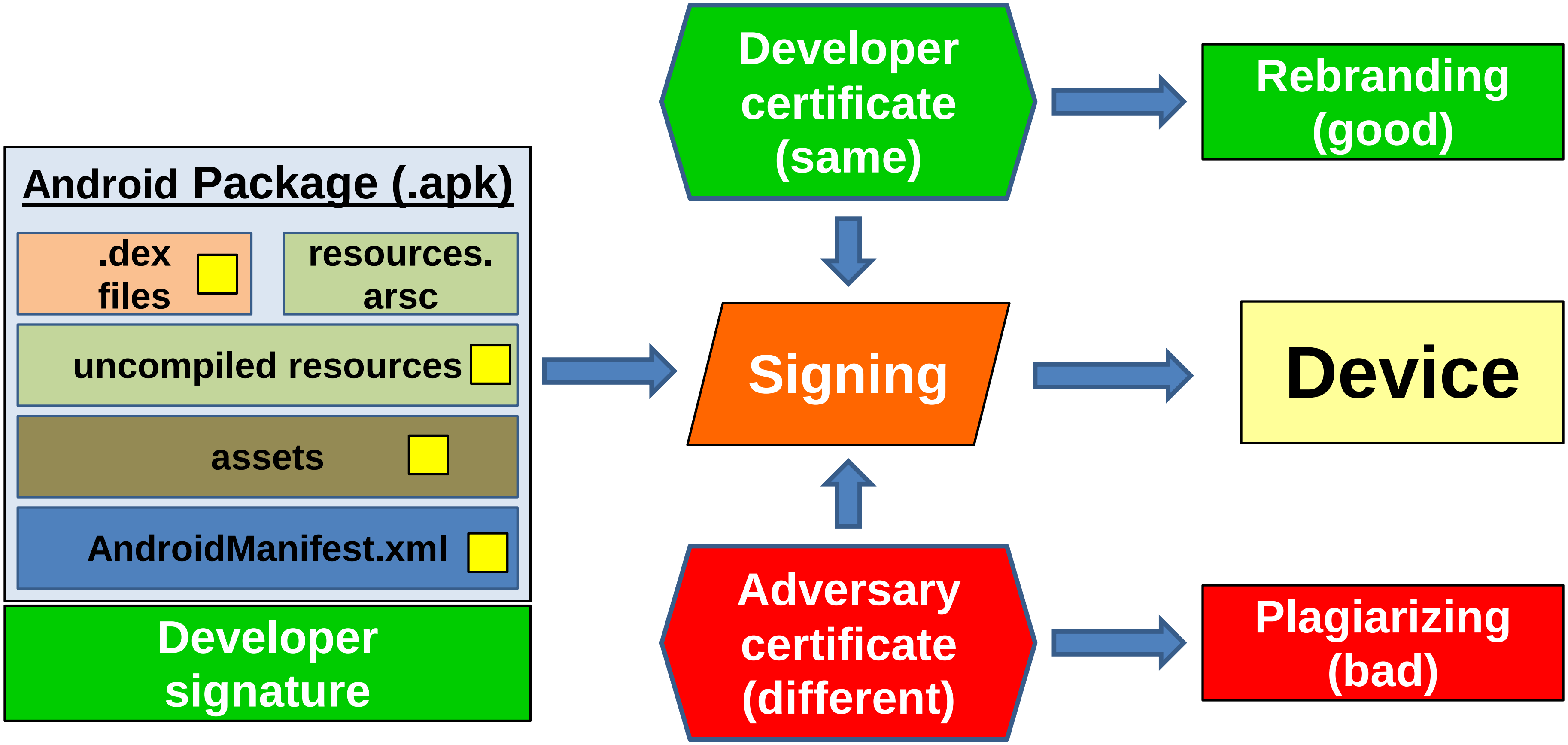
FSquaDRA: Fast Detection of Repackaged Applications

Yury Zhauniarovich, Olga Gadyatskaya, Bruno Crispo, Francesco La Spina, Ermanno Moser

zhauniarovich, gadyatskaya, crispo, laspina, moser@disi.unitn.it

University of Trento

Repackaging



Android Package (.apk)

.dex files

resources.arsc

uncompiled resources

assets

AndroidManifest.xml

Developer signature

Developer certificate (same)

Signing

Adversary certificate (different)

Rebranding (good)

Device

Plagiarizing (bad)

Motivation

- App repackaging is very easy on Android:
  - Fetch an app → Disassemble → Change → Assemble → Sign with own certificate → Publish
- The code of the application can be easily changed
  - smali/backsmali, AndroGuard, dex2jar, apktool, etc.
- **Plagiarizing** is used to:
  - steal advertising revenues (14% of ad revenues)*
  - collect user database (10% of user base)*
  - malware distribution (86% of Android malware samples use this distribution channel)**

*** C.Gibler et al. “Adrob: examining the landscape and impact of Android application plagiarism”. In *Proc. of MobiSys ’13***

**** Y. Zhou, X. Jiang. “Dissecting Android malware: Characterization and Evolution”. In *Proc. of S&P ’12***

Problem Statement

Issue: *How to detect repackaged Android applications*

- fast
  - 1.1+ million apps on Google Play *
  - 190+ third-party markets **
  - quadratic complexity
- in effective way?
  - need for a similarity metric to what extent one app is similar to another

*** N. Viennot et al. "A Measurement Study of Google Play". In *Proc. of SIGMETRICS '14***

**** T. Vidas, N. Christin. "Sweetening Android Lemon Markets: Measuring and Combating Malware in Application Marketplaces". In *Proc. of CODASPY '13***

FSquaDRA: Idea

- Repackaged apps want to maintain the “look and feel” of the originals
  - Opera Mini fake: 230 of 234 files are the same

- **IDEA:** compare apps based on the included resource files (**same files → same apps**)

FSquaDRA: Approach

- Compute hashes of all files inside two apps
- Calculate Jaccard index for the extracted hashes:

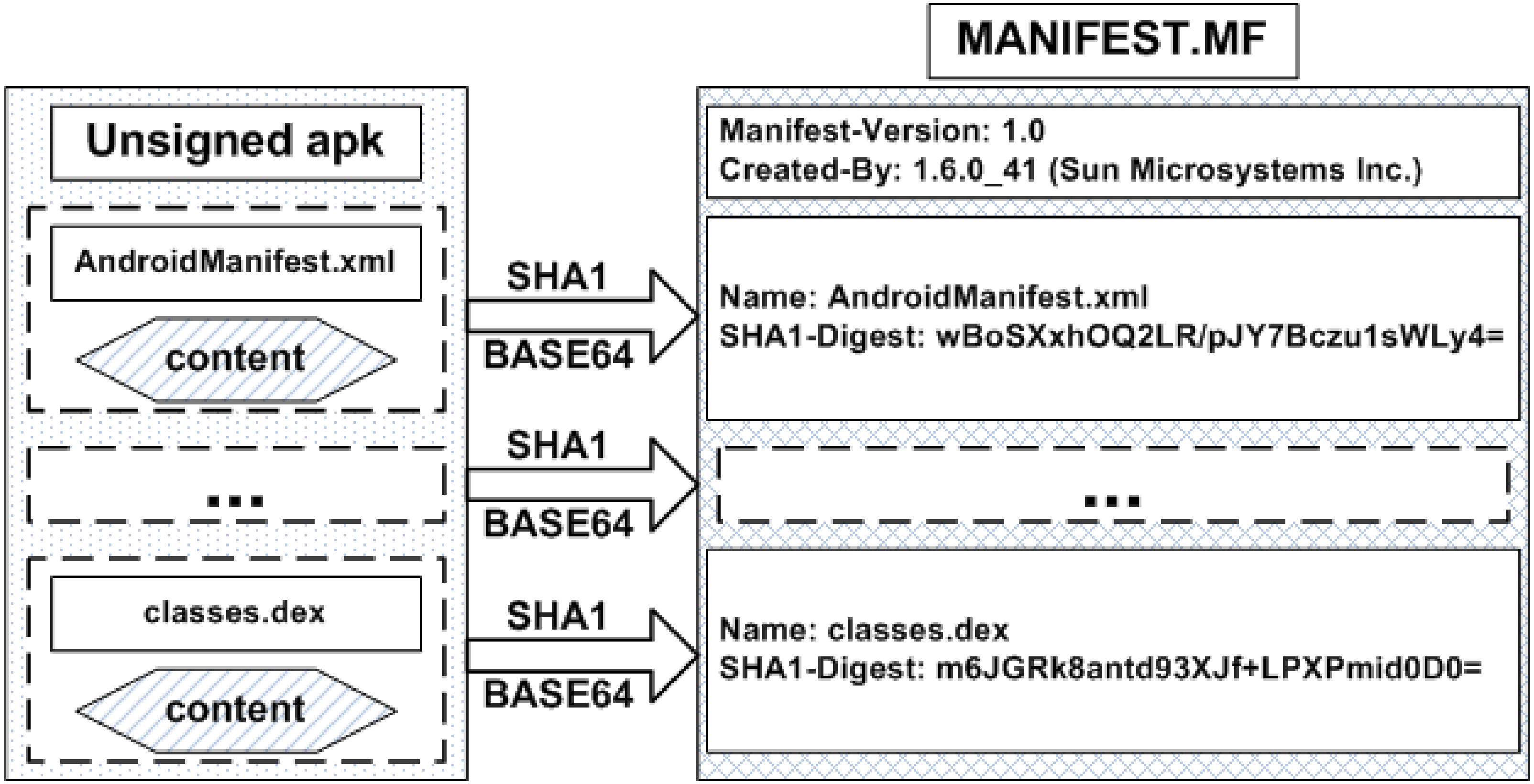
$$jSim(H_k, H_l) = \frac{|H_k \cap H_l|}{|H_k \cup H_l|}$$

H_i – set of hashes of files in apk *i*

- Compare the obtained value with a threshold
- **PROBLEM:** How to compute hashes efficiently?

Speeding Up Hash Calculations

As a part of application signing process SHA1 digest of each file inside apk is calculated



MANIFEST.MF

Manifest-Version: 1.0
Created-By: 1.6.0_41 (Sun Microsystems Inc.)

Name: AndroidManifest.xml
SHA1-Digest: wBoSXxhOQ2LR/pJY7Bczu1sWLy4=

...

Name: classes.dex
SHA1-Digest: m6JGRk8antd93XJf+LPXPmid0D0=

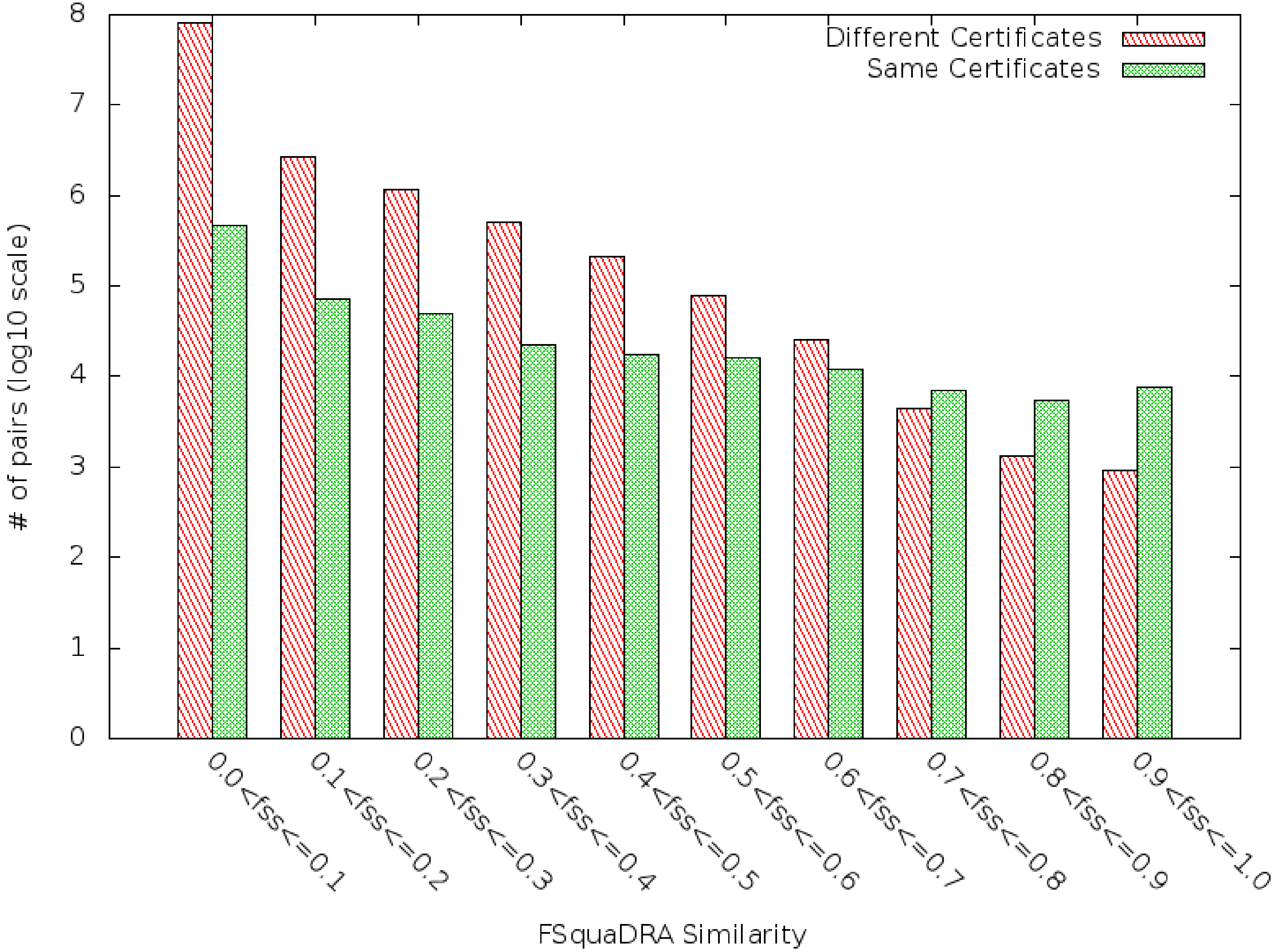
FSquaDRA: Evaluation

- Dataset:
  - 55779 apk samples
  - from 8 markets including Google Play
- Pairwise comparison of all apps in the dataset

- **Objectives:**
  - plagiarizing rates for apps signed with different certificate
  - rebranding rates for apps signed with the same certificate

- Evaluate **Efficiency** and **Effectiveness**

Evaluation: Pairwise Comparison



Evaluation: Efficiency

- FSquaDRA is implemented as a single-threaded Java program
  - not really optimized
- We ran experiments on a commodity laptop (2.9 GHz Intel Core i7, 8GB RAM)
  - **15,10 hours** to load hashes into memory
  - **64,41 hours** to compute similarity score for all app pairs
- On average **6700 app pairs per second**

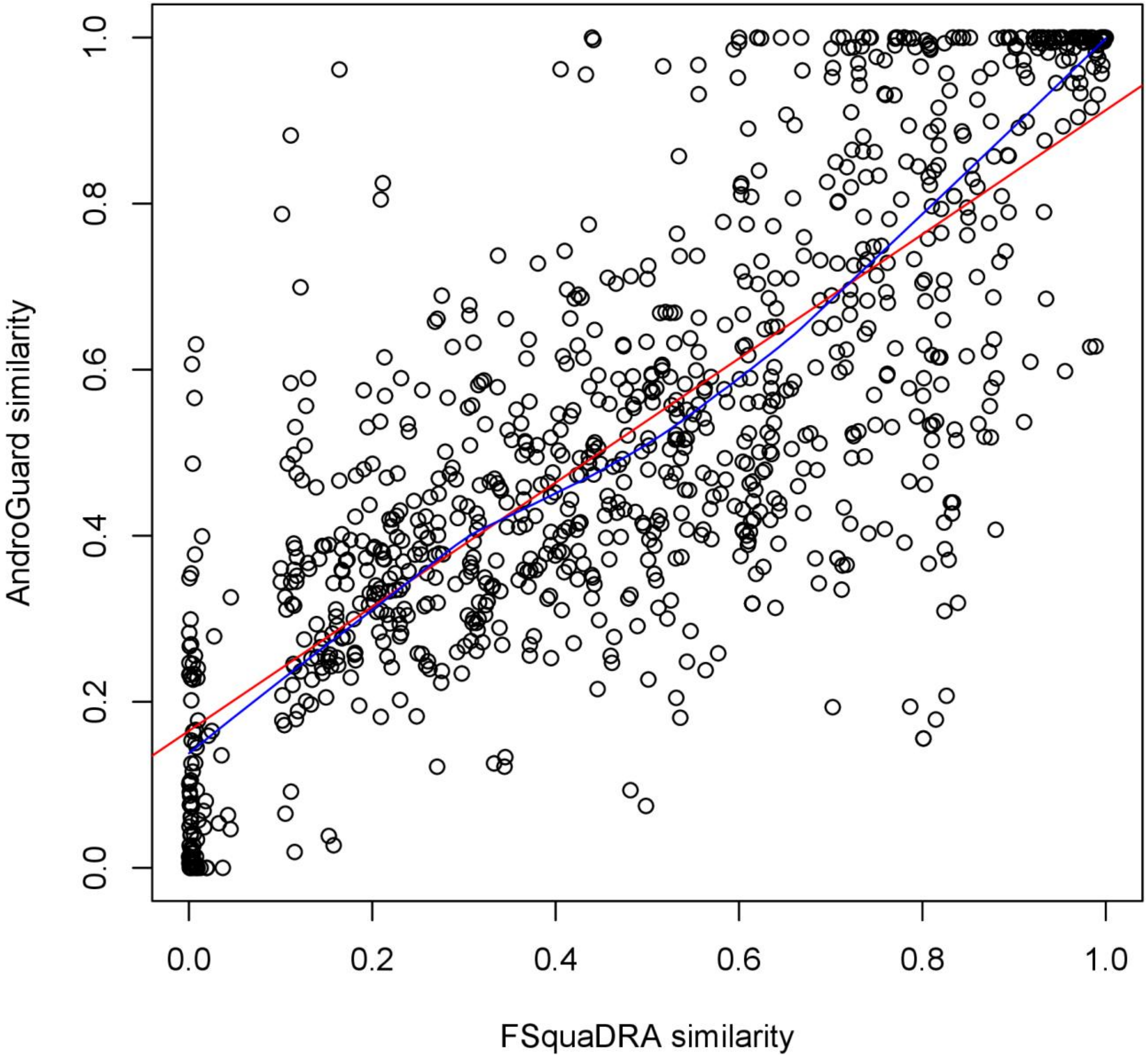
Evaluation: Effectiveness

- **Metrics:**
  - **False Positives?** For apps FSquaDRA considers repackaged, are they actually repackaged?
  - **False Negatives?** For apps FSquaDRA considers different, are they really not repackaged?
- **Approaches:**
  - analyze FSquaDRA on a dataset of repackaged apps
  - compare FSquaDRA metrics with the state-of-the-art tools
- **Problems:**
  - no public dataset with repackaged apps
  - only one public tool: AndroGuard

Effectiveness: Evaluation Setup

- AndroGuard – open-source tool by A. Desnos:
  - computes code-based similarity metric
  - slow (65 sec to compare an app pair on average)
  - does not produce symmetric values
- We use average score of (A,B) and (B,A) as the similarity score for AndroGuard (ags)
- For each selected bin:
  - randomly picked 100 app pairs with different certificates and 100 app pairs with the same certificate;
  - calculated their AndroGuard similarity score (ags)
  - compared with FSquaDRA similarity score (fss)

Effectiveness: Plagiarizing Results (different certificates, fss>0)



Correlation: 0.7919

Difference (fss-ags):

-mean: -0.0412

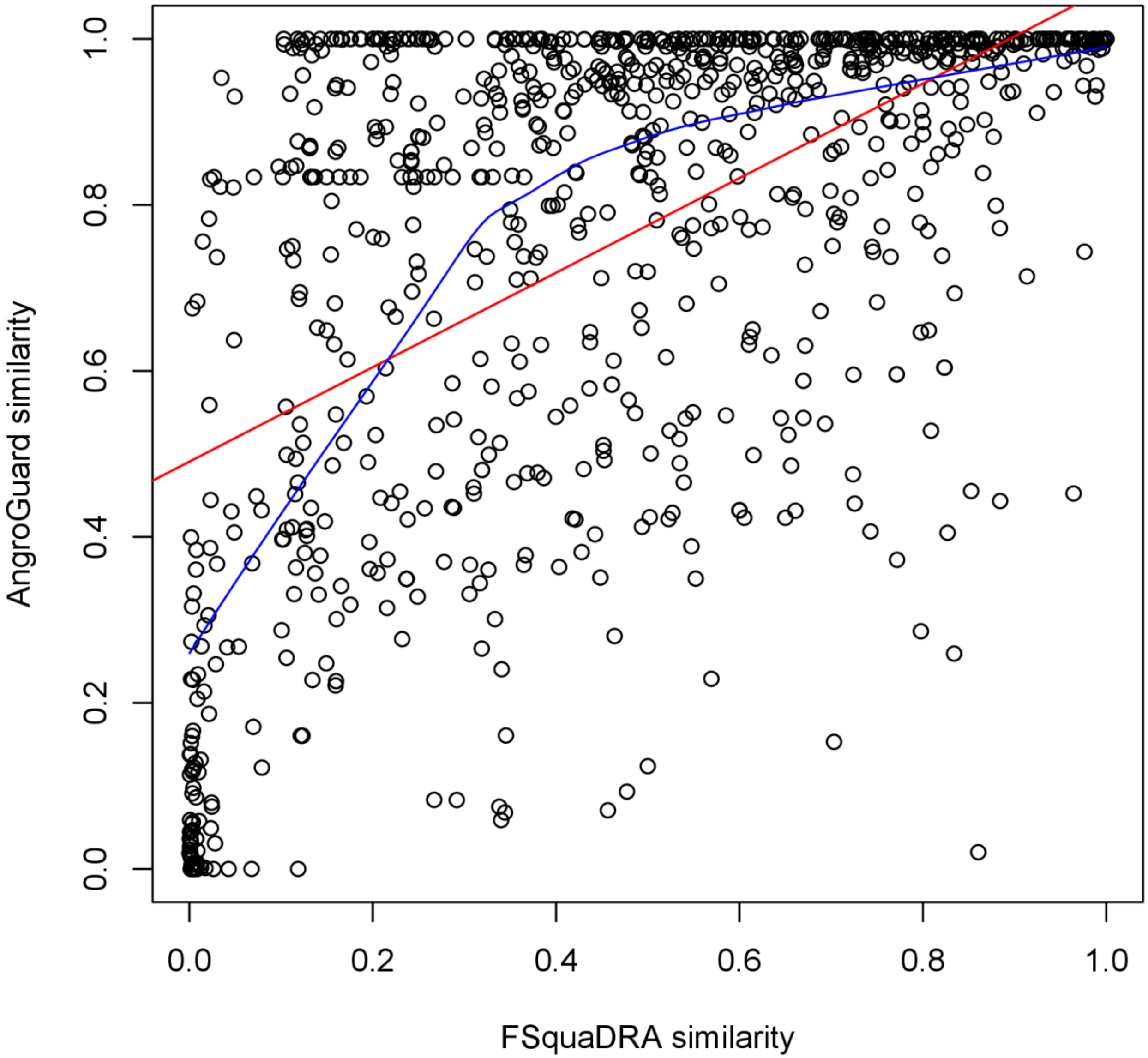
-st. dev.: 0.1862

-median: -0.0480

Red: line of best fit

Blue: LOWESS (locally weighted scatterplot smoothing line)

Effectiveness: Rebranding Results (same certificates, fss>0)



Correlation: 0.5807

Difference (fss-ags):

-mean: -0.2761

-st. dev.: 0.2704

-median: -0.2518

Red: line of best fit

Blue: LOWESS (locally weighted scatterplot smoothing line)

FSquaDRA: Features

- The first solution detecting repackaged apps based on resource files
- Our resource-based similarity score is highly correlated with the code-based similarity score of AndroGuard (**0.79** for plagiarizing, **0.58** for rebranding)
- Faster than any known competitor
  - DNADroid by J. Crussell et al. (ESORICS 2012) - **0.012 app pair/sec**
    - PDG subgraph isomorphism
    - Hadoop MapReduce framework with a server and 3 desktops
  - Juxtapp by S. Hanna et al. (DIMVA 2012) - **49.4 app pair/sec**
    - *k*-grams of opcodes → hashing → feature vector → Jaccard distance
    - Intel Xeon CPU (8 cores) , 8GB of RAM
  - Our approach - **6700 app pair/sec**
- Open-source *

*** https://github.com/zyrikby/FSquaDRA**

FSquaDRA: Future Work

- The proposed solution is not sustainable:
  - attackers can change a bit in all files in apk
  - adversaries can add a lot of new resources to decrease the similarity score
  - libraries containing resources may influence the similarity score
- No clear values for false positive and false negative scores
  - absence of publicly available dataset
  - almost all already developed tools (except AndroGuard) are not available

THANK YOU

zhauniarovich@disi.unitn.it