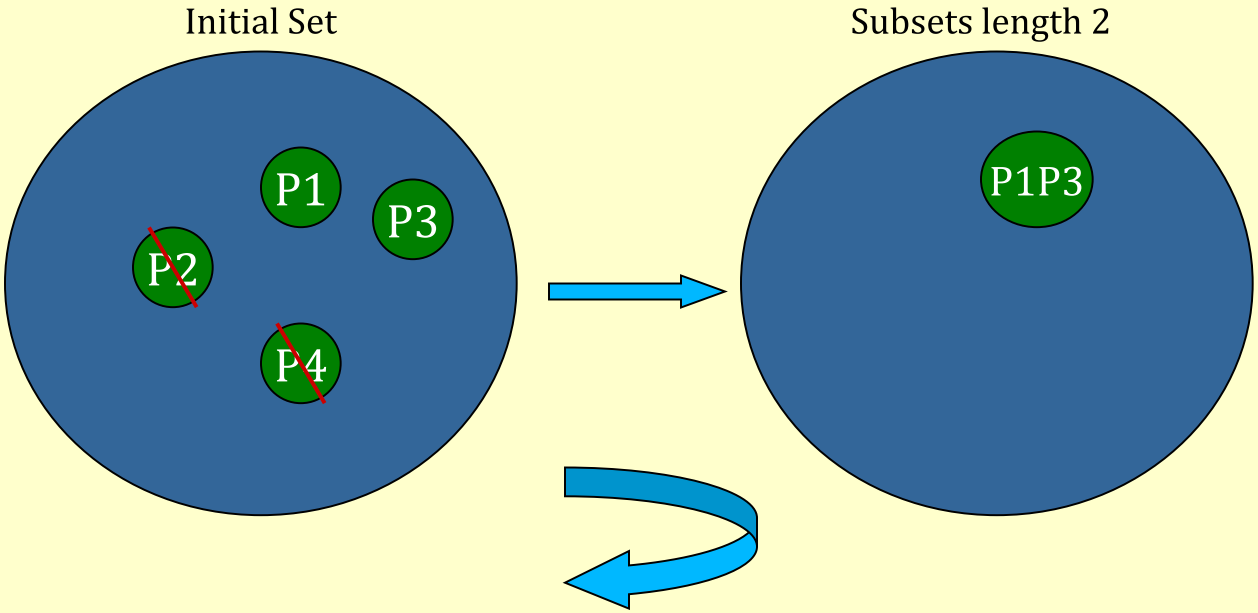
ANSAN Analysis (1)

ANSAN – image editor able to store secret contacts into the images ;

Permissions: write_contacts (P1), camera (P2), call_phone (P3), write_storage (P4).



ANSAN ANALYSIS (2)



Suspicious permissions
P1, P3

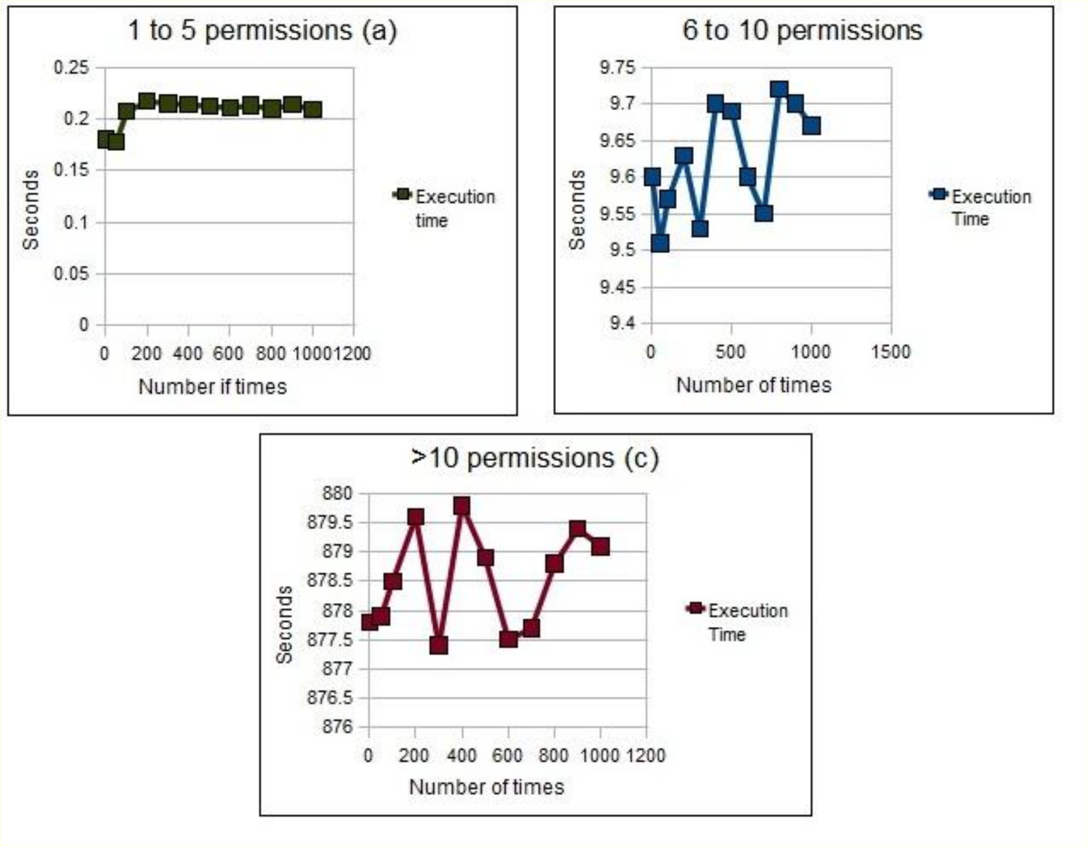
Suspicious permissions
write_contacts, call_phone

Performance Evaluation

Data: 450 apps collected from Android devices ad divided into 3 groups:

- apps with 1-5 permissions;
- apps with 6-10 permissions;
- apps with more than 10

permissions.



Correctness Evaluation

Data: application divided into groups as before and each groups includes additionally:

- 50 apps present in the classification model but renamed;
- 25 artificially created malicious apps, where 10 of them were inserted into classification model and renamed.

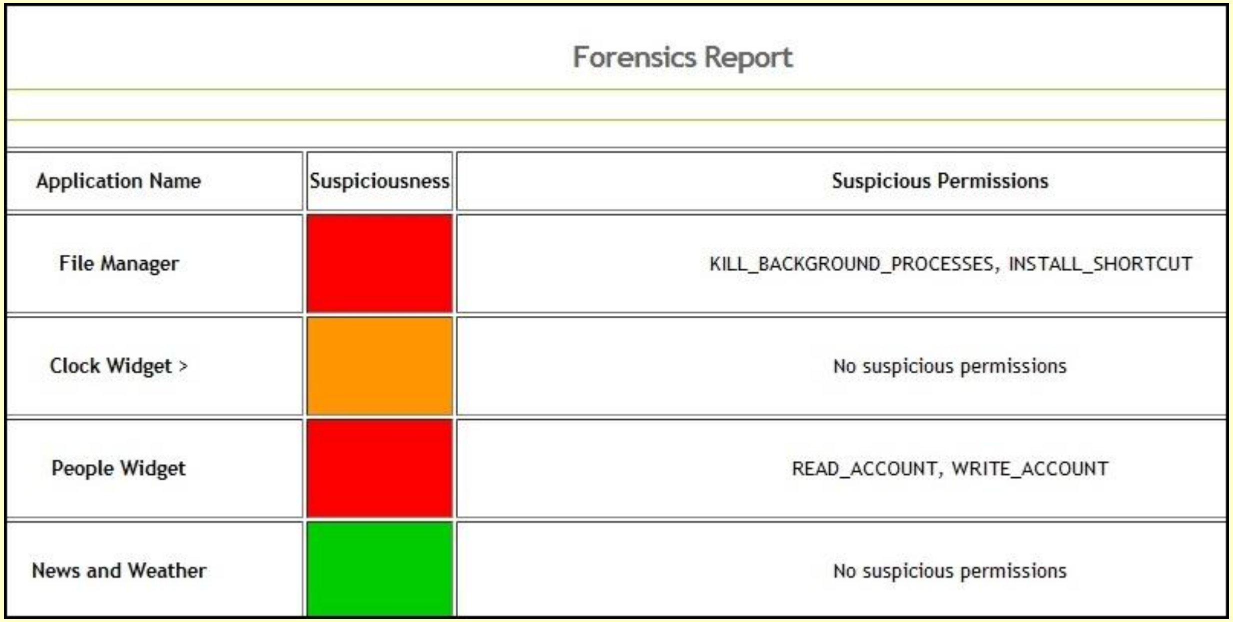
		G1	G2	G3
False Positives	New apps	30%	40%	>50%
	Renamed apps	0%	0%	0%
False Negatives	New apps	0%	0%	0%
	Renamed apps	0%	0%	0%

- The methodology is false positive oriented
- The complexity:$O(2^m)$, m- number of suspicious permissions in the set.

AForensics Suite (1)

AForensics Web:

- Performs analysis of data extracted from Android device
- Creates detailed report about all the Apps stored on the device



Forensics Report

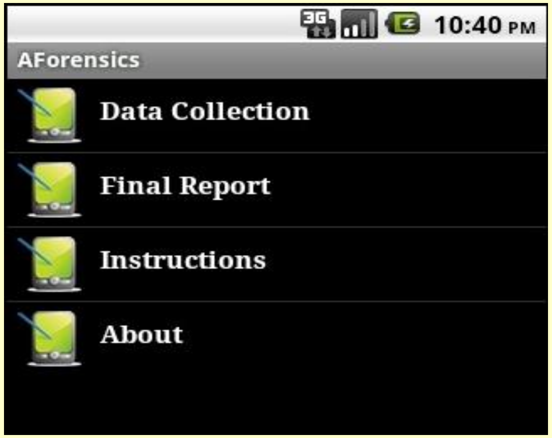
Application Name	Suspiciousness	Suspicious Permissions
File Manager	🟥	KILL_BACKGROUND_PROCESSES, INSTALL_SHORTCUT
Clock Widget >	🟧	No suspicious permissions
People Widget	🟥	READ_ACCOUNT, WRITE_ACCOUNT
News and Weather	🟩	No suspicious permissions

AForensics Suite(2)

AForenscis Android:

- Collects Information about Apps stored on the device
- Performs the analysis of Apps selected by user
- Creates a comprehensive report about suspicious Apps detected
- Collected data can be used for additional forensics analysis conducted with other tools than Aforensics Web

AForensics Suite(3)



10:40 PM

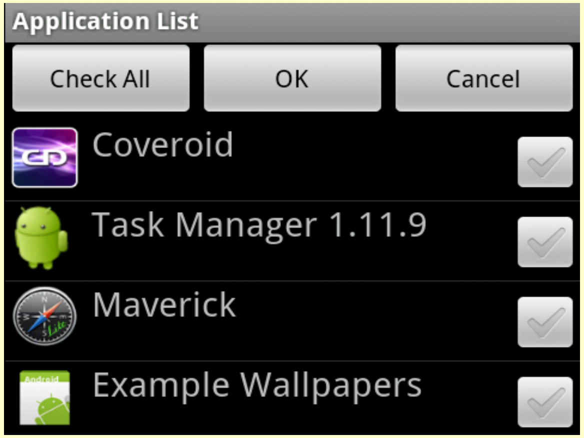
AForensics

Data Collection

Final Report

Instructions

About



Application List

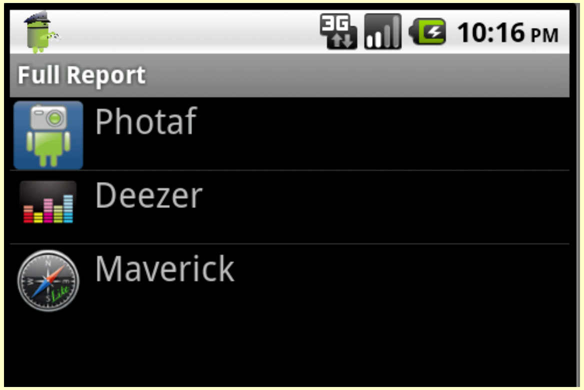
Check All | OK | Cancel

Coveroid ☑

Task Manager 1.11.9 ☑

Maverick ☑

Example Wallpapers ☑



10:16 PM

Full Report

Photaf

Deezer

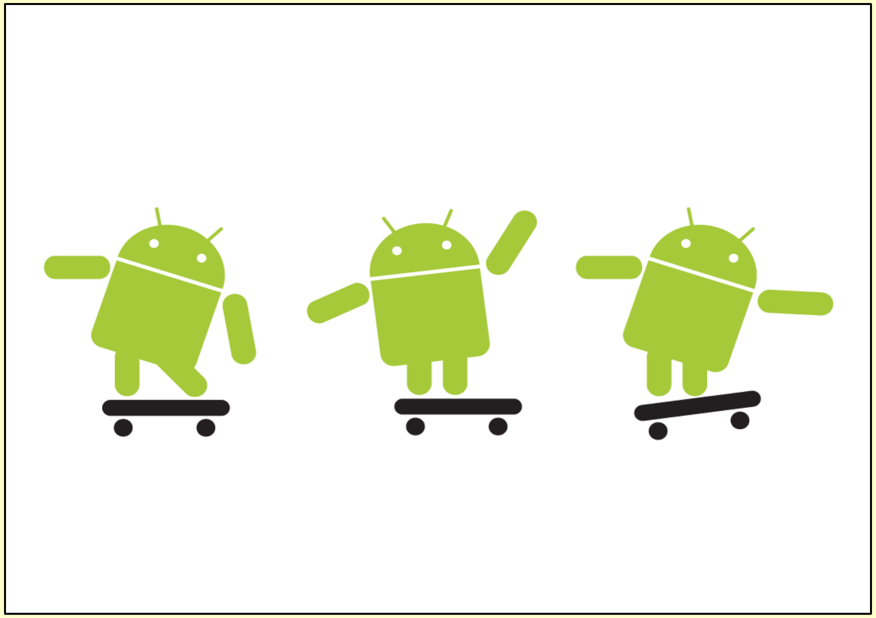
Maverick

Conclusions

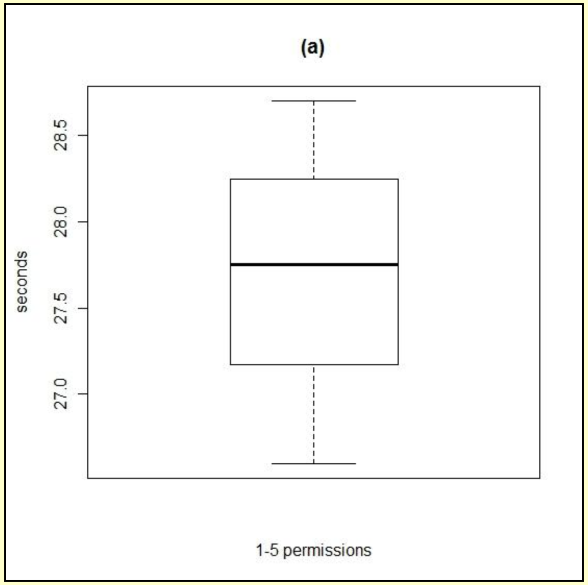
- Definition of Suspicious Application Detection Methodology and implementation on AForensics Suite;
- The evaluation of algorithm's complexity, performance and correctness;
- The publication of the paper and it's presentation in 4th International Workshop on Computational Forensics.

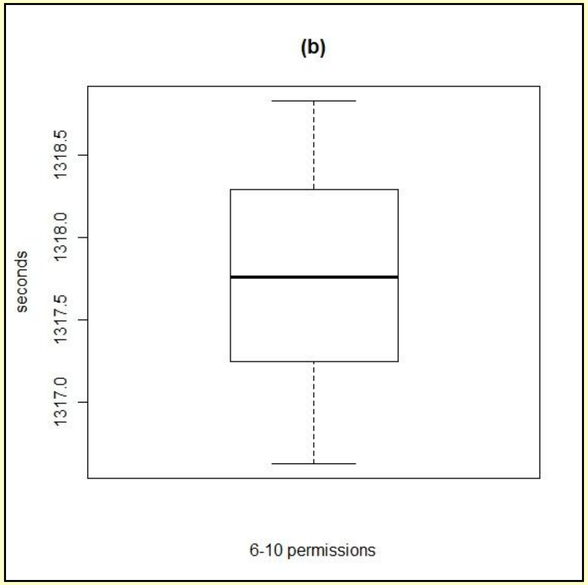
Thank you for your attention!

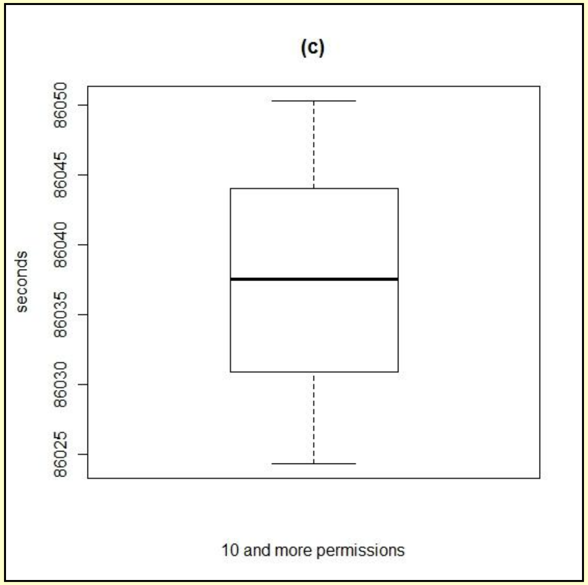
Questions?



Performance Evaluation (2)







The Complexity (2)

- Best case: $$C(N, N-2) + 1 = \frac{N!}{(N-2)!(N-(N-2))!} + 1 = \frac{n(n-1)}{2} + 1$$

- General Case : $$\sum_{k=1}^{m} C(N, N-k)$$

- Worst Case: $$\sum_{1 \geq m \leq N} C(N, N-m) + 1 = 2^N$$