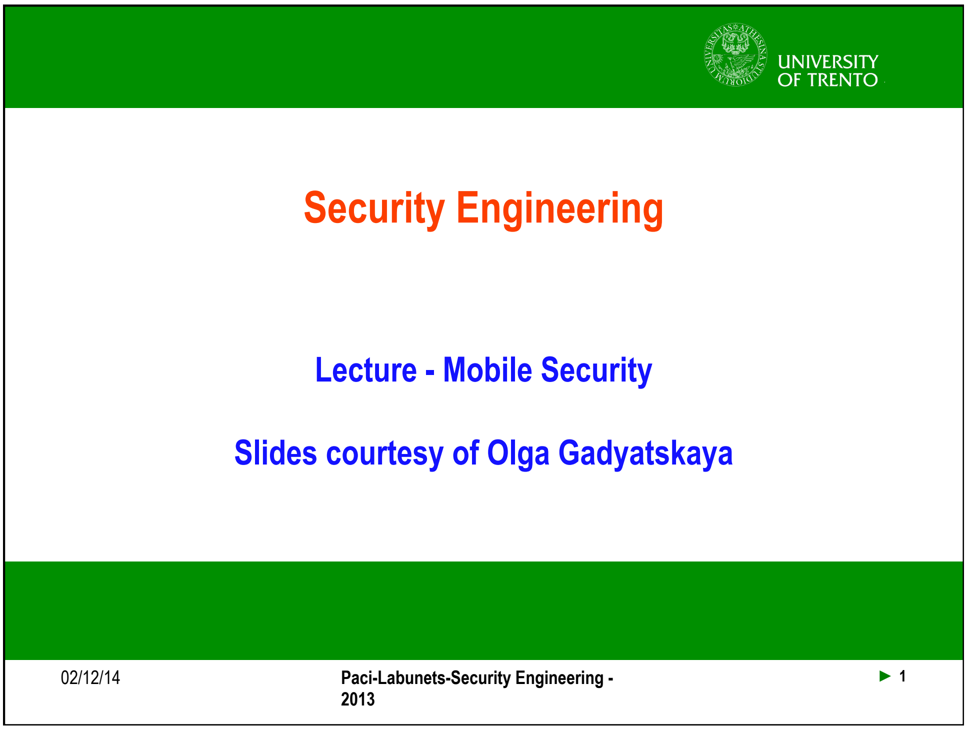


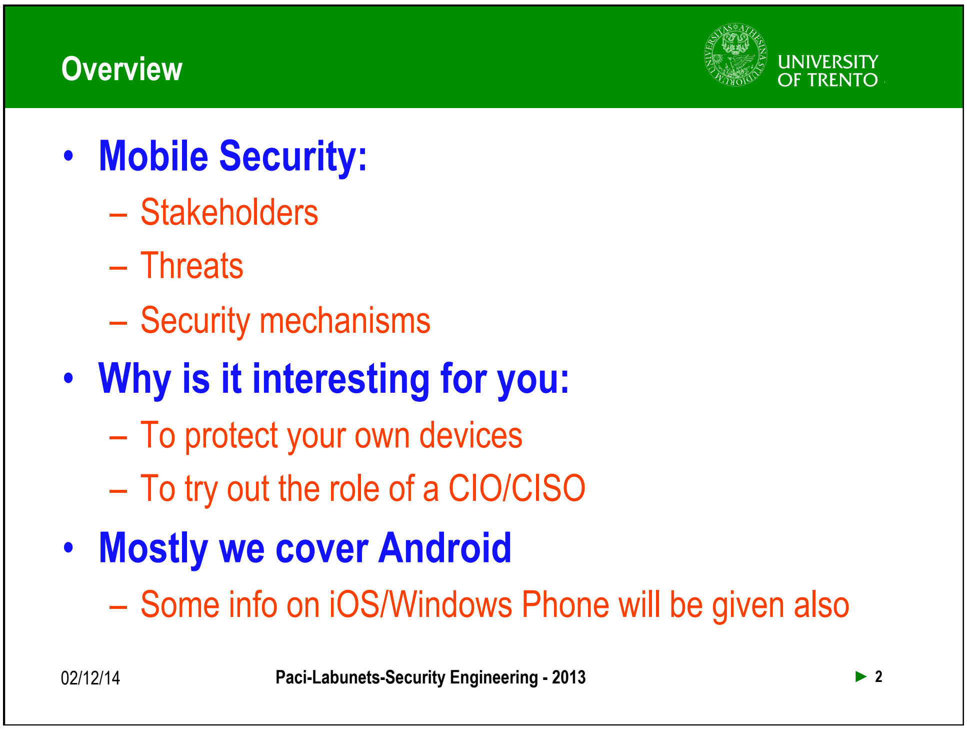
UNIVERSITY OF TRENTO

Security Engineering

Lecture - Mobile Security

Slides courtesy of Olga Gadyatskaya

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 1

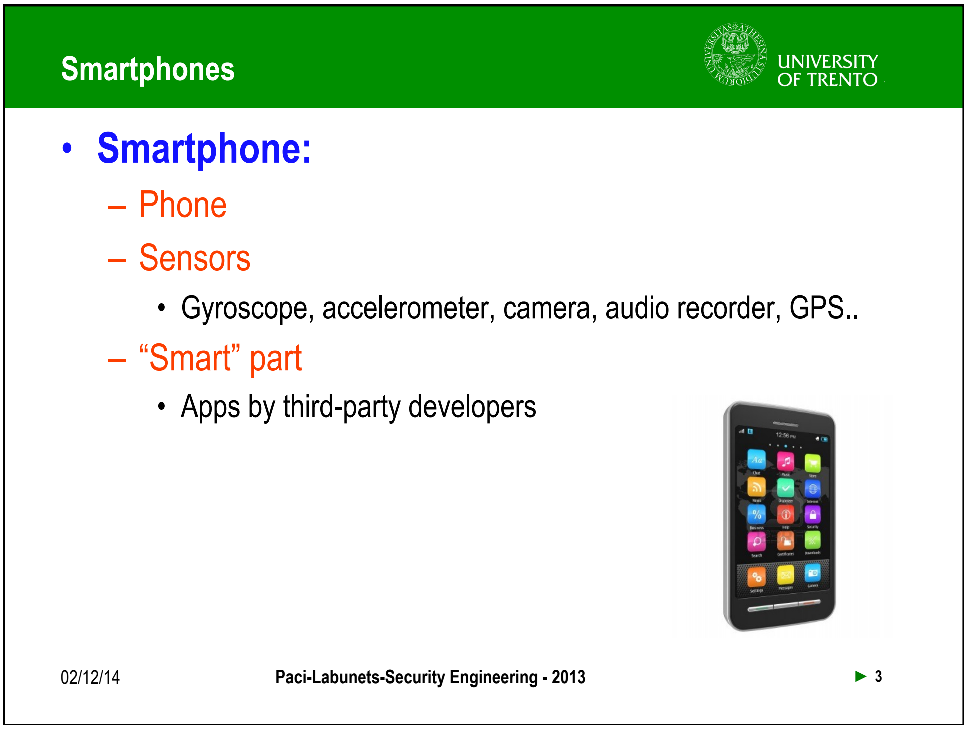


Overview

UNIVERSITY OF TRENTO

- **Mobile Security:**
  - Stakeholders
  - Threats
  - Security mechanisms
- **Why is it interesting for you:**
  - To protect your own devices
  - To try out the role of a CIO/CISO
- **Mostly we cover Android**
  - Some info on iOS/Windows Phone will be given also

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 2

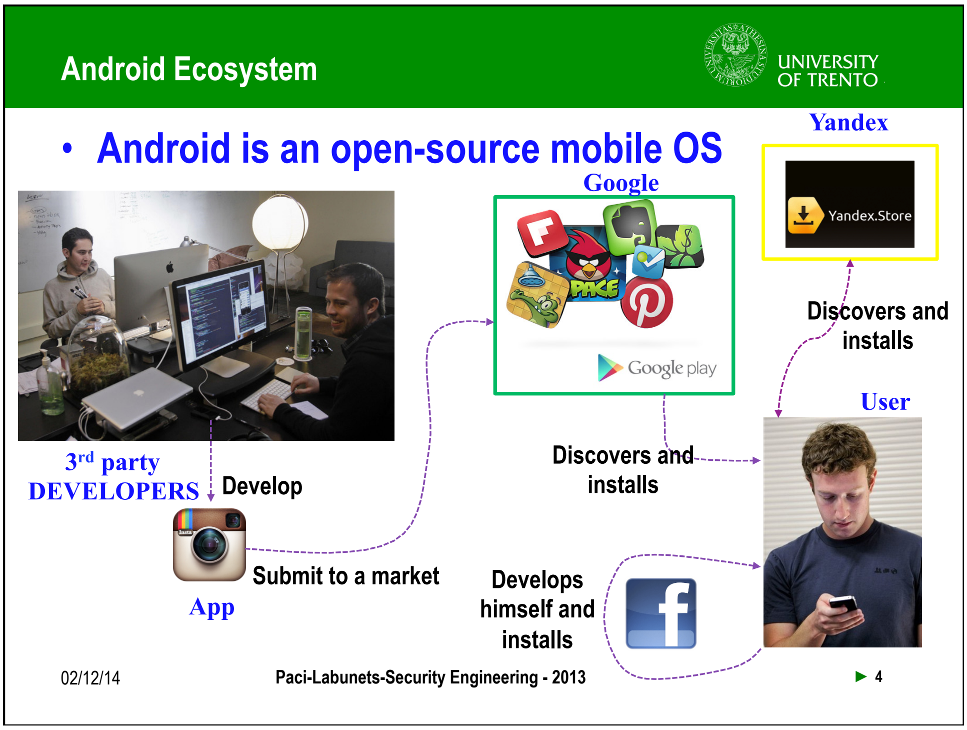


Smartphones

UNIVERSITY OF TRENTO

- **Smartphone:**
  - Phone
  - Sensors
    - Gyroscope, accelerometer, camera, audio recorder, GPS..
  - "Smart" part
    - Apps by third-party developers

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 3



Android Ecosystem

UNIVERSITY OF TRENTO

- **Android is an open-source mobile OS**

Yandex

Yandex.Store

Google

Google play

3rd party DEVELOPERS

Develop

App

Submit to a market

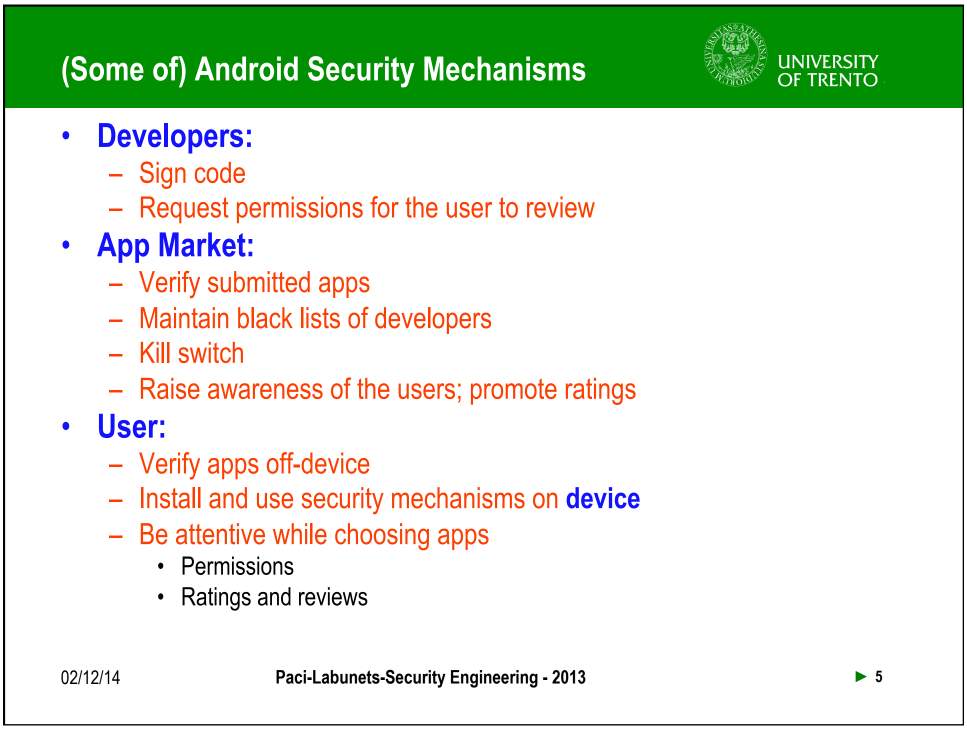
Discovers and installs

Discovers and installs

User

Develops himself and installs

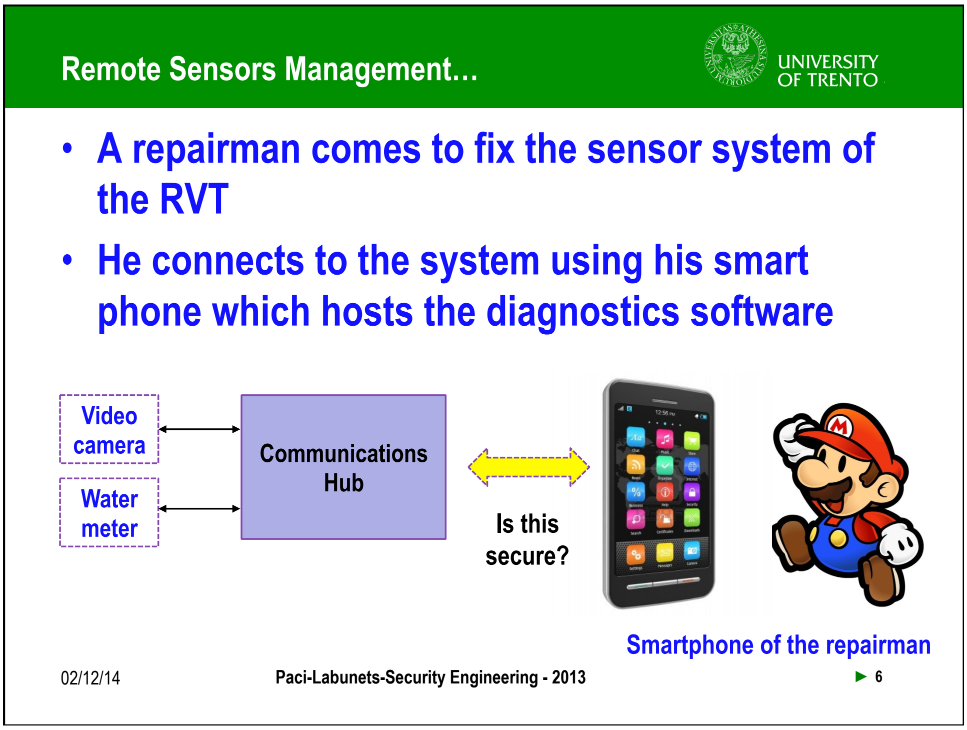
02/12/14 Paci-Labunets-Security Engineering - 2013 ► 4



(Some of) Android Security Mechanisms

UNIVERSITY OF TRENTO

- **Developers:**
  - Sign code
  - Request permissions for the user to review
- **App Market:**
  - Verify submitted apps
  - Maintain black lists of developers
  - Kill switch
  - Raise awareness of the users; promote ratings
- **User:**
  - Verify apps off-device
  - Install and use security mechanisms on **device**
  - Be attentive while choosing apps
    - Permissions
    - Ratings and reviews



Remote Sensors Management…

UNIVERSITY OF TRENTO

- **A repairman comes to fix the sensor system of the RVT**
- **He connects to the system using his smart phone which hosts the diagnostics software**

Video camera

Water meter

Communications Hub

Is this secure?

Smartphone of the repairman

Bring Your Own Device (BYOD)

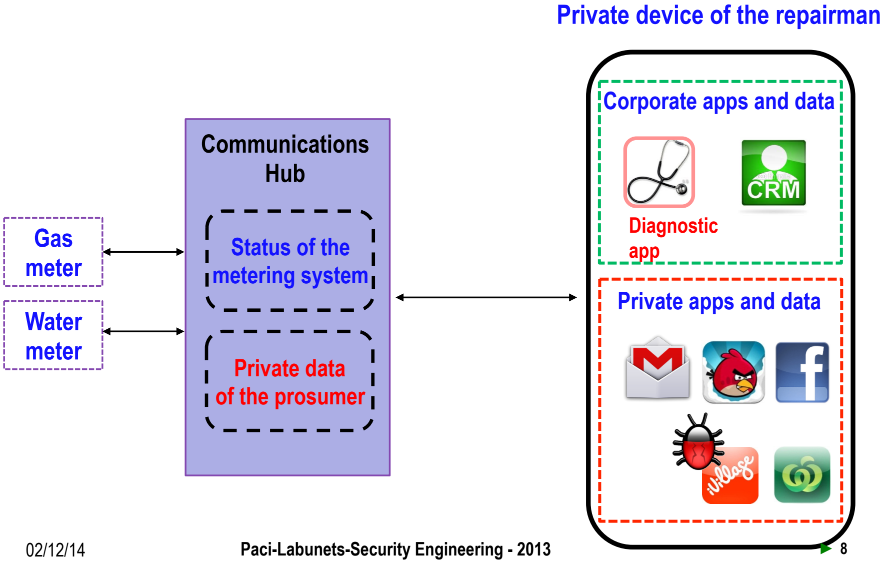
UNIVERSITY OF TRENTO

- **BYOD is a paradigm in which the employees can use their own mobile device for work purposes**
  - Pros:
    - Employers save money on devices
    - Performance of employees raises
  - Cons:
    - The repairman's phone is not trusted: there can be malware, spyware, etc
- **Security concerns in the BYOD scenario:**
  - Separation of corporate and private (personal) data
  - Protection of corporate data
  - Protection of personal data

03/12/14 Paci-Labunets-Security Engineering - 2013 *CIO/CISO* ► 7

Where is the real problem?

UNIVERSITY OF TRENTO



Private device of the repairman

Corporate apps and data

Diagnostic app

CRM

Private apps and data

Communications Hub

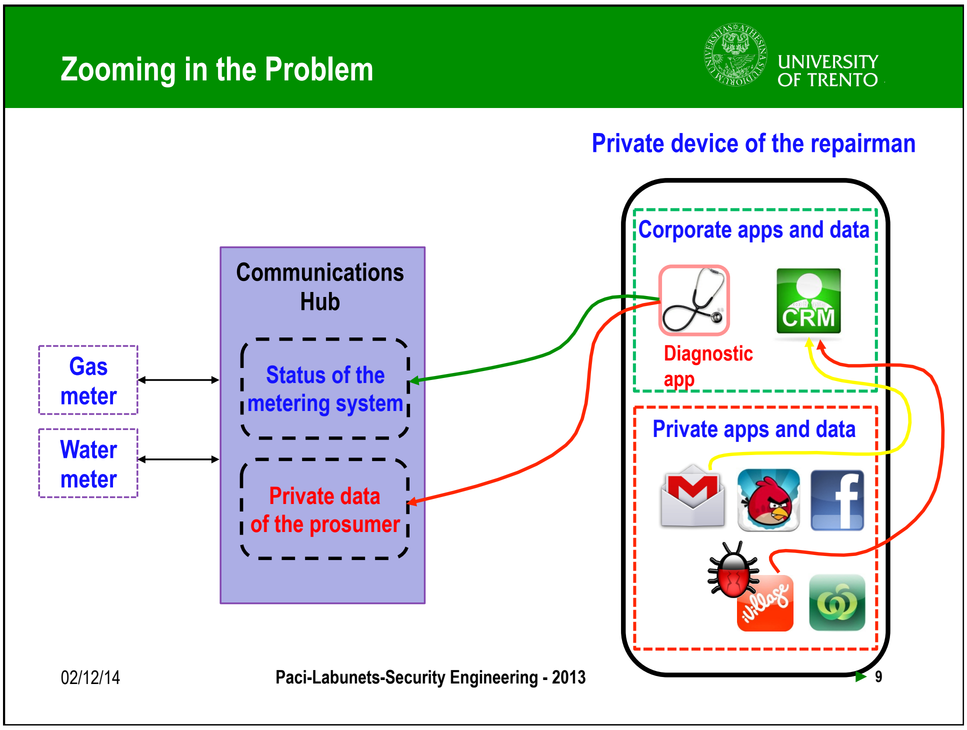
Status of the metering system

Private data of the prosumer

Gas meter

Water meter

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 8



Zooming in the Problem

UNIVERSITY OF TRENTO

Private device of the repairman

Corporate apps and data

Diagnostic app

CRM

Private apps and data

Communications Hub

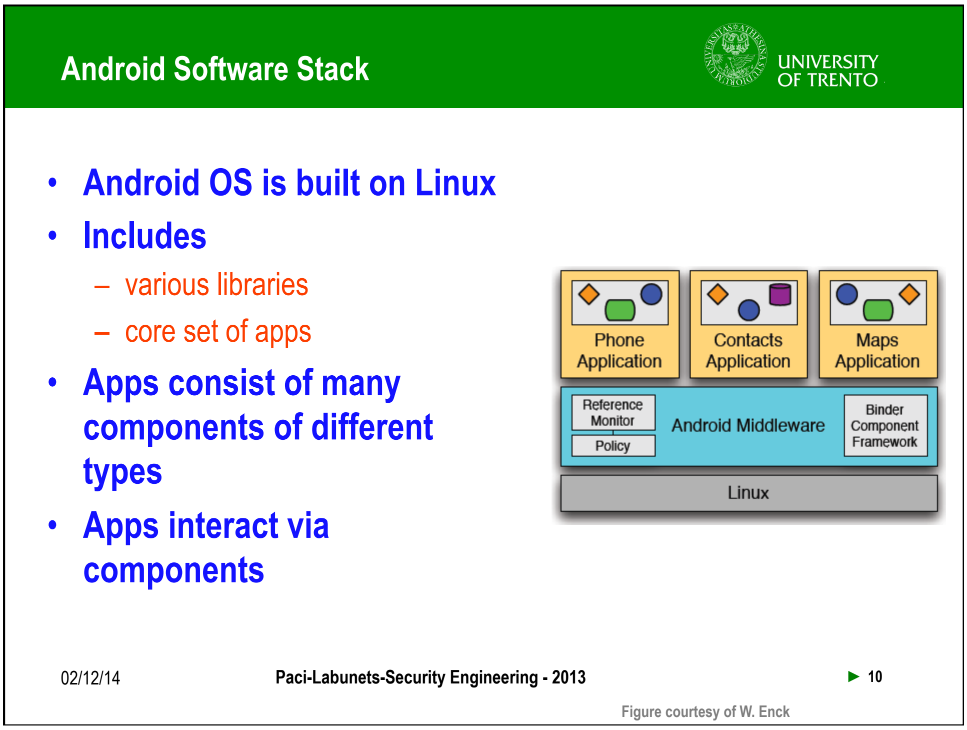
Status of the metering system

Private data of the prosumer

Gas meter

Water meter

02/12/14 Paci-Labunets-Security Engineering - 2013 9



Android Software Stack

UNIVERSITY OF TRENTO

- Android OS is built on Linux
- Includes
  - various libraries
  - core set of apps
- Apps consist of many components of different types
- Apps interact via components

Phone Application

Contacts Application

Maps Application

Reference Monitor

Policy

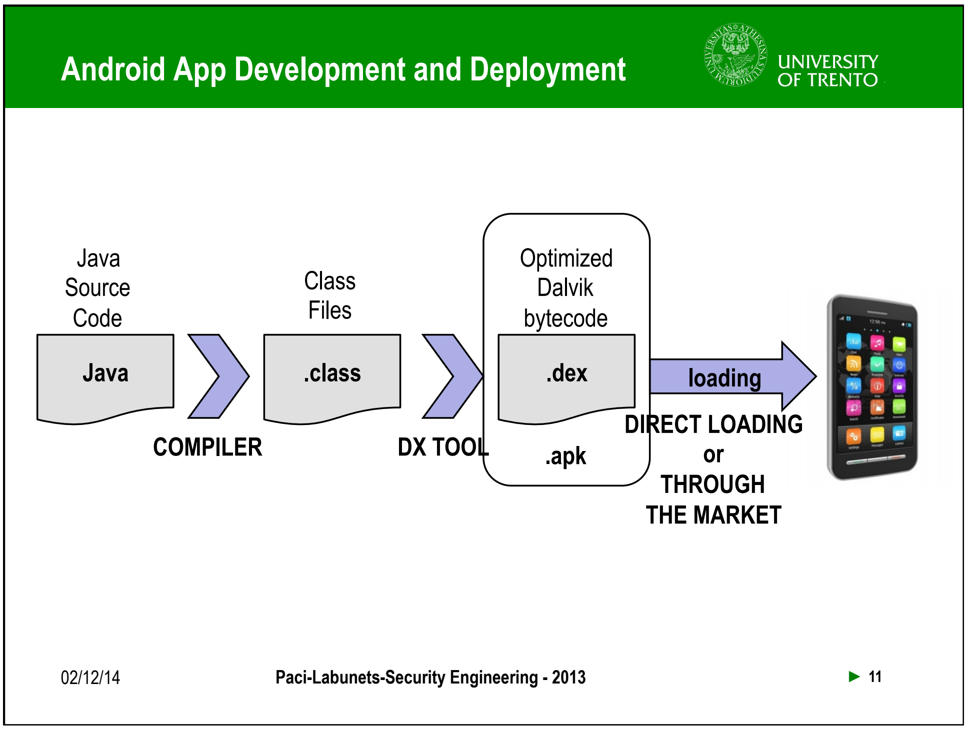
Android Middleware

Binder Component Framework

Linux

02/12/14 Paci-Labunets-Security Engineering - 2013 10

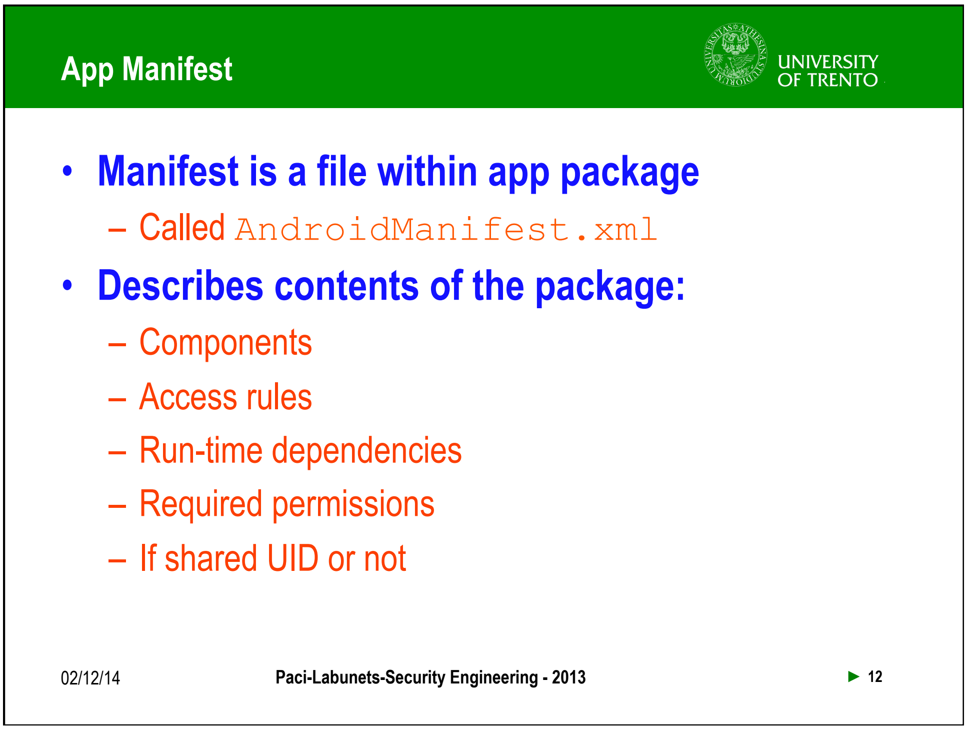
Figure courtesy of W. Enck



Android App Development and Deployment

UNIVERSITY OF TRENTO

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 11

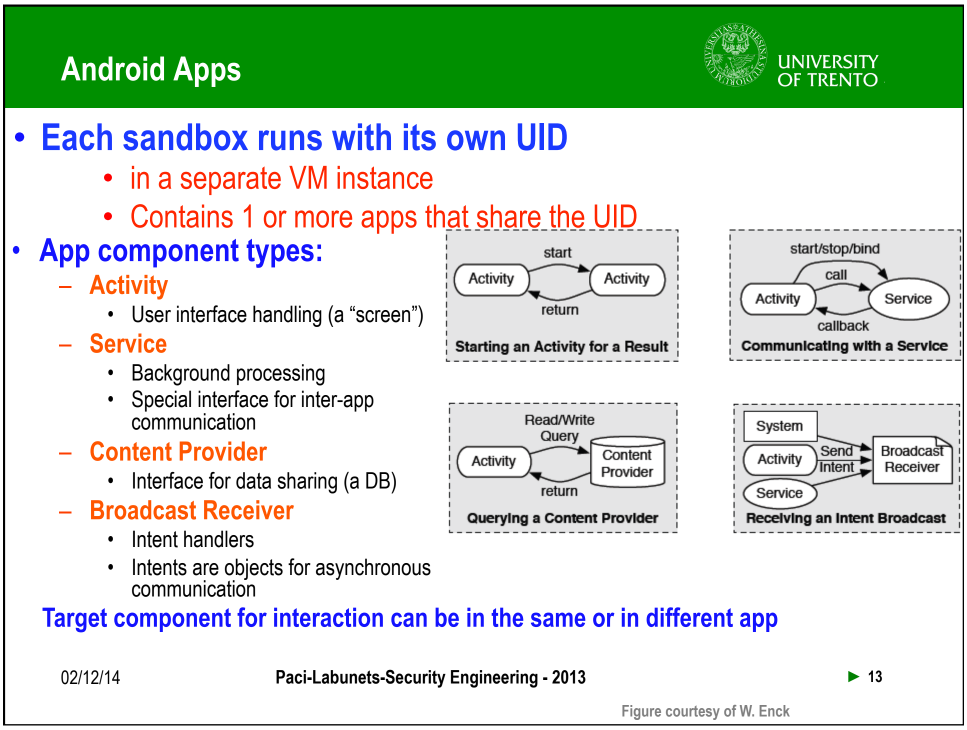


App Manifest

UNIVERSITY OF TRENTO

- **Manifest is a file within app package**
  - Called `AndroidManifest.xml`
- **Describes contents of the package:**
  - Components
  - Access rules
  - Run-time dependencies
  - Required permissions
  - If shared UID or not

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 12



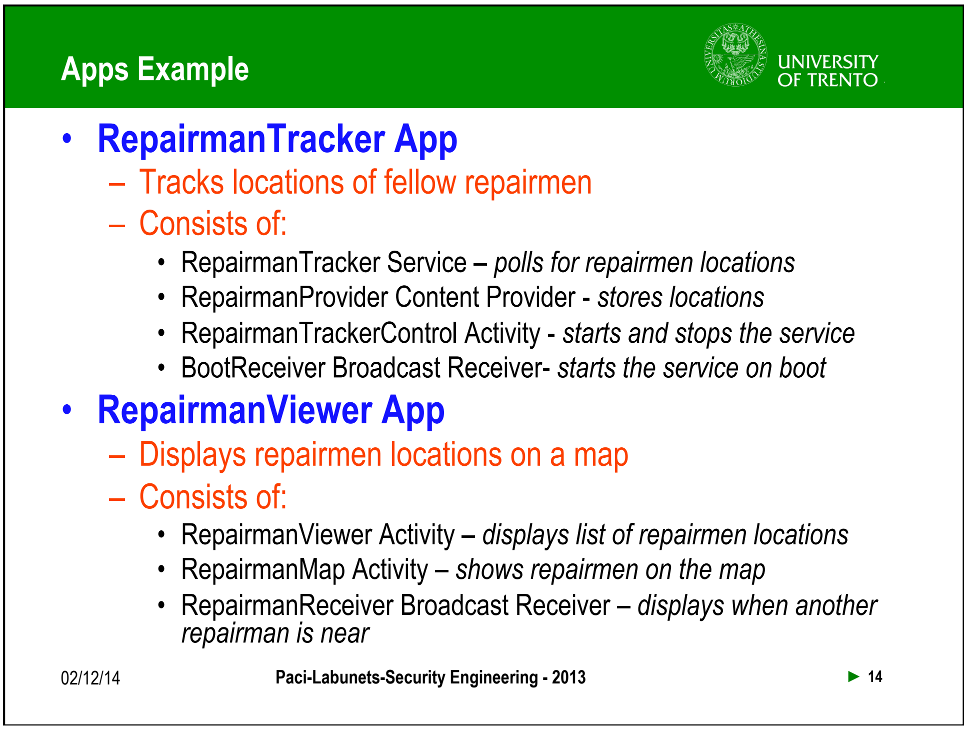
Android Apps

UNIVERSITY OF TRENTO

- **Each sandbox runs with its own UID**
  - in a separate VM instance
  - Contains 1 or more apps that share the UID
- **App component types:**
  - **Activity**
    - User interface handling (a "screen")
  - **Service**
    - Background processing
    - Special interface for inter-app communication
  - **Content Provider**
    - Interface for data sharing (a DB)
  - **Broadcast Receiver**
    - Intent handlers
    - Intents are objects for asynchronous communication

Target component for interaction can be in the same or in different app

Figure courtesy of W. Enck



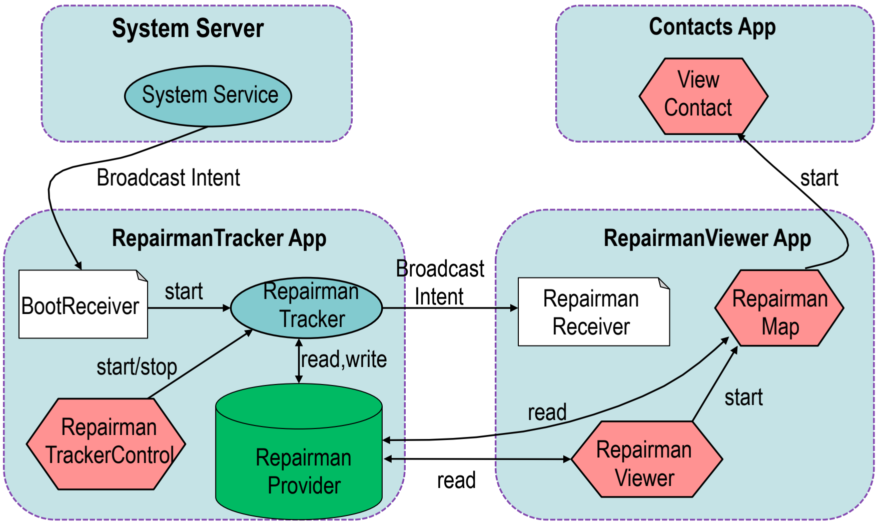
Apps Example

UNIVERSITY OF TRENTO

- **RepairmanTracker App**
  - Tracks locations of fellow repairmen
  - Consists of:
    - RepairmanTracker Service – *polls for repairmen locations*
    - RepairmanProvider Content Provider - *stores locations*
    - RepairmanTrackerControl Activity - *starts and stops the service*
    - BootReceiver Broadcast Receiver- *starts the service on boot*
- **RepairmanViewer App**
  - Displays repairmen locations on a map
  - Consists of:
    - RepairmanViewer Activity – *displays list of repairmen locations*
    - RepairmanMap Activity – *shows repairmen on the map*
    - RepairmanReceiver Broadcast Receiver – *displays when another repairman is near*

RepairmanTracker and RepairmanViewer

UNIVERSITY OF TRENTO



System Server

System Service

Broadcast Intent

Contacts App

View Contact

start

RepairmanTracker App

BootReceiver

start

Repairman Tracker

start/stop

read,write

Repairman TrackerControl

Repairman Provider

Broadcast Intent

RepairmanViewer App

Repairman Receiver

Repairman Map

read

start

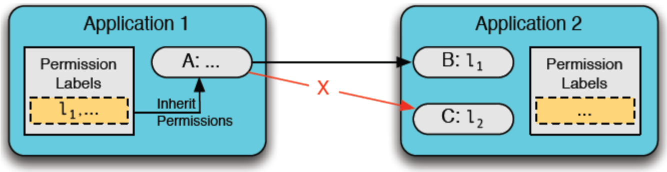
Repairman Viewer

read

Android Security Policy for IPC

UNIVERSITY OF TRENTO

- **Android is focused on inter-component communication**
- **Developers can define in the manifest file access control policy to access components**
  - Each component can be labeled with an access permission
  - Each app requests a list of permissions
    - Fixed at install
- **Android IPC Security Policy can be summarized as:**



Application 1

Permission Labels

l_1,...

A: ...

Inherit Permissions

X

Application 2

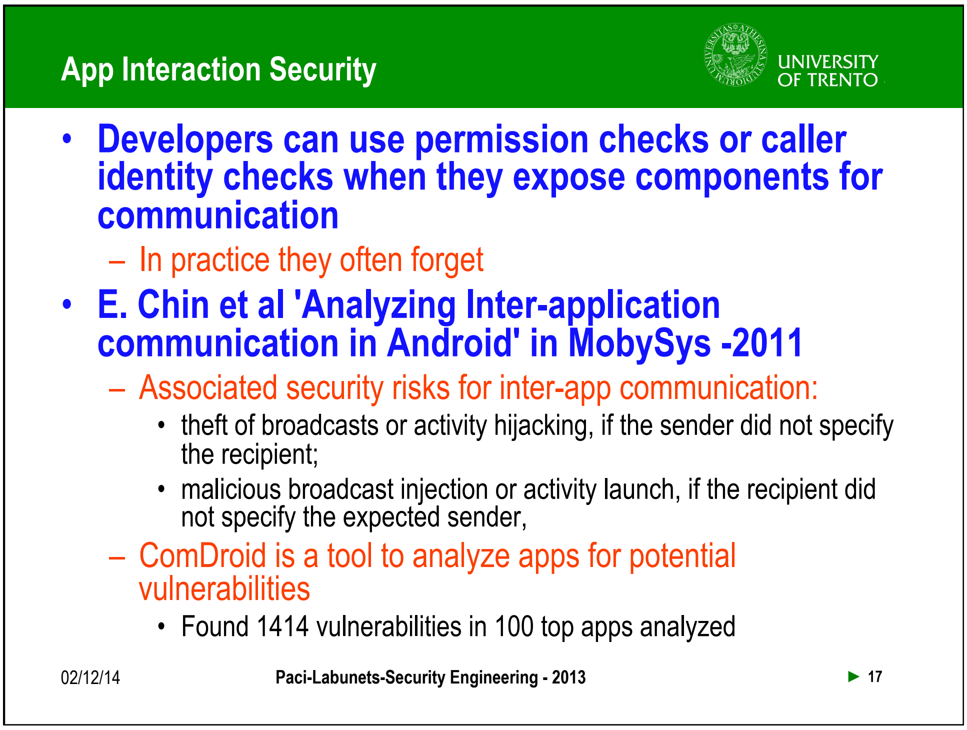
B: l_1

C: l_2

Permission Labels

...

Figure courtesy of W. Enck

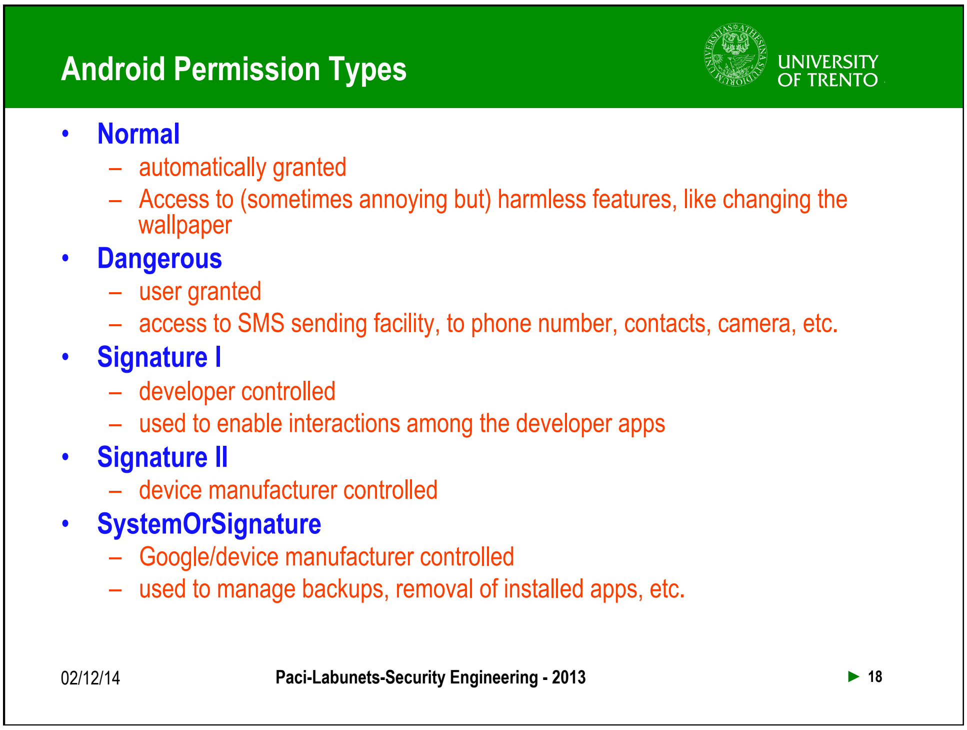


App Interaction Security

UNIVERSITY OF TRENTO

- **Developers can use permission checks or caller identity checks when they expose components for communication**
  - In practice they often forget
- **E. Chin et al 'Analyzing Inter-application communication in Android' in MobySys -2011**
  - Associated security risks for inter-app communication:
    - theft of broadcasts or activity hijacking, if the sender did not specify the recipient;
    - malicious broadcast injection or activity launch, if the recipient did not specify the expected sender,
  - ComDroid is a tool to analyze apps for potential vulnerabilities
    - Found 1414 vulnerabilities in 100 top apps analyzed

02/12/14 **Paci-Labunets-Security Engineering - 2013** ► **17**

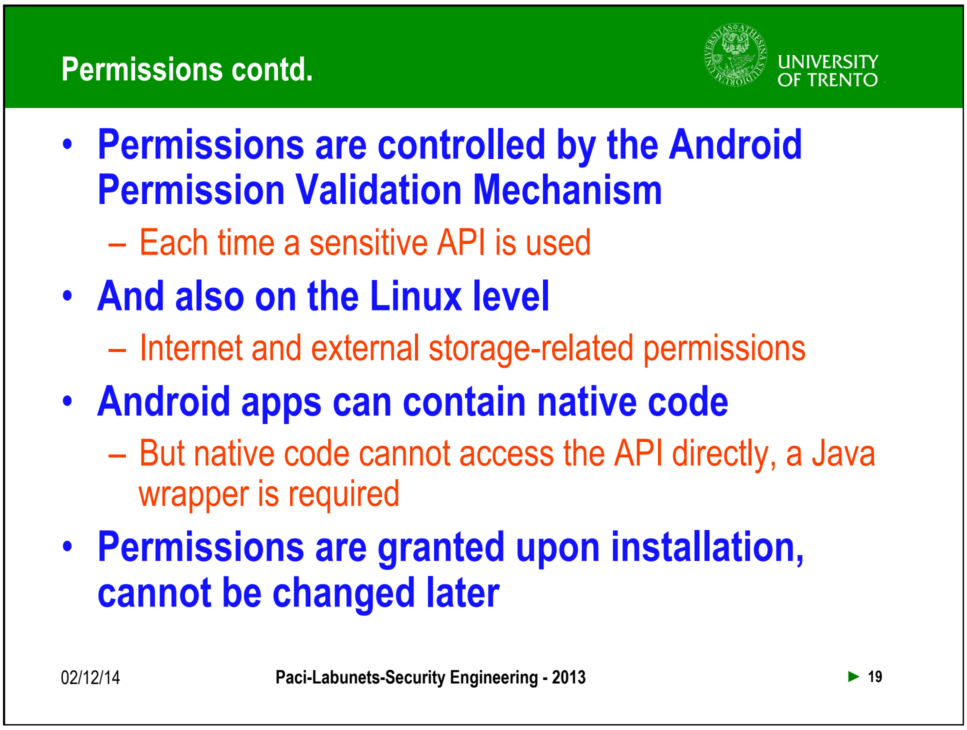


Android Permission Types

UNIVERSITY OF TRENTO

- **Normal**
  - automatically granted
  - Access to (sometimes annoying but) harmless features, like changing the wallpaper
- **Dangerous**
  - user granted
  - access to SMS sending facility, to phone number, contacts, camera, etc.
- **Signature I**
  - developer controlled
  - used to enable interactions among the developer apps
- **Signature II**
  - device manufacturer controlled
- **SystemOrSignature**
  - Google/device manufacturer controlled
  - used to manage backups, removal of installed apps, etc.

02/12/14 **Paci-Labunets-Security Engineering - 2013** ► **18**

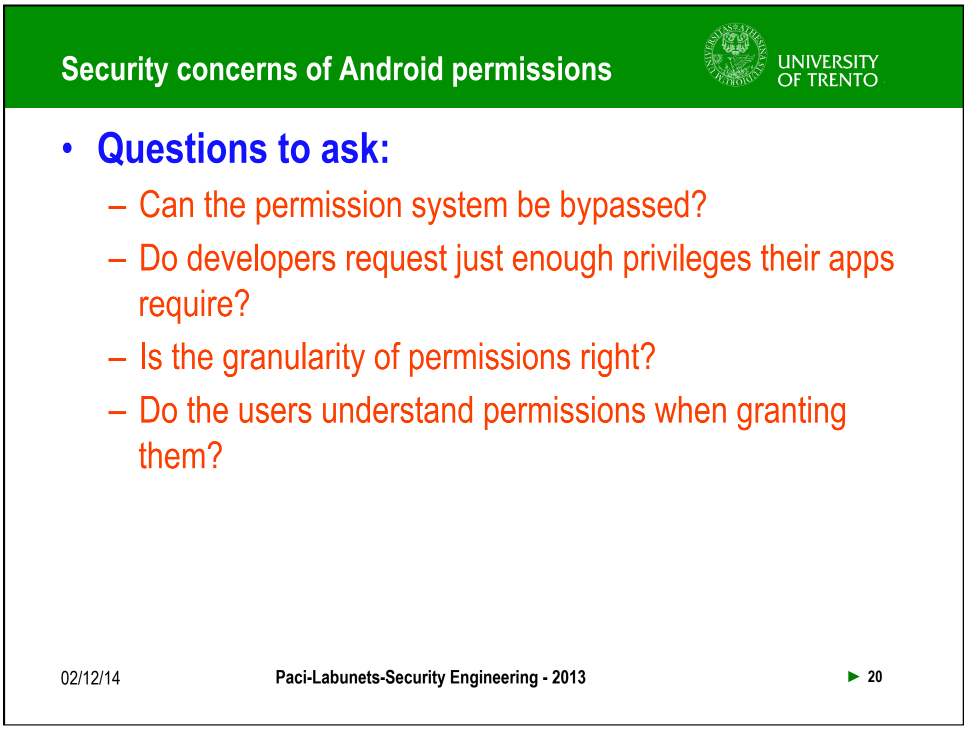


Permissions contd.

UNIVERSITY OF TRENTO

- Permissions are controlled by the Android Permission Validation Mechanism
  - Each time a sensitive API is used
- And also on the Linux level
  - Internet and external storage-related permissions
- Android apps can contain native code
  - But native code cannot access the API directly, a Java wrapper is required
- Permissions are granted upon installation, cannot be changed later

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 19

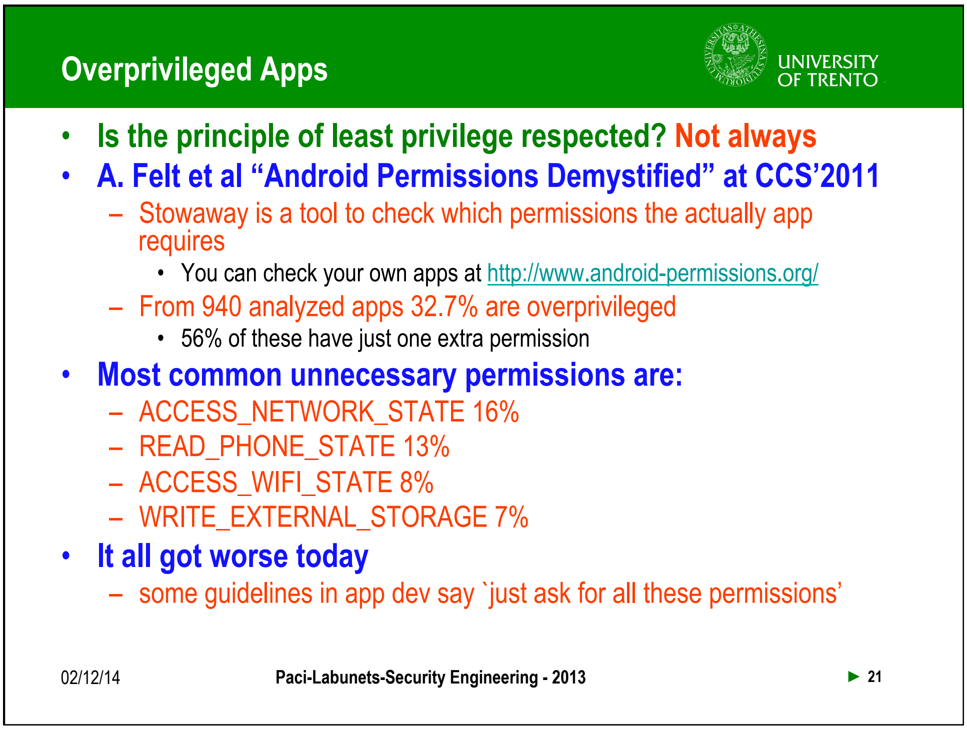


Security concerns of Android permissions

UNIVERSITY OF TRENTO

- Questions to ask:
  - Can the permission system be bypassed?
  - Do developers request just enough privileges their apps require?
  - Is the granularity of permissions right?
  - Do the users understand permissions when granting them?

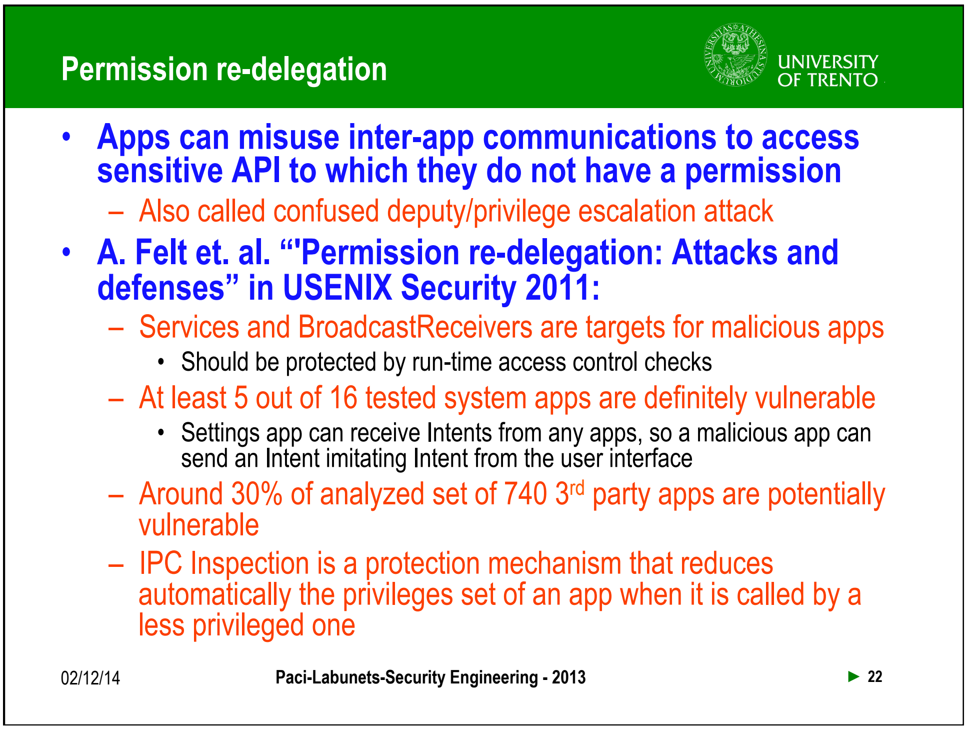
02/12/14 Paci-Labunets-Security Engineering - 2013 ► 20



Overprivileged Apps

UNIVERSITY OF TRENTO

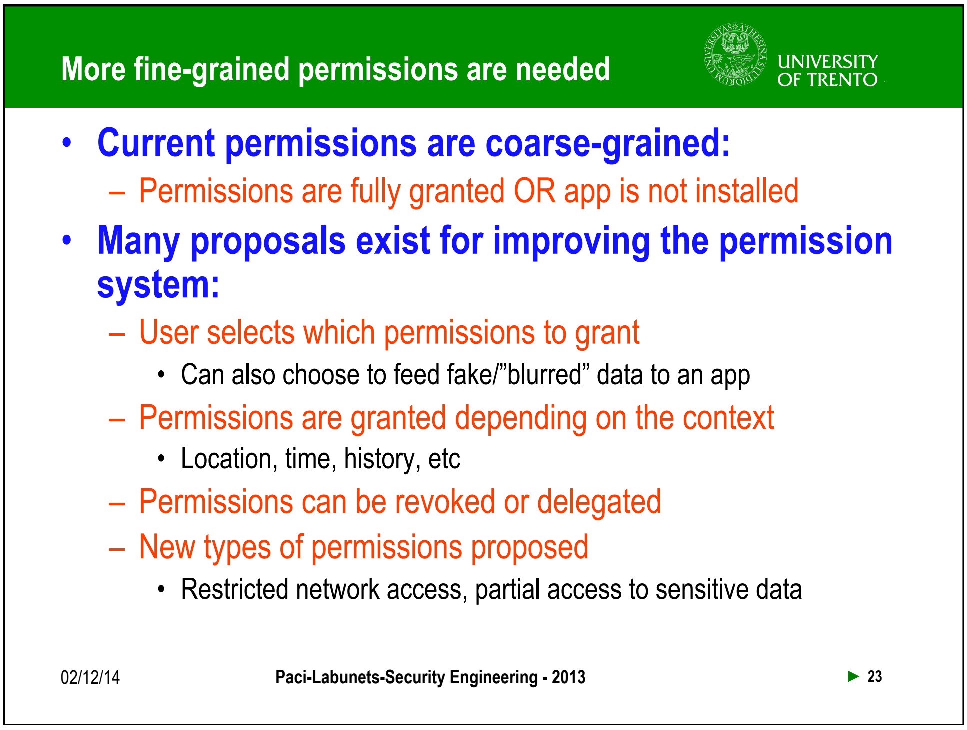
- Is the principle of least privilege respected? Not always
- A. Felt et al “Android Permissions Demystified” at CCS’2011
  - Stowaway is a tool to check which permissions the actually app requires
    - You can check your own apps at http://www.android-permissions.org/
  - From 940 analyzed apps 32.7% are overprivileged
    - 56% of these have just one extra permission
- Most common unnecessary permissions are:
  - ACCESS_NETWORK_STATE 16%
  - READ_PHONE_STATE 13%
  - ACCESS_WIFI_STATE 8%
  - WRITE_EXTERNAL_STORAGE 7%
- It all got worse today
  - some guidelines in app dev say `just ask for all these permissions’



Permission re-delegation

UNIVERSITY OF TRENTO

- Apps can misuse inter-app communications to access sensitive API to which they do not have a permission
  - Also called confused deputy/privilege escalation attack
- A. Felt et. al. “'Permission re-delegation: Attacks and defenses” in USENIX Security 2011:
  - Services and BroadcastReceivers are targets for malicious apps
    - Should be protected by run-time access control checks
  - At least 5 out of 16 tested system apps are definitely vulnerable
    - Settings app can receive Intents from any apps, so a malicious app can send an Intent imitating Intent from the user interface
  - Around 30% of analyzed set of 740 3rd party apps are potentially vulnerable
  - IPC Inspection is a protection mechanism that reduces automatically the privileges set of an app when it is called by a less privileged one

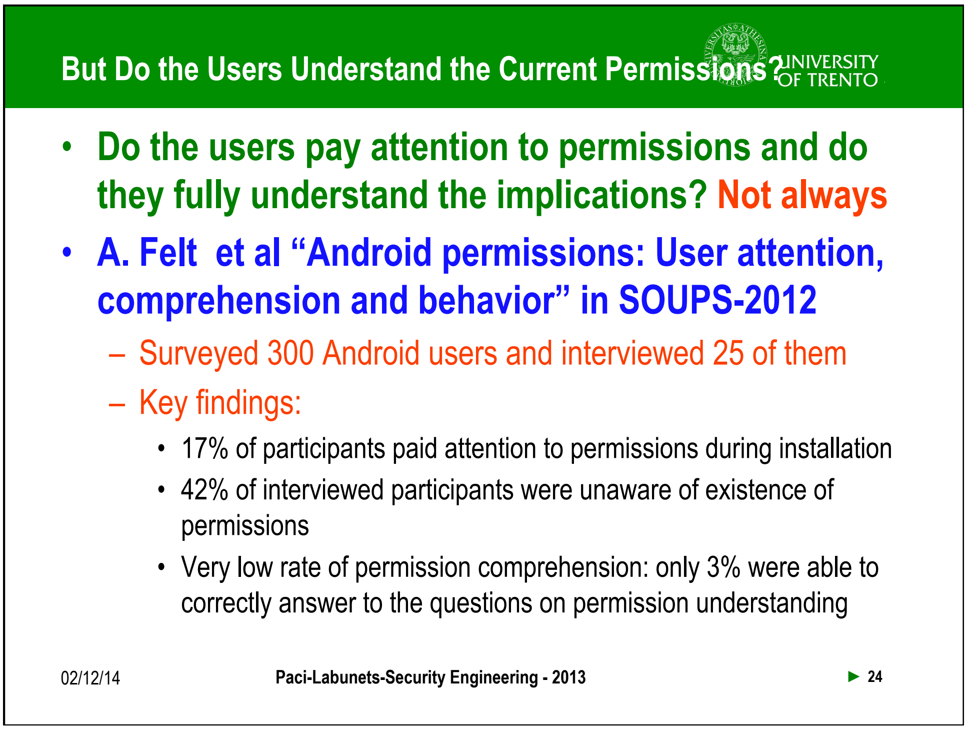


More fine-grained permissions are needed

UNIVERSITY OF TRENTO

- **Current permissions are coarse-grained:**
  - Permissions are fully granted OR app is not installed
- **Many proposals exist for improving the permission system:**
  - User selects which permissions to grant
    - Can also choose to feed fake/"blurred" data to an app
  - Permissions are granted depending on the context
    - Location, time, history, etc
  - Permissions can be revoked or delegated
  - New types of permissions proposed
    - Restricted network access, partial access to sensitive data

02/12/14 **Paci-Labunets-Security Engineering - 2013** ► 23

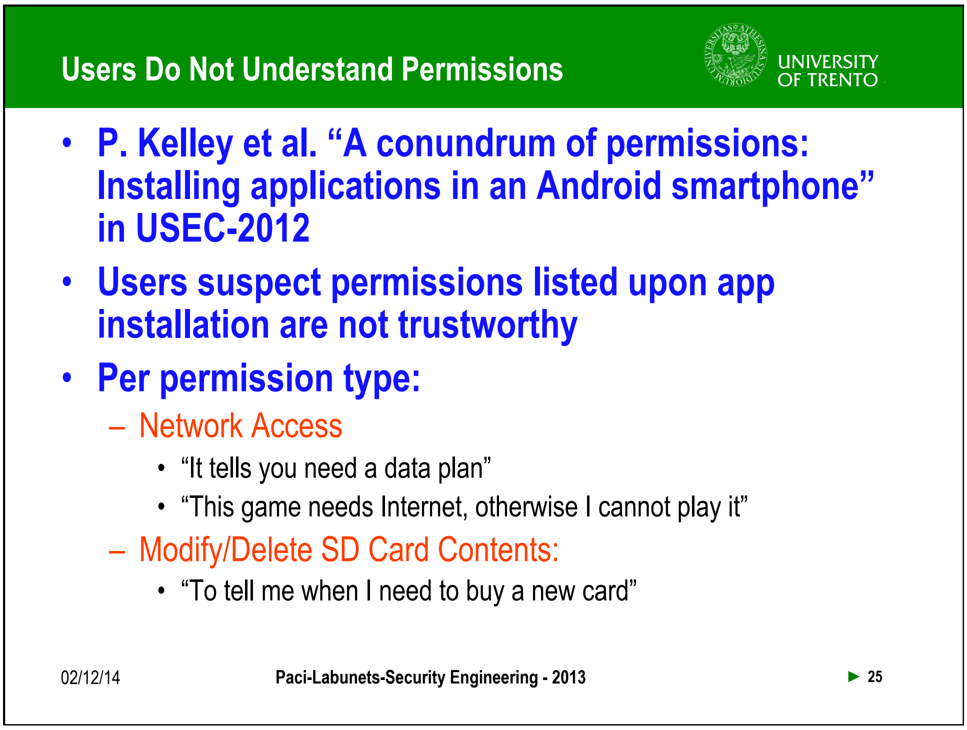


But Do the Users Understand the Current Permissions?

UNIVERSITY OF TRENTO

- **Do the users pay attention to permissions and do they fully understand the implications? Not always**
- **A. Felt et al "Android permissions: User attention, comprehension and behavior" in SOUPS-2012**
  - Surveyed 300 Android users and interviewed 25 of them
  - Key findings:
    - 17% of participants paid attention to permissions during installation
    - 42% of interviewed participants were unaware of existence of permissions
    - Very low rate of permission comprehension: only 3% were able to correctly answer to the questions on permission understanding

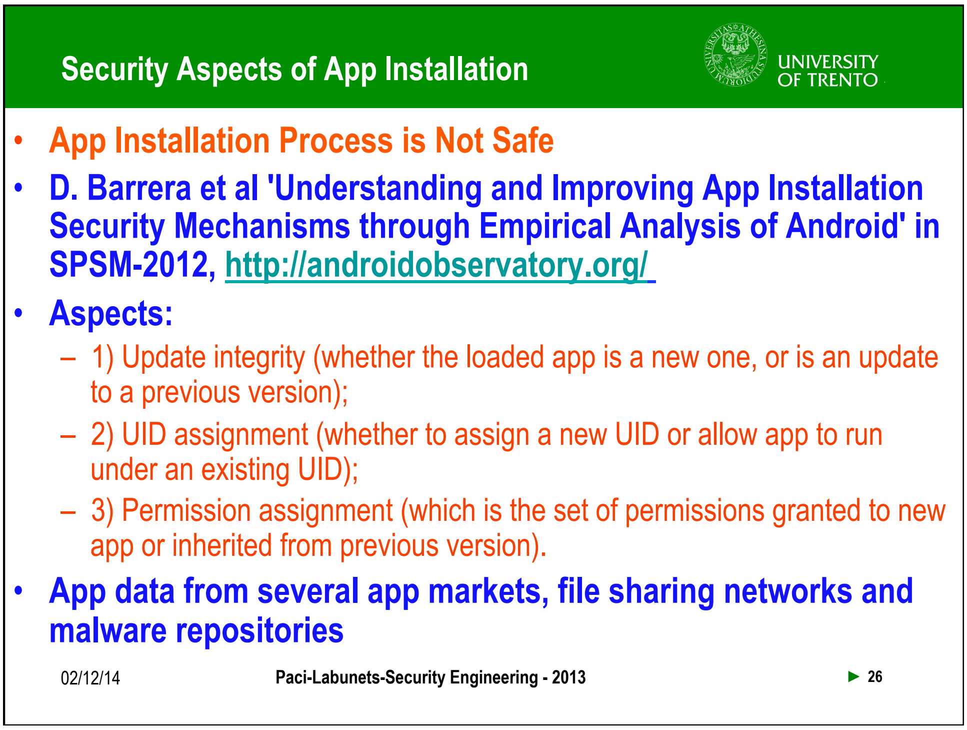
02/12/14 **Paci-Labunets-Security Engineering - 2013** ► 24



Users Do Not Understand Permissions

UNIVERSITY OF TRENTO

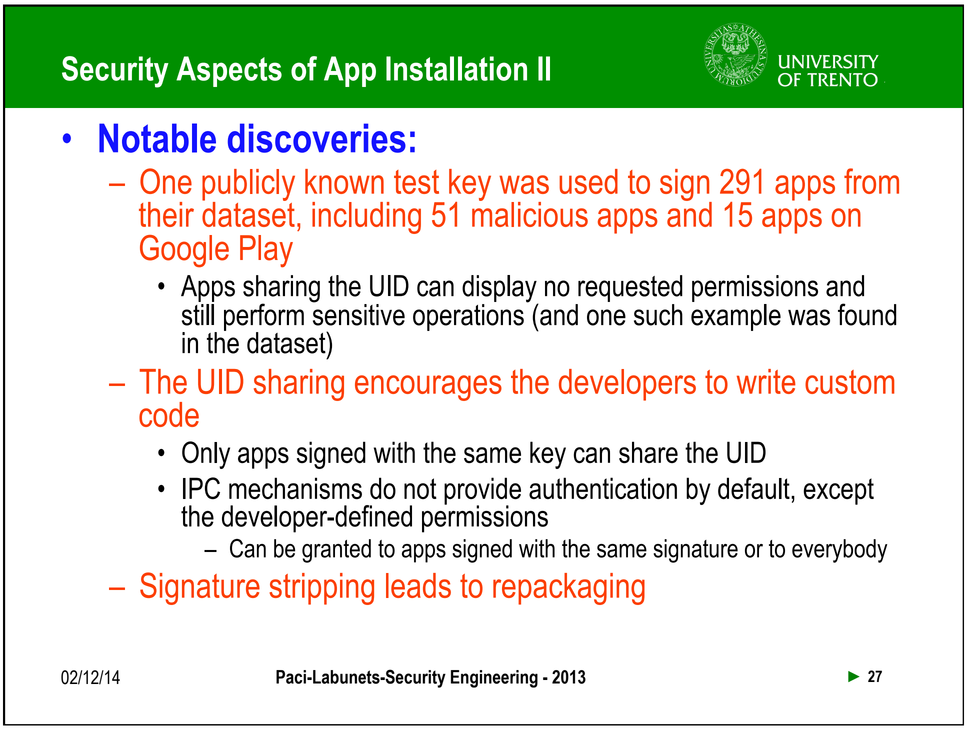
- **P. Kelley et al. “A conundrum of permissions: Installing applications in an Android smartphone” in USEC-2012**
- **Users suspect permissions listed upon app installation are not trustworthy**
- **Per permission type:**
  - Network Access
    - “It tells you need a data plan”
    - “This game needs Internet, otherwise I cannot play it”
  - Modify/Delete SD Card Contents:
    - “To tell me when I need to buy a new card”



Security Aspects of App Installation

UNIVERSITY OF TRENTO

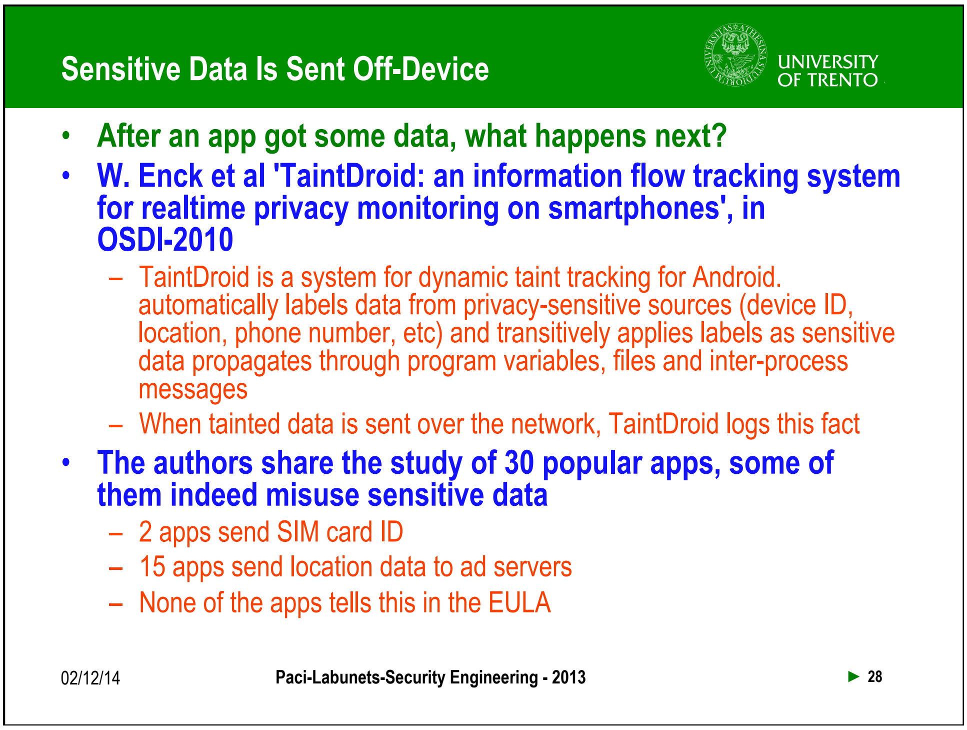
- **App Installation Process is Not Safe**
- **D. Barrera et al 'Understanding and Improving App Installation Security Mechanisms through Empirical Analysis of Android' in SPSM-2012, http://androidobservatory.org/**
- **Aspects:**
  - 1) Update integrity (whether the loaded app is a new one, or is an update to a previous version);
  - 2) UID assignment (whether to assign a new UID or allow app to run under an existing UID);
  - 3) Permission assignment (which is the set of permissions granted to new app or inherited from previous version).
- **App data from several app markets, file sharing networks and malware repositories**



Security Aspects of App Installation II

UNIVERSITY OF TRENTO

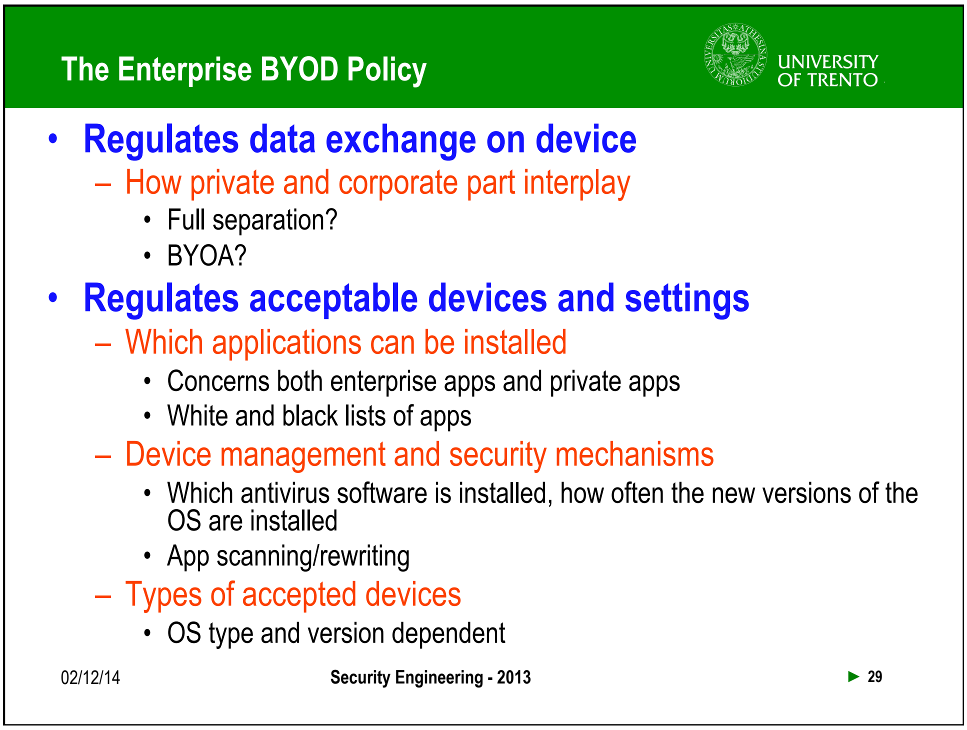
- **Notable discoveries:**
  - One publicly known test key was used to sign 291 apps from their dataset, including 51 malicious apps and 15 apps on Google Play
    - Apps sharing the UID can display no requested permissions and still perform sensitive operations (and one such example was found in the dataset)
  - The UID sharing encourages the developers to write custom code
    - Only apps signed with the same key can share the UID
    - IPC mechanisms do not provide authentication by default, except the developer-defined permissions
      - Can be granted to apps signed with the same signature or to everybody
  - Signature stripping leads to repackaging



Sensitive Data Is Sent Off-Device

UNIVERSITY OF TRENTO

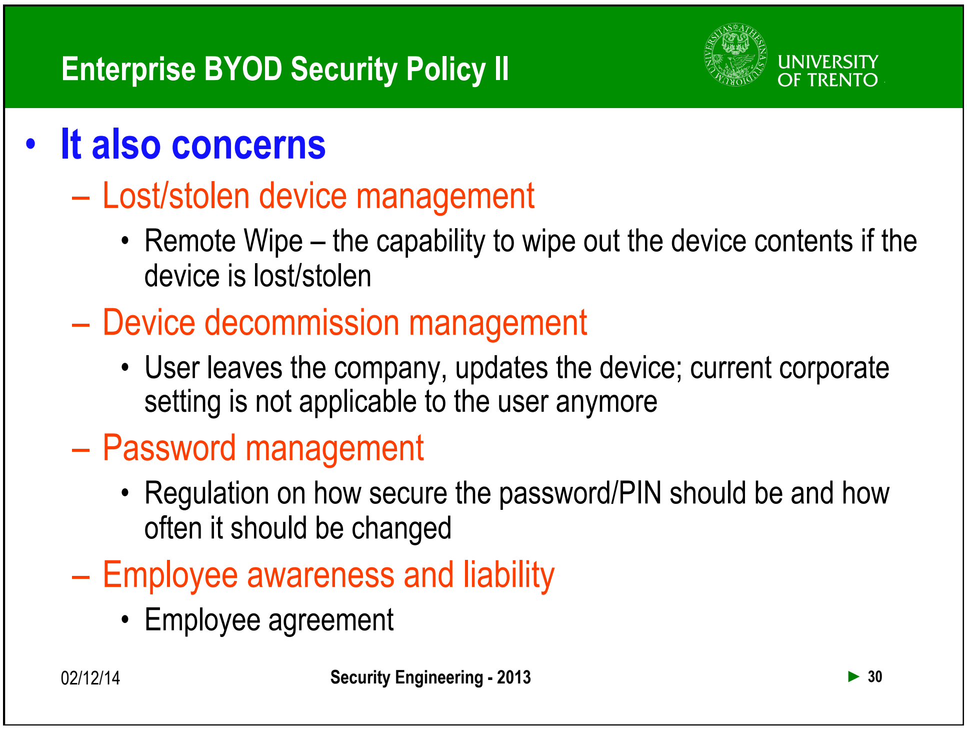
- **After an app got some data, what happens next?**
- **W. Enck et al 'TaintDroid: an information flow tracking system for realtime privacy monitoring on smartphones', in OSDI-2010**
  - TaintDroid is a system for dynamic taint tracking for Android. automatically labels data from privacy-sensitive sources (device ID, location, phone number, etc) and transitively applies labels as sensitive data propagates through program variables, files and inter-process messages
  - When tainted data is sent over the network, TaintDroid logs this fact
- **The authors share the study of 30 popular apps, some of them indeed misuse sensitive data**
  - 2 apps send SIM card ID
  - 15 apps send location data to ad servers
  - None of the apps tells this in the EULA



The Enterprise BYOD Policy

UNIVERSITY OF TRENTO

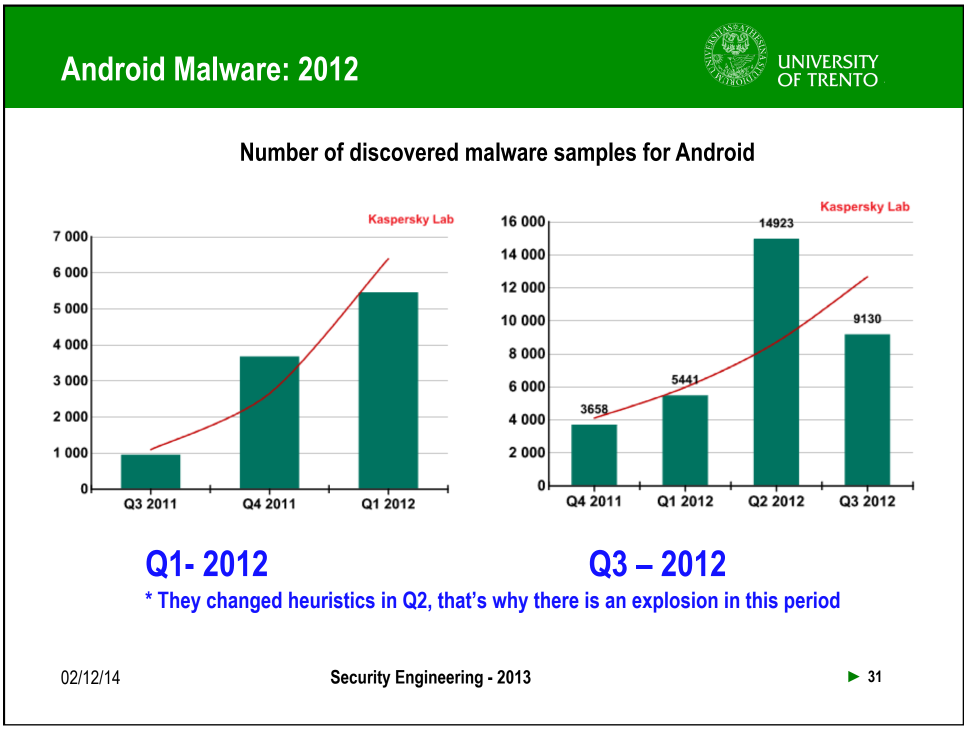
- **Regulates data exchange on device**
  - How private and corporate part interplay
    - Full separation?
    - BYOA?
- **Regulates acceptable devices and settings**
  - Which applications can be installed
    - Concerns both enterprise apps and private apps
    - White and black lists of apps
  - Device management and security mechanisms
    - Which antivirus software is installed, how often the new versions of the OS are installed
    - App scanning/rewriting
  - Types of accepted devices
    - OS type and version dependent



Enterprise BYOD Security Policy II

UNIVERSITY OF TRENTO

- **It also concerns**
  - Lost/stolen device management
    - Remote Wipe – the capability to wipe out the device contents if the device is lost/stolen
  - Device decommission management
    - User leaves the company, updates the device; current corporate setting is not applicable to the user anymore
  - Password management
    - Regulation on how secure the password/PIN should be and how often it should be changed
  - Employee awareness and liability
    - Employee agreement



Android Malware: 2012

UNIVERSITY OF TRENTO

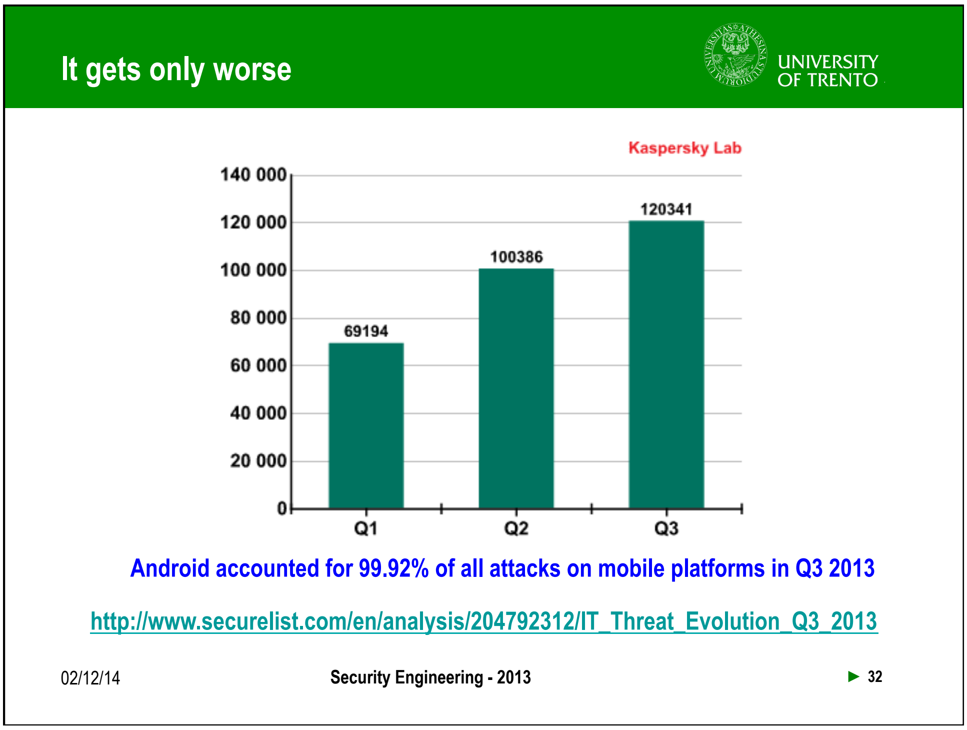
Number of discovered malware samples for Android

Q1- 2012

Q3 – 2012

* They changed heuristics in Q2, that's why there is an explosion in this period

02/12/14 Security Engineering - 2013 ► 31



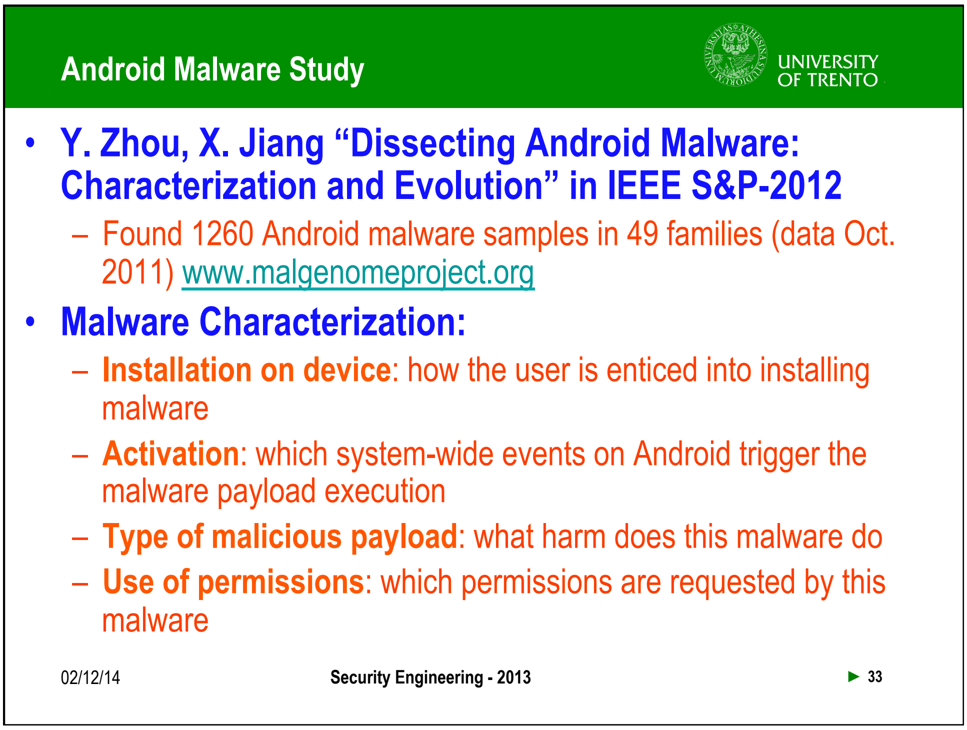
It gets only worse

UNIVERSITY OF TRENTO

Android accounted for 99.92% of all attacks on mobile platforms in Q3 2013

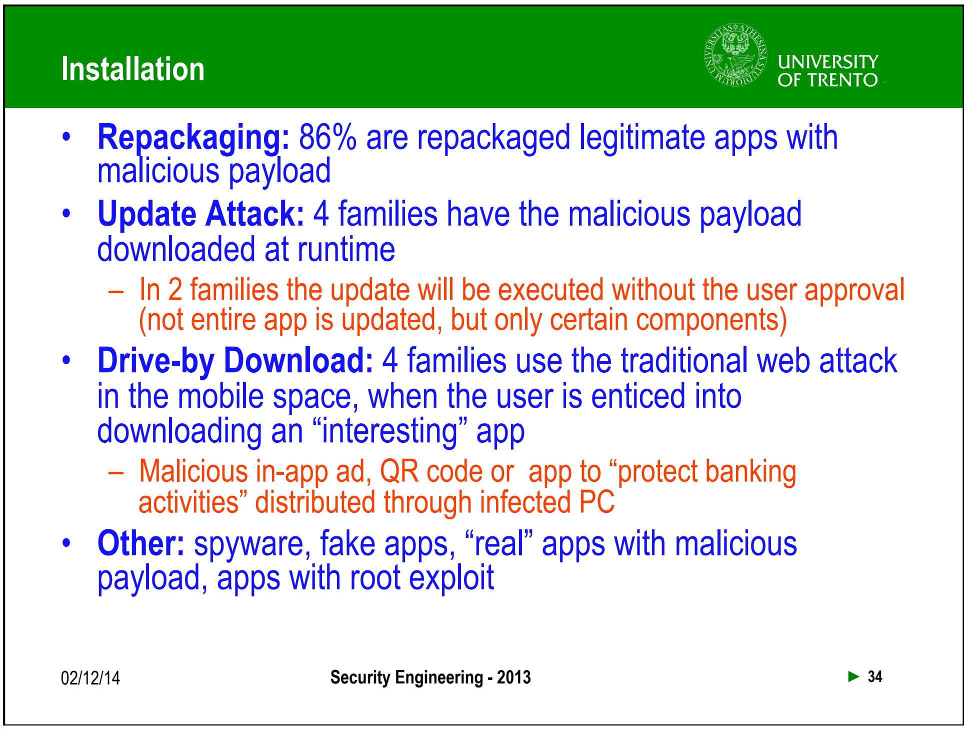
http://www.securelist.com/en/analysis/204792312/IT_Threat_Evolution_Q3_2013

02/12/14 Security Engineering - 2013 ► 32



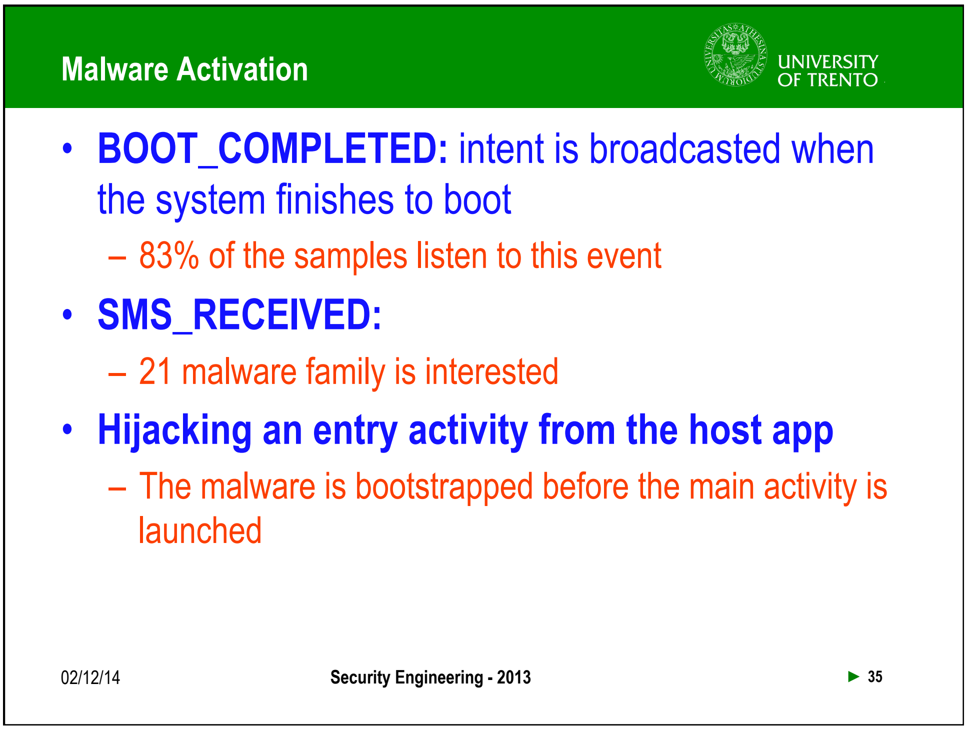
Android Malware Study

- **Y. Zhou, X. Jiang “Dissecting Android Malware: Characterization and Evolution” in IEEE S&P-2012**
  - Found 1260 Android malware samples in 49 families (data Oct. 2011) www.malgenomeproject.org
- **Malware Characterization:**
  - **Installation on device**: how the user is enticed into installing malware
  - **Activation**: which system-wide events on Android trigger the malware payload execution
  - **Type of malicious payload**: what harm does this malware do
  - **Use of permissions**: which permissions are requested by this malware



Installation

- **Repackaging:** 86% are repackaged legitimate apps with malicious payload
- **Update Attack:** 4 families have the malicious payload downloaded at runtime
  - In 2 families the update will be executed without the user approval (not entire app is updated, but only certain components)
- **Drive-by Download:** 4 families use the traditional web attack in the mobile space, when the user is enticed into downloading an “interesting” app
  - Malicious in-app ad, QR code or app to “protect banking activities” distributed through infected PC
- **Other:** spyware, fake apps, “real” apps with malicious payload, apps with root exploit

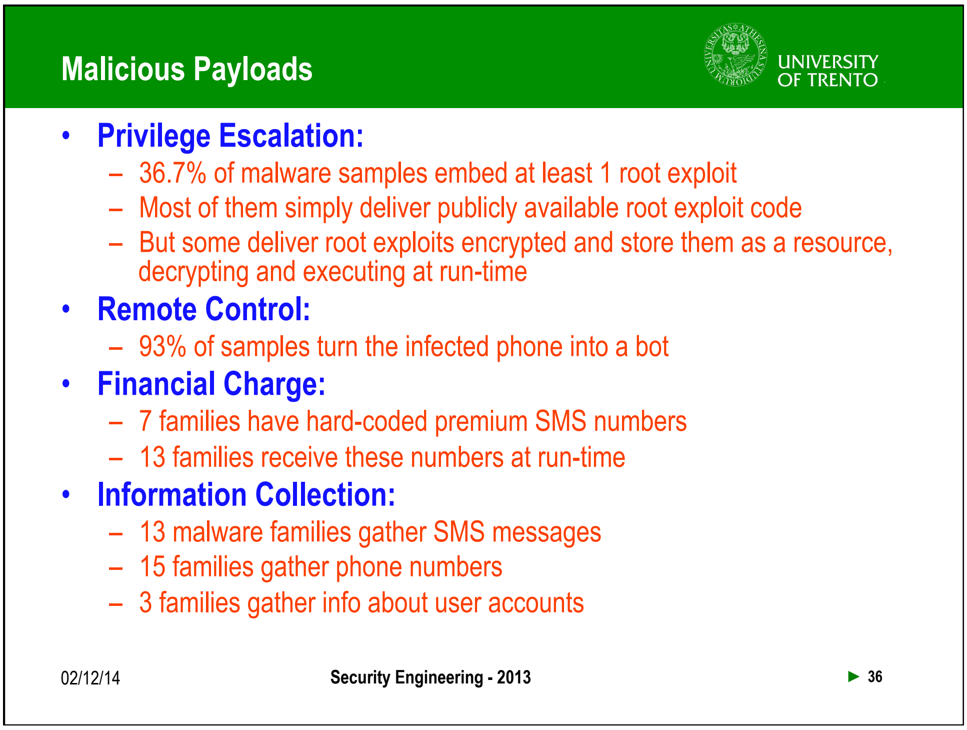


Malware Activation

UNIVERSITY OF TRENTO

- **BOOT_COMPLETED:** intent is broadcasted when the system finishes to boot
  - 83% of the samples listen to this event
- **SMS_RECEIVED:**
  - 21 malware family is interested
- **Hijacking an entry activity from the host app**
  - The malware is bootstrapped before the main activity is launched

02/12/14 **Security Engineering - 2013** ► 35

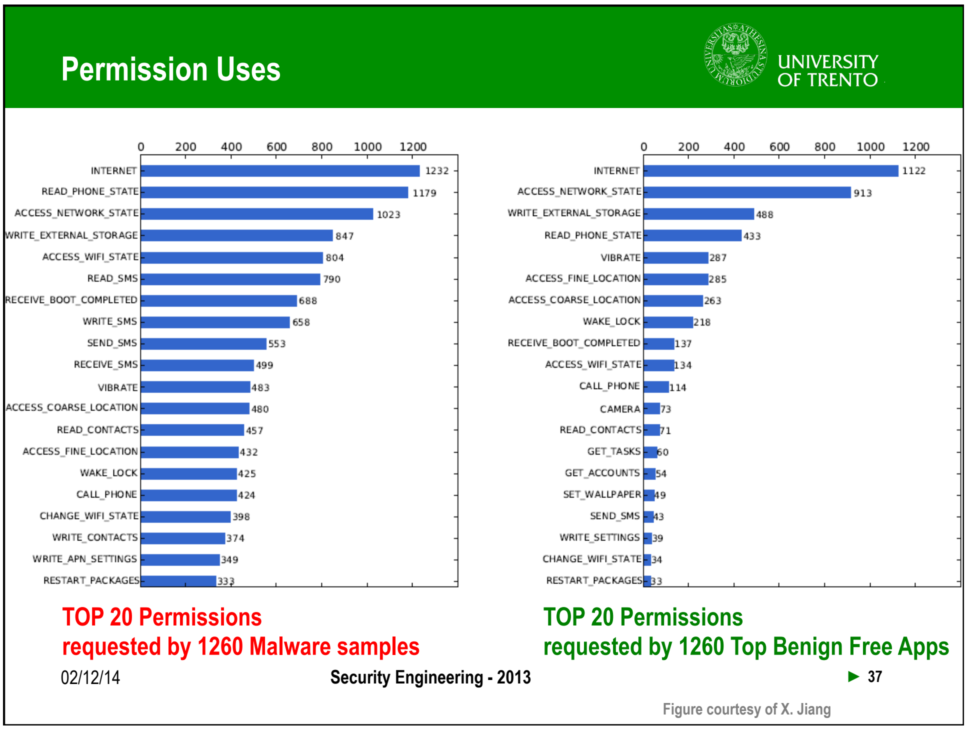


Malicious Payloads

UNIVERSITY OF TRENTO

- **Privilege Escalation:**
  - 36.7% of malware samples embed at least 1 root exploit
  - Most of them simply deliver publicly available root exploit code
  - But some deliver root exploits encrypted and store them as a resource, decrypting and executing at run-time
- **Remote Control:**
  - 93% of samples turn the infected phone into a bot
- **Financial Charge:**
  - 7 families have hard-coded premium SMS numbers
  - 13 families receive these numbers at run-time
- **Information Collection:**
  - 13 malware families gather SMS messages
  - 15 families gather phone numbers
  - 3 families gather info about user accounts

02/12/14 **Security Engineering - 2013** ► 36



Permission Uses

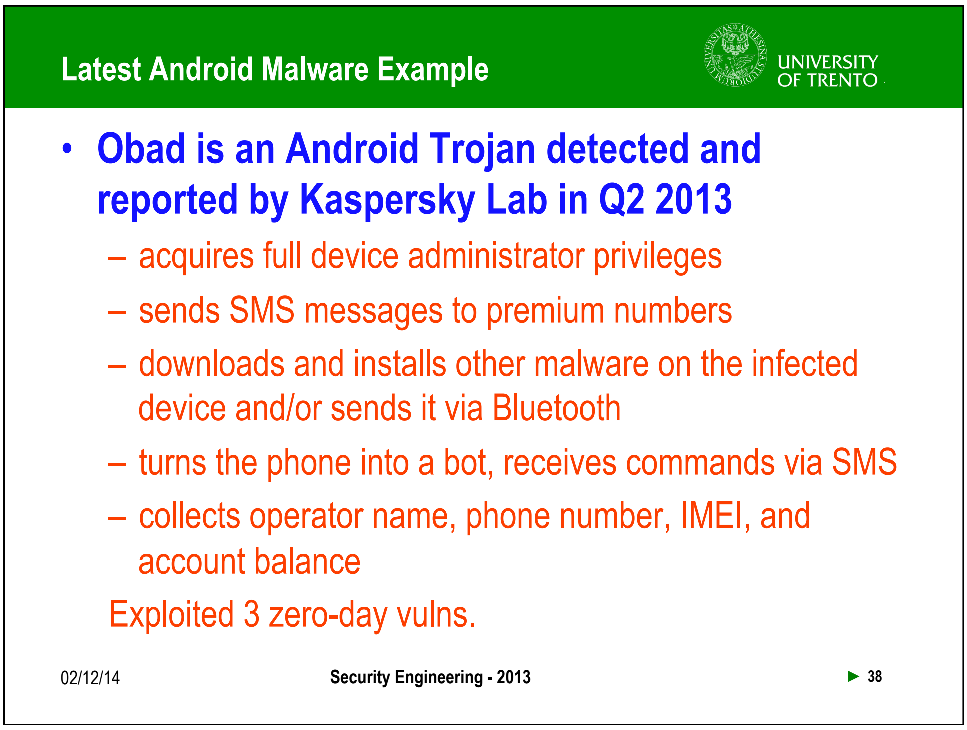
UNIVERSITY OF TRENTO

TOP 20 Permissions
requested by 1260 Malware samples

TOP 20 Permissions
requested by 1260 Top Benign Free Apps

02/12/14 Security Engineering - 2013 ► 37

Figure courtesy of X. Jiang



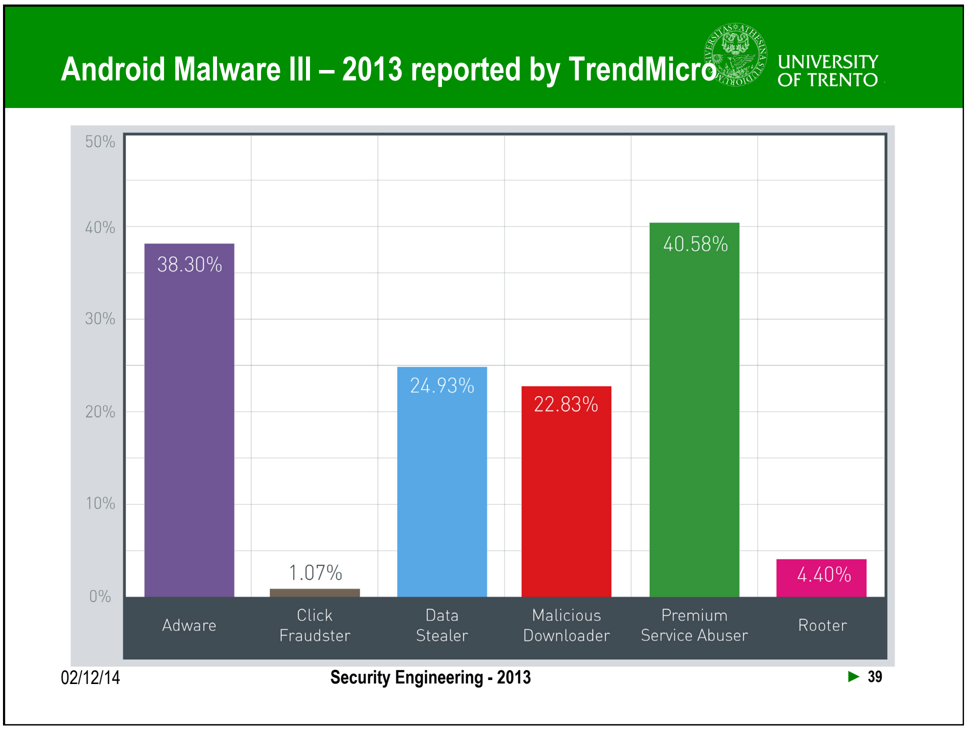
Latest Android Malware Example

UNIVERSITY OF TRENTO

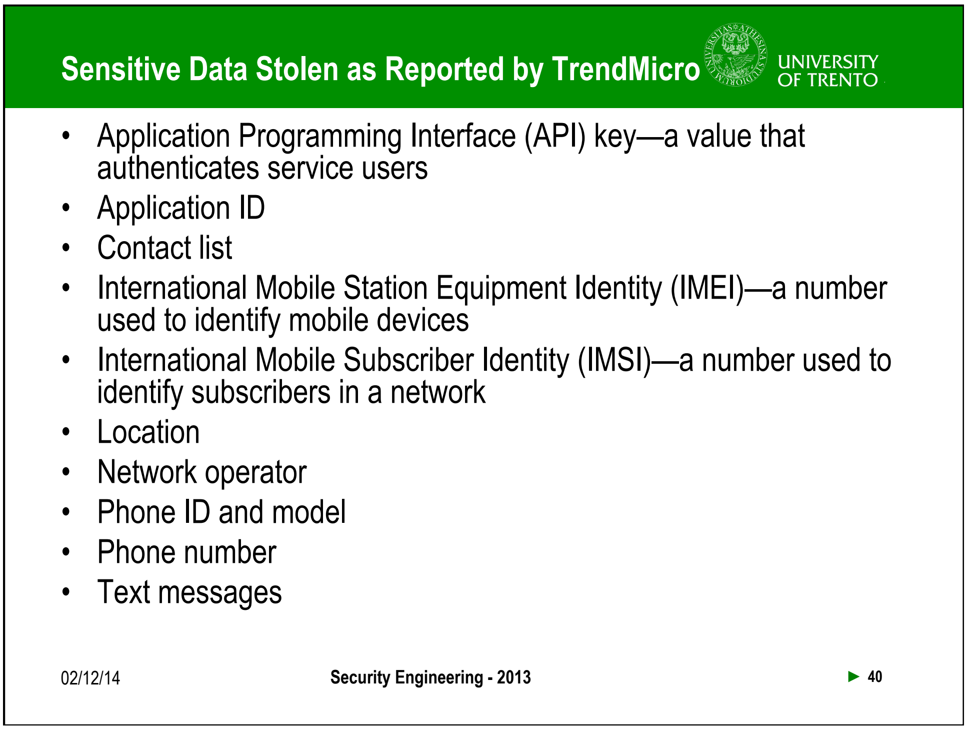
- Obad is an Android Trojan detected and reported by Kaspersky Lab in Q2 2013
  - acquires full device administrator privileges
  - sends SMS messages to premium numbers
  - downloads and installs other malware on the infected device and/or sends it via Bluetooth
  - turns the phone into a bot, receives commands via SMS
  - collects operator name, phone number, IMEI, and account balance

  Exploited 3 zero-day vulns.

02/12/14 Security Engineering - 2013 ► 38

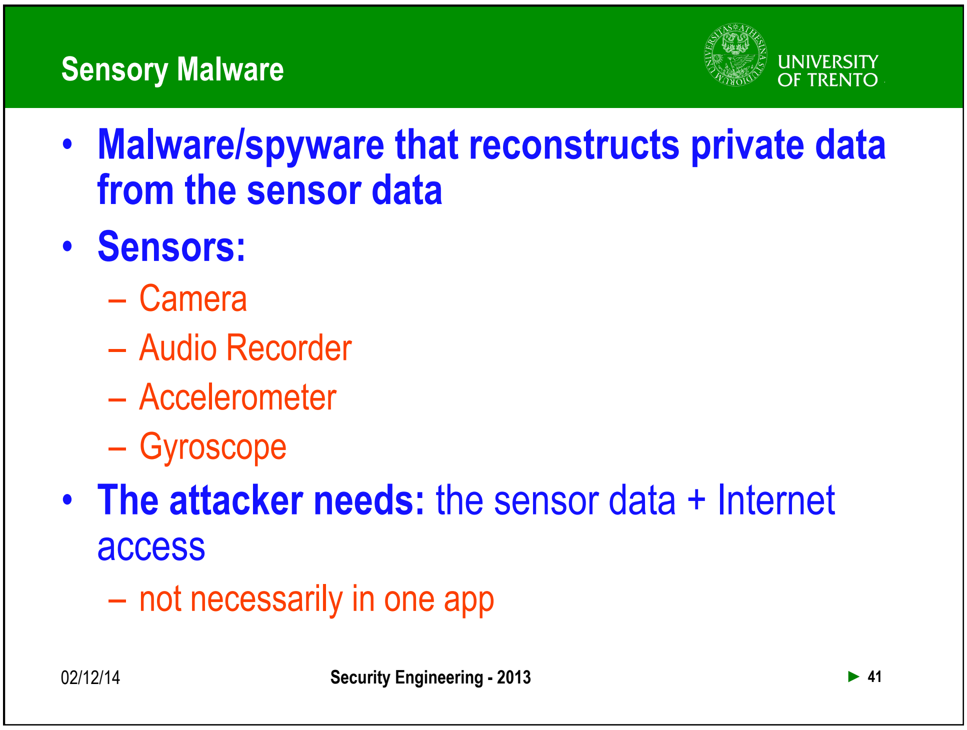


Android Malware III – 2013 reported by TrendMicro



Sensitive Data Stolen as Reported by TrendMicro

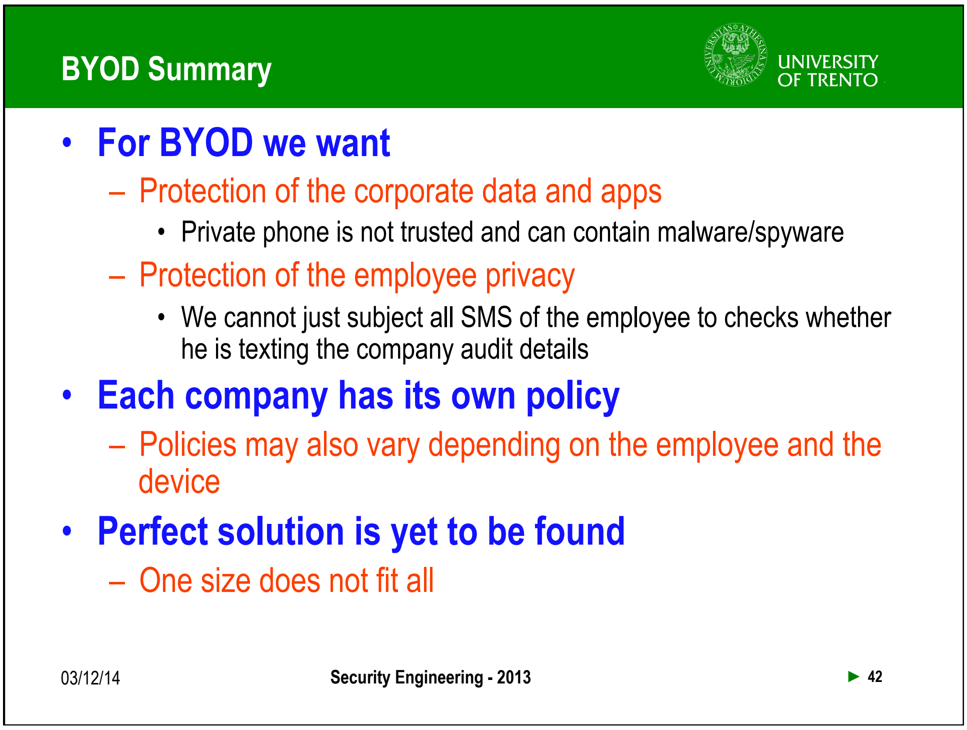
- Application Programming Interface (API) key—a value that authenticates service users
- Application ID
- Contact list
- International Mobile Station Equipment Identity (IMEI)—a number used to identify mobile devices
- International Mobile Subscriber Identity (IMSI)—a number used to identify subscribers in a network
- Location
- Network operator
- Phone ID and model
- Phone number
- Text messages



Sensory Malware

UNIVERSITY OF TRENTO

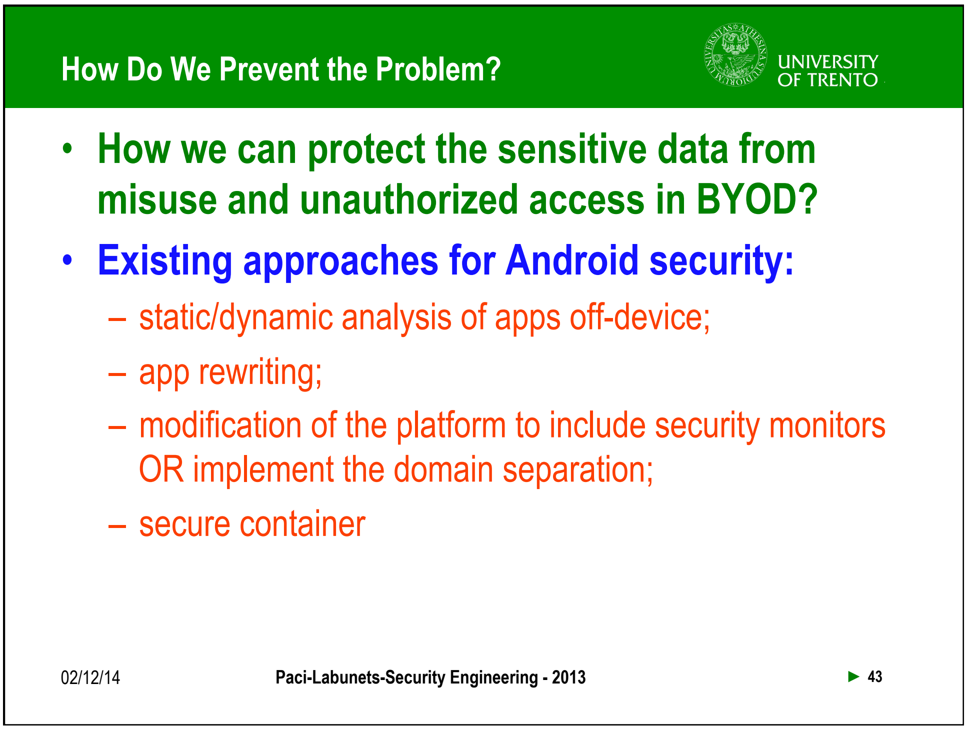
- **Malware/spyware that reconstructs private data from the sensor data**
- **Sensors:**
  - Camera
  - Audio Recorder
  - Accelerometer
  - Gyroscope
- **The attacker needs:** the sensor data + Internet access
  - not necessarily in one app



BYOD Summary

UNIVERSITY OF TRENTO

- **For BYOD we want**
  - Protection of the corporate data and apps
    - Private phone is not trusted and can contain malware/spyware
  - Protection of the employee privacy
    - We cannot just subject all SMS of the employee to checks whether he is texting the company audit details
- **Each company has its own policy**
  - Policies may also vary depending on the employee and the device
- **Perfect solution is yet to be found**
  - One size does not fit all

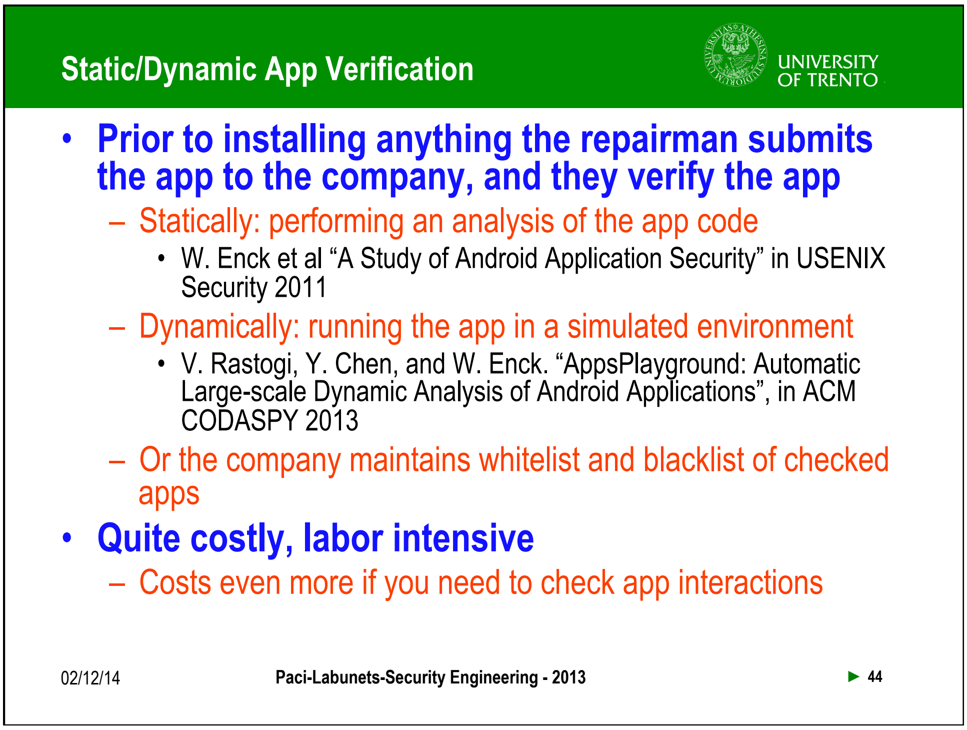


How Do We Prevent the Problem?

UNIVERSITY OF TRENTO

- How we can protect the sensitive data from misuse and unauthorized access in BYOD?
- Existing approaches for Android security:
  - static/dynamic analysis of apps off-device;
  - app rewriting;
  - modification of the platform to include security monitors OR implement the domain separation;
  - secure container

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 43

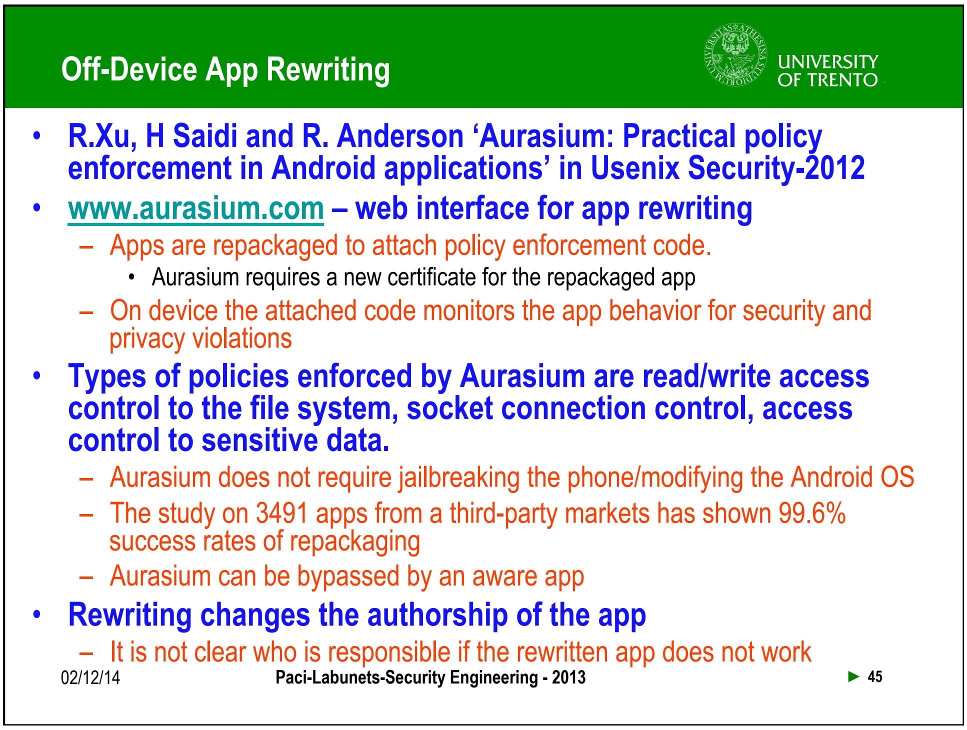


Static/Dynamic App Verification

UNIVERSITY OF TRENTO

- Prior to installing anything the repairman submits the app to the company, and they verify the app
  - Statically: performing an analysis of the app code
    - W. Enck et al “A Study of Android Application Security” in USENIX Security 2011
  - Dynamically: running the app in a simulated environment
    - V. Rastogi, Y. Chen, and W. Enck. “AppsPlayground: Automatic Large-scale Dynamic Analysis of Android Applications”, in ACM CODASPY 2013
  - Or the company maintains whitelist and blacklist of checked apps
- Quite costly, labor intensive
  - Costs even more if you need to check app interactions

02/12/14 Paci-Labunets-Security Engineering - 2013 ► 44

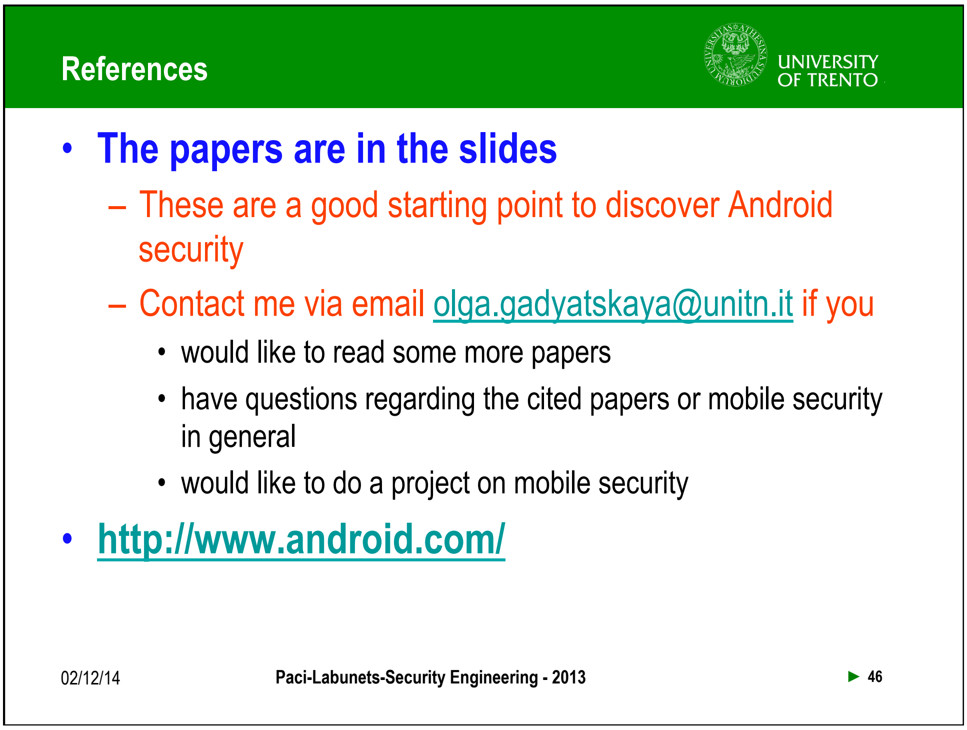


Off-Device App Rewriting

UNIVERSITY OF TRENTO

- **R.Xu, H Saidi and R. Anderson 'Aurasium: Practical policy enforcement in Android applications' in Usenix Security-2012**
- **www.aurasium.com – web interface for app rewriting**
  - Apps are repackaged to attach policy enforcement code.
    - Aurasium requires a new certificate for the repackaged app
  - On device the attached code monitors the app behavior for security and privacy violations
- **Types of policies enforced by Aurasium are read/write access control to the file system, socket connection control, access control to sensitive data.**
  - Aurasium does not require jailbreaking the phone/modifying the Android OS
  - The study on 3491 apps from a third-party markets has shown 99.6% success rates of repackaging
  - Aurasium can be bypassed by an aware app
- **Rewriting changes the authorship of the app**
  - It is not clear who is responsible if the rewritten app does not work

02/12/14 **Paci-Labunets-Security Engineering - 2013** ► **45**



References

UNIVERSITY OF TRENTO

- **The papers are in the slides**
  - These are a good starting point to discover Android security
  - Contact me via email olga.gadyatskaya@unitn.it if you
    - would like to read some more papers
    - have questions regarding the cited papers or mobile security in general
    - would like to do a project on mobile security
- **http://www.android.com/**

02/12/14 **Paci-Labunets-Security Engineering - 2013** ► **46**