

# Post-GDPR Threat Hunting on Android Phones: Dissecting OS-level Safeguards of User-unresettable Identifiers

核心设计UI2I工具、用六类UIIs:

Serial number、Device ID、ICCID、IMSI、蓝牙MAC、WIFIMAC)

三个主要问题

- 1个AOSP漏洞被继承 (ICCID) ;
- 厂商扩展时不遵守AOSP隐私准则;
- 厂商白名单机制被绕过。

## I. Introduction

大背景: GDPR等隐私规范; Android10以后采用READ\_PRIVILEGED\_PHONE\_STATE限制用户级程序获取UII, 同时推出可重置UID (例如广告ID)

现有工作: 大多假设操作系统足够安全, 聚焦于程序分析和流量分析等, 从旁路攻击、侧信道攻击等绕过权限获取UII, 而没有从系统级本身研究UII

两个难点:

- 找到所有应用可访问的接口 (枚举)
- 识别UIIs (回归测试、差分分析、UII足够长)

主要发现和贡献 (contribution)

- 从系统级研究UIIs;
- 发现51个漏洞, 其中包含1个AOSP的漏洞且因安卓10兼容性问题补课修复;
- 51个中45个来自未记录的访问渠道 (Access Channel) ;
- 发现厂商白名单机制存在绕过;
- 分析的300个APPs中12个经过未记录的访问渠道 (Access Channel) 进行获取UIIs

II. Background

安卓权限机制

权限标签根据敏感性分三类：

这两类是用户级别，非系统应用程序最多只能获得危险级别的权限。

- 1.最低级（正常保护级）：安装时自动授予；
- 2.危险级别（运行时权限）：运行时需获得用户的同意；
- 3.签名级别（专为特权操作而设计）

Android 对数据法规的回应

GDPR等法规、更严格的安卓权限规则、可重置ID（如广告ID）

III. Android UIIs

**UII定义:** 将可用于直接或间接识别设备的信息视为标识符，将与设备永久绑定的标识符（例如硬件设备ID）称为用户不可重置标识符（UII）。

TABLE I: List of recognized Android UIIs

No.	UII	Category	Permission Updates across Android Versions <sup>†</sup>	Literature		
				Considered Sensitive in	Collection Techniques Used	Validity in Android 11
1	Serial number	Chip & Cellular	v8-9: READ_PHONE_STATE required. ≥v10: READ_PRIVILEGED_PHONE_STATE required.	[57], [59], [68], [69]	Documented APIs: [68] System services: [69]	✗ (since v10) ✓
2	Device ID (IMEI or MEID)		<v10: READ_PHONE_STATE required. ≥v10: READ_PRIVILEGED_PHONE_STATE required.	[7], [14], [20], [37], [46], [54], [56], [57], [58], [59], [60], [68], [69], [74], [77]	Documented APIs: [14], [20], [56], [58], [68] System properties: [7] System services: [69]	✗ (since v10) ✓ ✓
3	ICCID		<v10: READ_PHONE_STATE required. ≥v10: READ_PRIVILEGED_PHONE_STATE required.	[14], [46], [54], [56], [57], [59], [60], [69], [74], [77]	Documented APIs: [14], [56] System services: [69]	✗ (since v10) ✓
4	IMSI		<v10: READ_PHONE_STATE required. ≥v10: READ_PRIVILEGED_PHONE_STATE required.	[14], [20], [46], [54], [56], [57], [58], [59], [60], [69], [77]	Documented APIs: [14], [20] System services: [69]	✗ (since v10) ✓
5	Bluetooth MAC address	Wireless Module	All versions: BLUETOOTH required. ≥v6: Randomization or a fixed return value required. v6-10: ACCESS_COARSE_LOCATION or ACCESS_FINE_LOCATION required. ≥v10: ACCESS_FINE_LOCATION becomes mandatory	[68]	Documented APIs: [68]	✗ (since v6)
6	WiFi MAC address		All versions: ACCESS_WIFI_STATE required. v6-9: Randomization suggested. ≥v10: Randomization becomes mandatory.	[7], [20], [37], [54], [57], [58], [59], [60], [63], [74], [77]	ioctl usage: [58] Documented APIs: [20], [63], [68] Read from /sys/class/net: [63]	✗ (since v6) ✗ (since v10) ✗ (since v7)

<sup>†</sup> The color schemes indicate the permission protection levels: normal, dangerous and signature.

Chip and cellular identifiers: 4个UII

- Serial numbers（序列号）:设备唯一序列号，用于制造商追溯产品售后保修；
- Device ID（设备ID）：注册使用全球移动通信系统（GSM）服务的设备ID是国际移动设备标识（IMEI），而对于码分多址（CDMA）设备，设备ID是移动设备标识符（MEID）。
- SIM card IDs（SIM卡ID）：
  - 集成电路卡ID（ICCID）

- 国际移动用户身份 (IMSI)

Wireless module identifiers (无线模块标识符): 2个UUI

- Bluetooth MAC address (蓝牙MAC地址)
- WiFi MAC address (WiFiMAC地址)

## IV. Approach 方法概述

U2-I2模型目标:

- 检测非系统应用越权访问UUI接口;
- 识别UUI实际权限和安卓规定不一致情况;
- 识别厂商扩展系统后存在的不符合AOSP隐私规范的情况;

威胁模型 (Threat Model) :

- UUI收集器为非系统应用程序 (或作为非系统应用程序软件库) ;
- 应用程序开放Internet权限, 未公开的Access Channel不需要其他的权限、已公开的Access Channel可以获取早期Android版本中开放的权限;
- 攻击者可离线分析检测, 即攻击者可对设备root、对固件逆向调试等。

难点 (Challenges) :

- 识别未公开的Access channel;
- 自动化评估测试过程;
- 精准识别新的UUIs; (差分分析)

## Assessing Document Channels (评估公开的访问渠道)

U2-I2在Android开发者文档中搜索到9个用于获取六种已知的UUIs的API (下表前两列)

TABLE II: Documented APIs to access six known UIIs and corresponding system services

UII	Developers API Name(s)	System Service(s)
Serial	android.os.Build.getSerial()	device_identifiers
Device ID/IMEI/MEID	android.telephony.TelephonyManager.getImei()	phone
	android.telephony.TelephonyManager.getDeviceId()	phone
	android.telephony.TelephonyManager.getMeid()	phone
ICCID	android.telephony.TelephonyManager.getSimSerialNumber()	phone
	android.telephony.SubscriptionInfo.getIccId() <sup>†</sup>	isub
IMSI	android.telephony.TelephonyManager.getSubscriberId()	phone
Bluetooth MAC	android.bluetooth.BluetoothAdapter.getAddress()	bluetooth_manager
WiFi MAC	android.net.wifi.WifiInfo.getMacAddress() <sup>‡</sup>	wifi

<sup>†</sup> To invoke that API, the app must obtain an instance of *SubscriptionInfo* through another API *SubscriptionManager.getActiveSubscriptionInfo()* at first.  
<sup>‡</sup> To invoke that API, the app must obtain an instance of *ConnectionInfo* through another API *WifiManager.getConnectionInfo()* at first.

U2-I2开发了一个非系统应用，采用软件回归测试检查系统升级或厂商自定义操作系统时，是否引入了新的漏洞（UIIs）。这个测试应用程序包括了跨版本验证和跨制造商验证。

测试期间检测两类错误：

- 旧版遗留权限：以 `getSerial()` 方法为例，自 Android 10 起，该方法仅限系统应用使用。因此，在 Android 10 或 11 中，任何非系统应用即使具有 `READ_PHONE_STATE` 权限（适用于Android8和9），也不应允许访问设备序列号。
- 失去标识性：Android 文档声称某些API会对非系统应用返回随机化的 UIIs，例如蓝牙 MAC 地址。这意味着，当非系统应用请求这些敏感信息时，系统应该提供**随机化的值**以保护用户隐私。（这里的工作就是验证API是否进行了随机化）

## Discovering And Assessing Undocumented Access Channels(发现和评估未公开的访问渠道)

U2-I2 提出了一种三阶段方法来探索和评估未记录的访问渠道。

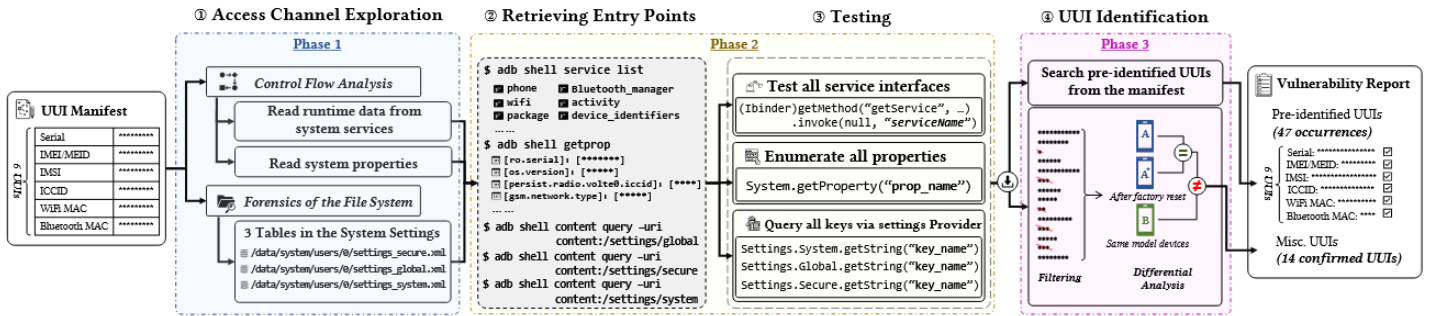


Fig. 1: The workflow of our UII exploration and assessment through undocumented channels

- 1. Access Channel Exploration: System setting、System Service、System properties
  - i. 静态控制流分析：

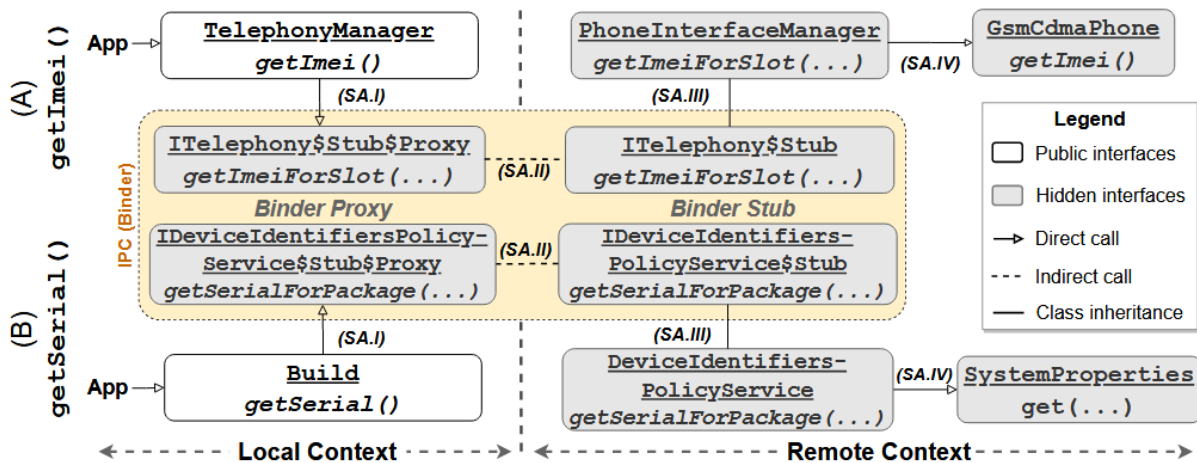


Fig. 2: Visualization of two typical control flows starting from documented APIs

ii. 文件系统取证 (Forensics of the file systems) :

## 2. Automatic Access Channel Testing

i. Retrieving Entry Points: 如图1所示, 使用ADB检索所有完整服务列表和系统属性Keys

ii. Testing: 通过开发的APP进行评估。

o System services:

U2-I2 通过 Java 反射采用了一种“黑客方式”来绕过 `getSystemService()` 的权限检查。

o System properties and system settings:

## 3. 厂商自定义的标识:

o 过滤掉长度不足的值 (包含小于 4 个十六进制数字的字符串)

o 排除跨设备相同的值、保留设备出厂设置前后保持不变的字段

# Evaluation And Landscape Of UUI Protection In Android Phones

- RQ1. What is U2-I2's performance in identifying UUI mishandling issues? Based on U2-I2's findings, what is the status quo of UUI protection in the latest Android phones? Do their OSes comply with the up-to-date privacy policy of AOSP?
- RQ2. Based on U2-I2's findings, what UUIs are potentially exposed to non-system apps, and what are the typical exfiltration points?
- RQ3. Have our identified access channels of UUI access already been (ab)used by apps in the wild?

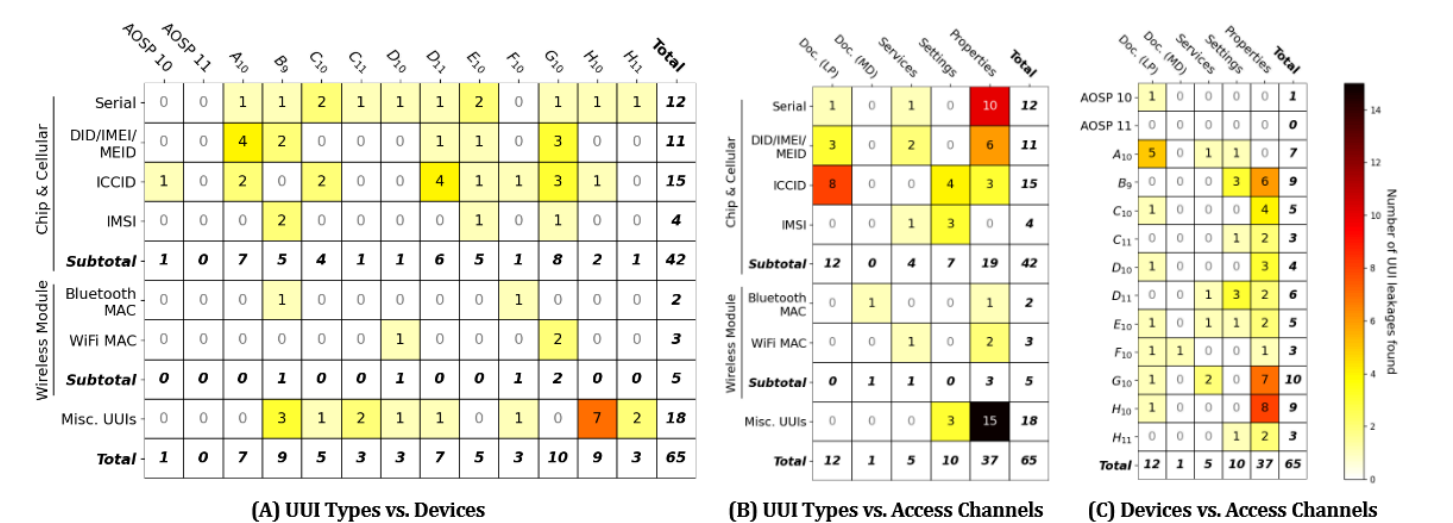
## RQ1: 操作系统级 UUI 保护现状

AOSP在所有设备中漏洞数量最少, 仅发现一个漏洞 (即 CVE-2021-04283)。

与 AOSP 相比，OEM 操作系统的 UUI 处理不当问题更多。基于 Android 10 安装的操作系统继承了 AOSP 的漏洞（即 CVE-20210428）。文中厂商 A 在其 API 中错误地公开了序列号和设备 ID 的 UUI、厂商 F 未在返回的蓝牙 MAC 地址上随机化。并且厂商白名单机制存在安全风险。

RQ1 中的发现：  
U2-I2 能够检测已知 UUI 和 OEM 定制的 UUI 的泄漏。我们的研究发现，UUI 处理不当在流行的 Android 手机中普遍存在。从 9 家制造商的 13 种型号的测试设备中，确定了 51 个漏洞。它们导致 65 个 UUI 泄漏。除 AOSP 11 外，每个测试设备都至少检测到一次泄漏，最高为 10，平均值为 6，中位数为 5。OEM 操作系统的问题比 AOSP 多。主要原因在于未记录的访问渠道。3 家 OEM 操作系统将 UUI 访问应用列入白名单，从其白名单机制中发现安全风险。

RQ2: UUI 泄漏漏洞的特征描述


















Legend for documented channels: Doc. (LP): Documented (Legacy Permission), Doc. (MD): Documented (Missing De-identification)  
Fig. 4: Statistics of the occurrence of UUI leakages detected in our assessment, counted by UUI types and devices (A), by UUI types and access channels (B), and by devices and access channels (C)

RQ2中的发现：  
绝大多数泄露的UUIs属于芯片和蜂窝（Chip and cellular identifiers）类别。系统属性是主要的泄露点，占总泄露量的50%以上。其余两个未记录的访问渠道，即系统服务和系统设置，分别占5个和10个泄露。所有新发现的杂项UUIs都是通过系统设置和系统属性发现的。

# RQ3: 现有应用程序收集 UII

TABLE VI: List of apps found potential UII collection

#	App Package Name	Market & Downloads*	Involved UIIs	Affected Devices	Access Channel
1	com.kwai.m2u	 81mil+	#2, Misc. <sup>†</sup>	<u>B9,C10,11,D10,11</u>	Properties
2	com.kwai.videoeditor	 86mil+	#2, Misc. <sup>†</sup>	<u>B9,C10,11,D10,11</u>	Properties
3	com.lbe.parallel.intl	 100mil+	#4	<u>G10</u>	Services
4	com.liulishuo.engzo	 116mil+	#2	<u>G10</u>	Properties
5	com.oppo.store	 3mil+	Misc. <sup>†</sup>	<u>C10,11, D10,11</u>	Properties
6	com.renrendai.haohuan	 25mil+	#2	<u>G10</u>	Properties
7	com.tencent.qqimsecure	 682mil+	#4	Unidentified	Settings
8	com.tencent.wifimanager	 192mil+	#4	Unidentified	Settings
9	com.wuba.zhuanzhuan	 234mil+	#2	<u>G10</u>	Properties
10	com.xunmeng.merchant	 50mil+	#1	<u>H10,11</u>	Properties
11	com.xunmeng.pinduoduo	 5bil+/  500k+	#1	<u>H10,11</u>	Properties
12	ru.yandex.searchplugin	 100mil+	#2	<u>G10</u>	Properties

\* The apps labelled with  are downloaded from Xiaomi App Store, with the statistics provided by Kuchuan.com, those with  are downloaded from Google Play Store.

<sup>†</sup> The caught UII access is through requesting the system property `gsm.serial`.

RQ3中的发现：  
已识别的UII收集渠道已被主流应用商店中的现有热门应用使用。然而，尚未观察到广泛的UII收集行为，300个应用中仅有12个表现出相关行为。所有三种访问渠道都被用于UII收集。

## Whiteness Issues（白名单机制）

白名单中列出的应用程序可以绕过其权限控制。显然，这种白名单机制是可被利用的。它完全依赖于包名称进行访问控制，而不是应用程序签名，恶意应用程序可以通过简单地使用白名单中的任何软件包名称来欺骗此机制。

## Dissscussion（讨论）

## Related Work（相关工作）

- 访问控制（Access Control）
- 应用程序级数据收集（App-Level data gathering）