Log: It’s Big, It’s Heavy, It’s Filled with Personal Data!

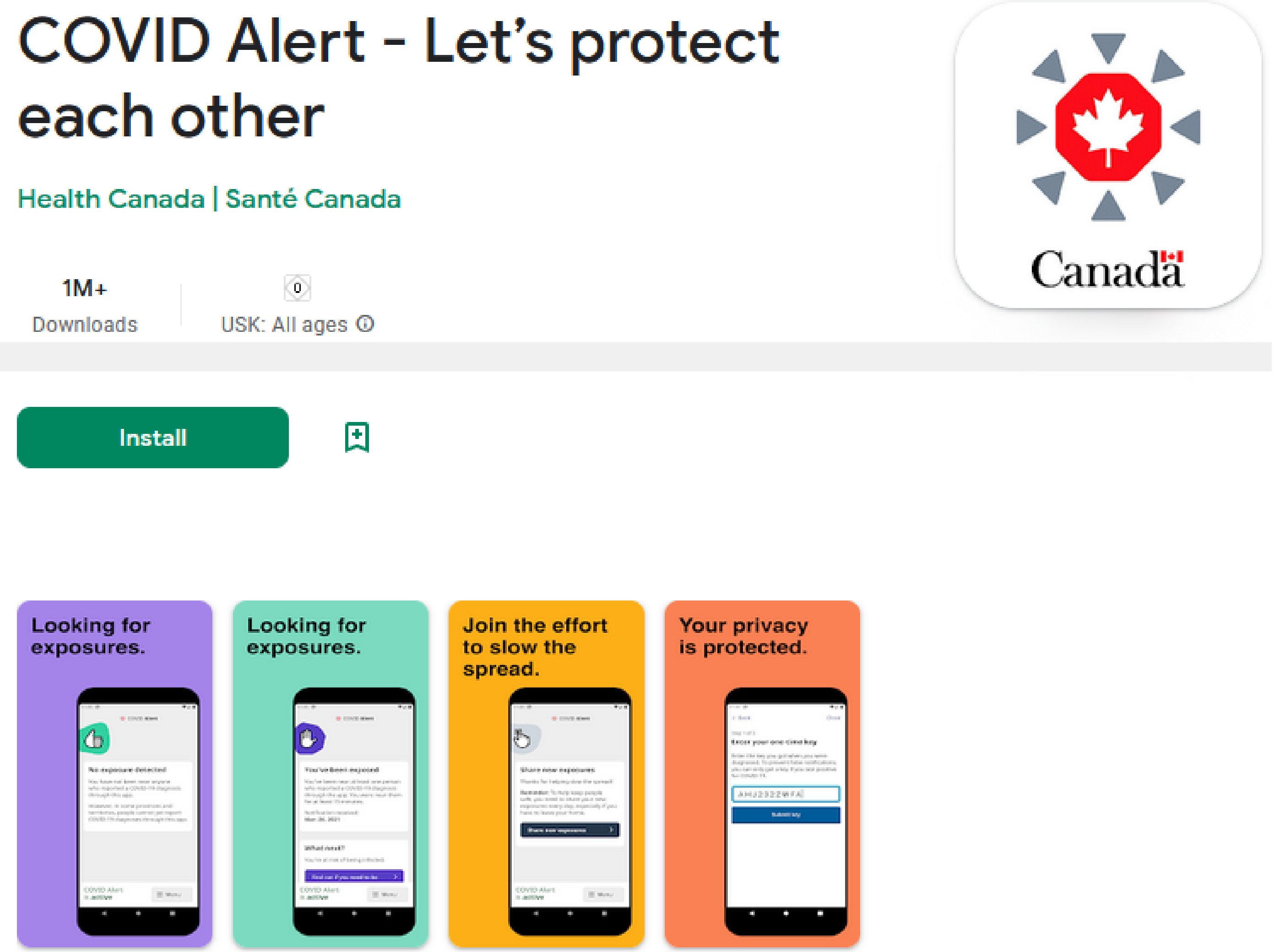
Measuring the Logging of Sensitive Information in the Android Ecosystem

Allan Lyons Julien Gamba Austin Shawaga

Joel Reardon Juan Tapiador Serge Egelman Narseo Vallina-Rodríguez

UNIVERSITY OF CALGARY · institute imdea networks · uc3m Universidad Carlos III de Madrid · AppCensus · Berkeley UNIVERSITY OF CALIFORNIA · ICSI

Google-Apple Exposure Notification



COVID Alert - Let's protect each other

Health Canada | Santé Canada

1M+ Downloads

0 USK: All ages

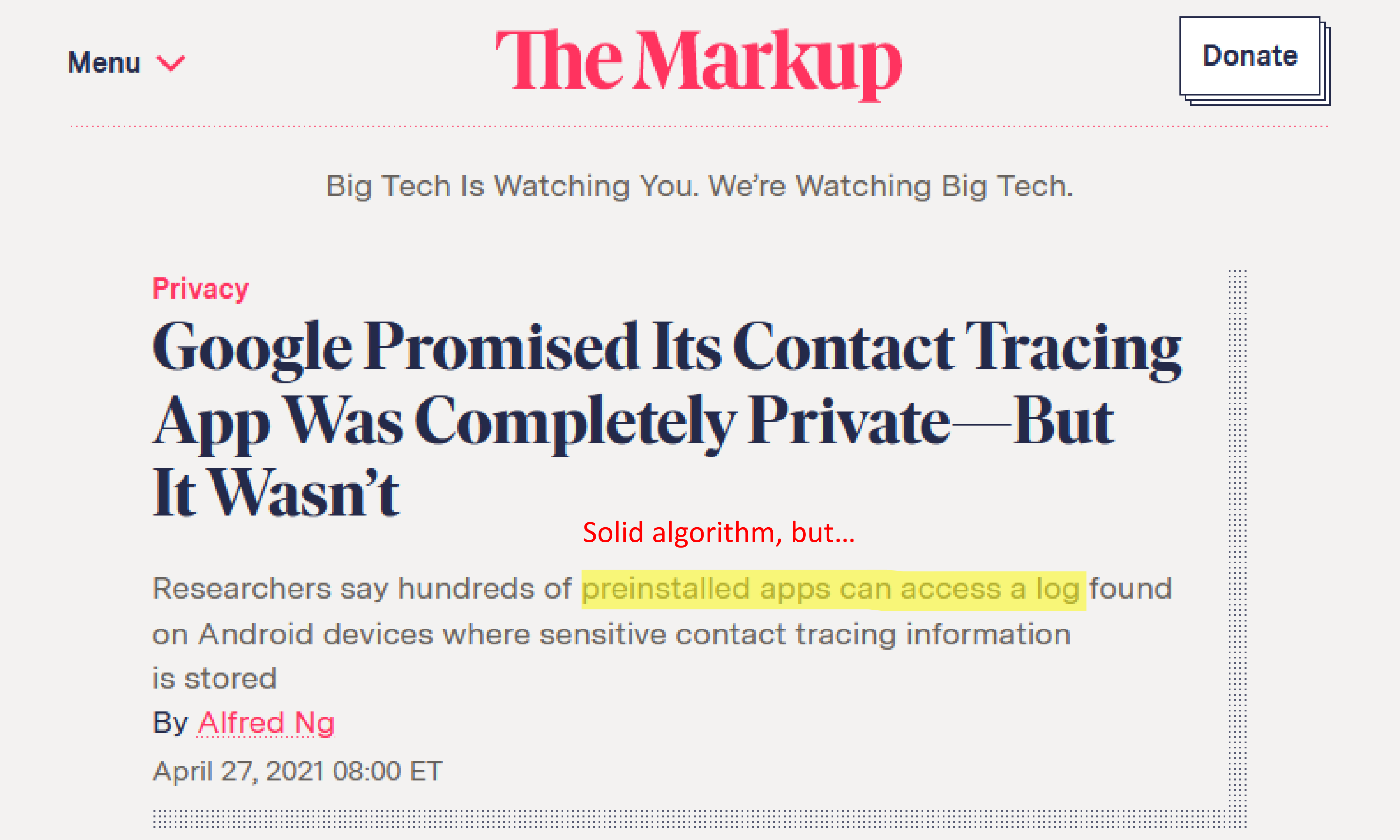
Install

Looking for exposures.

Looking for exposures.

Join the effort to slow the spread.

Your privacy is protected.



The Markup

Big Tech Is Watching You. We're Watching Big Tech.

Privacy

Google Promised Its Contact Tracing App Was Completely Private—But It Wasn't

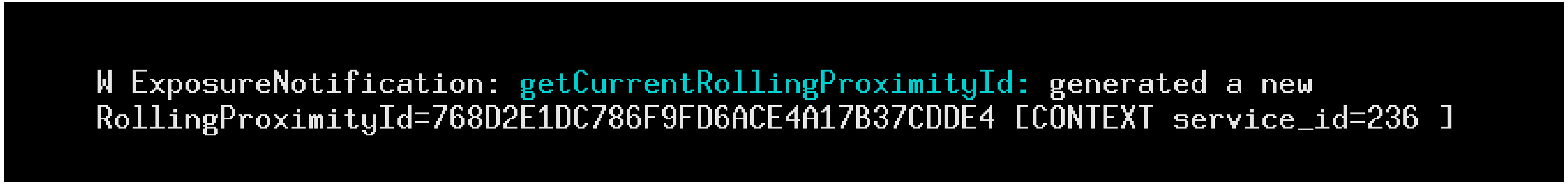
Solid algorithm, but…

Researchers say hundreds of preinstalled apps can access a log found on Android devices where sensitive contact tracing information is stored

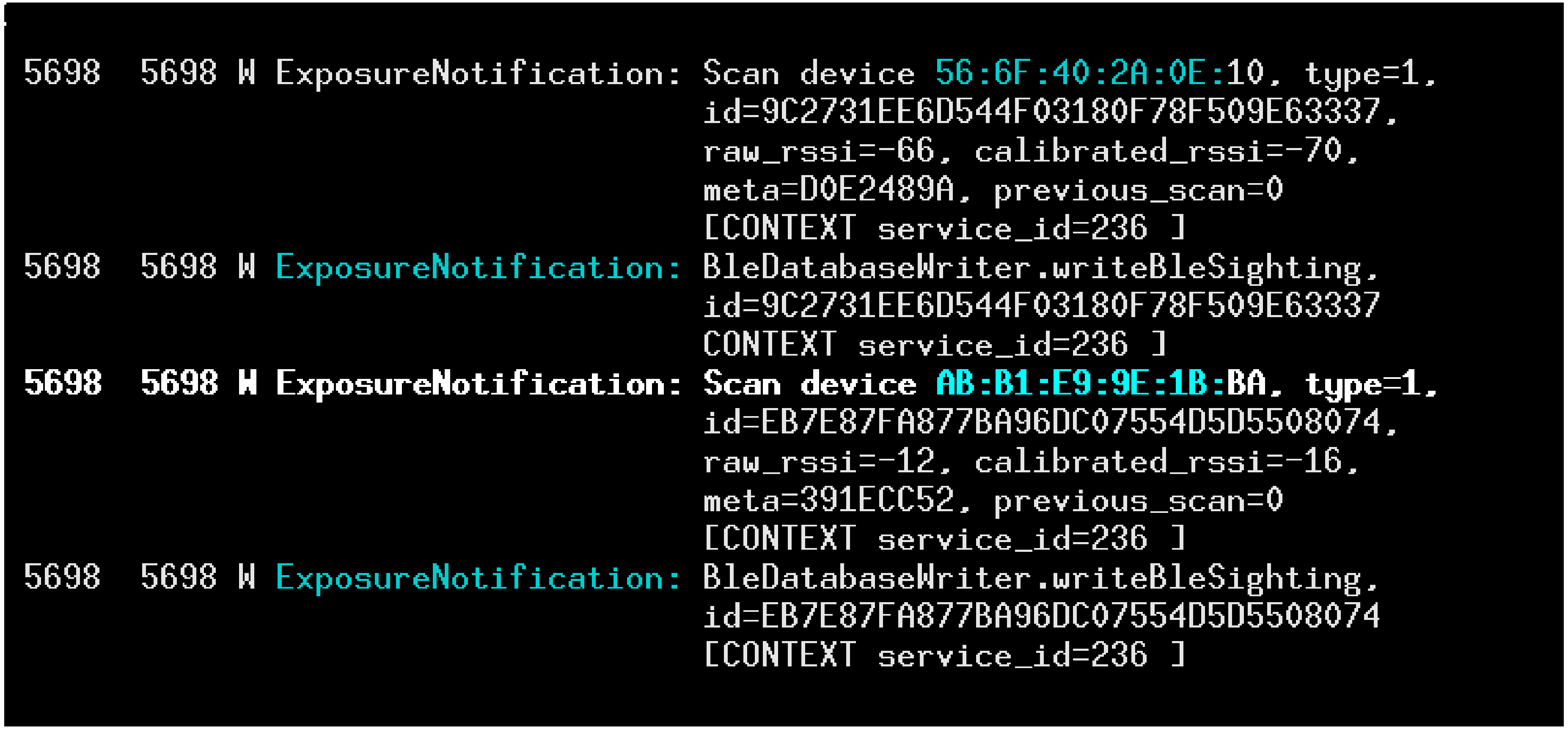
By Alfred Ng

April 27, 2021 08:00 ET

GAEN Logged “Anonymous” Identifiers

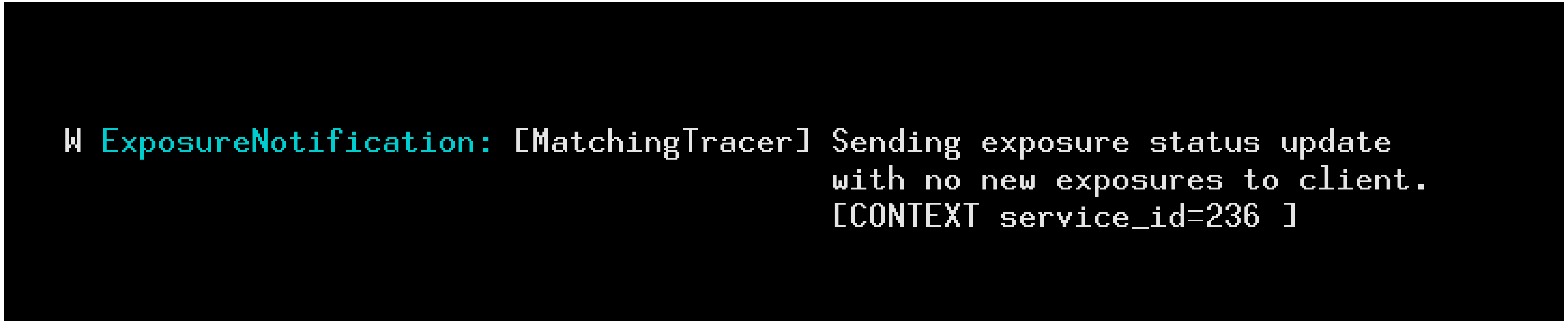


```
W ExposureNotification: getCurrentRollingProximityId: generated a new
RollingProximityId=768D2E1DC786F9FD6ACE4A17B37CDDE4 [CONTEXT service_id=236 ]
```

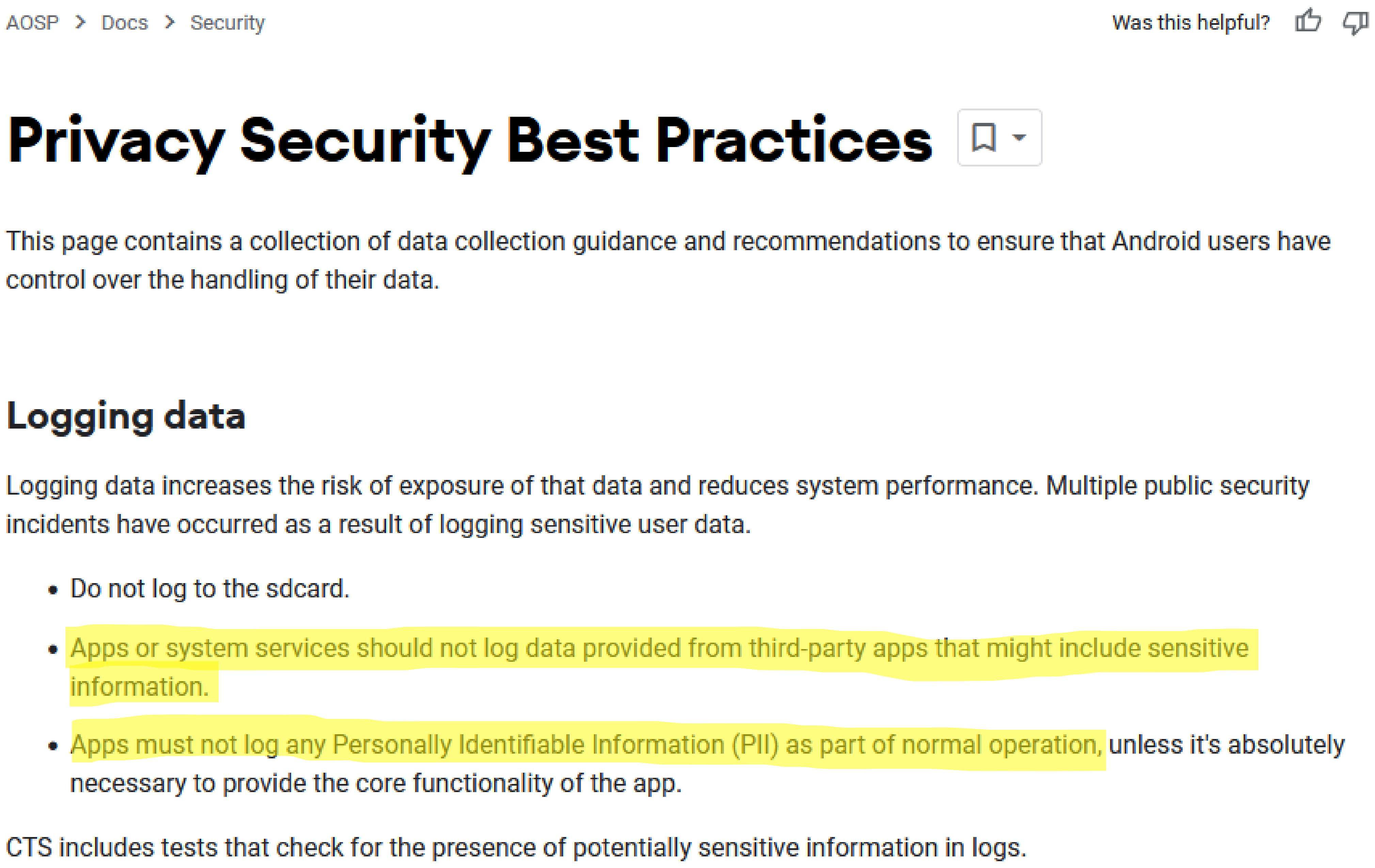


```
5698  5698 W ExposureNotification: Scan device 56:6F:40:2A:0E:10, type=1,
                                    id=9C2731EE6D544F03180F78F509E63337,
                                    raw_rssi=-66, calibrated_rssi=-70,
                                    meta=D0E2489A, previous_scan=0
                                    [CONTEXT service_id=236 ]
5698  5698 W ExposureNotification: BleDatabaseWriter.writeBleSighting,
                                    id=9C2731EE6D544F03180F78F509E63337
                                    CONTEXT service_id=236 ]
5698  5698 W ExposureNotification: Scan device AB:B1:E9:9E:1B:BA, type=1,
                                    id=EB7E87FA877BA96DC07554D5D5508074,
                                    raw_rssi=-12, calibrated_rssi=-16,
                                    meta=391ECC52, previous_scan=0
                                    [CONTEXT service_id=236 ]
5698  5698 W ExposureNotification: BleDatabaseWriter.writeBleSighting,
                                    id=EB7E87FA877BA96DC07554D5D5508074
                                    [CONTEXT service_id=236 ]
```

Exposure Status Also Logged



```
W ExposureNotification: [MatchingTracer] Sending exposure status update
                                         with no new exposures to client.
                                         [CONTEXT service_id=236 ]
```



Privacy Security Best Practices

This page contains a collection of data collection guidance and recommendations to ensure that Android users have control over the handling of their data.

Logging data

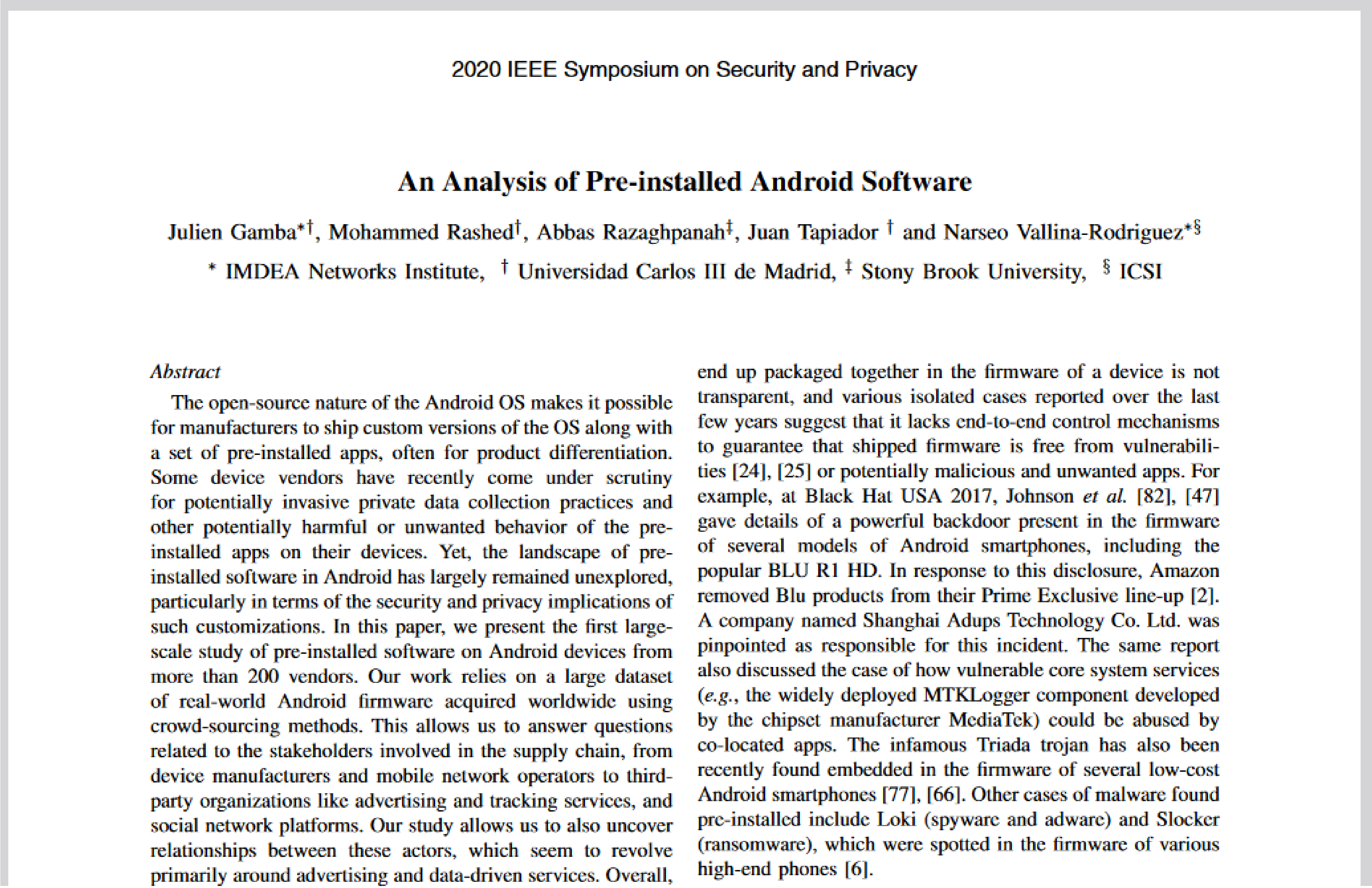
Logging data increases the risk of exposure of that data and reduces system performance. Multiple public security incidents have occurred as a result of logging sensitive user data.

- Do not log to the sdcard.
- Apps or system services should not log data provided from third-party apps that might include sensitive information.
- Apps must not log any Personally Identifiable Information (PII) as part of normal operation, unless it's absolutely necessary to provide the core functionality of the app.

CTS includes tests that check for the presence of potentially sensitive information in logs.

READ_LOGS Permission

“Not for use by **third-party** applications, because Log entries can contain the user's private information.”



2020 IEEE Symposium on Security and Privacy

An Analysis of Pre-installed Android Software

Julien Gamba*†, Mohammed Rashed†, Abbas Razaghpanah‡, Juan Tapiador † and Narseo Vallina-Rodriguez*§

* IMDEA Networks Institute, † Universidad Carlos III de Madrid, ‡ Stony Brook University, § ICSI

Abstract

The open-source nature of the Android OS makes it possible for manufacturers to ship custom versions of the OS along with a set of pre-installed apps, often for product differentiation. Some device vendors have recently come under scrutiny for potentially invasive private data collection practices and other potentially harmful or unwanted behavior of the pre-installed apps on their devices. Yet, the landscape of pre-installed software in Android has largely remained unexplored, particularly in terms of the security and privacy implications of such customizations. In this paper, we present the first large-scale study of pre-installed software on Android devices from more than 200 vendors. Our work relies on a large dataset of real-world Android firmware acquired worldwide using crowd-sourcing methods. This allows us to answer questions related to the stakeholders involved in the supply chain, from device manufacturers and mobile network operators to third-party organizations like advertising and tracking services, and social network platforms. Our study allows us to also uncover relationships between these actors, which seem to revolve primarily around advertising and data-driven services. Overall,

end up packaged together in the firmware of a device is not transparent, and various isolated cases reported over the last few years suggest that it lacks end-to-end control mechanisms to guarantee that shipped firmware is free from vulnerabilities [24], [25] or potentially malicious and unwanted apps. For example, at Black Hat USA 2017, Johnson *et al.* [82], [47] gave details of a powerful backdoor present in the firmware of several models of Android smartphones, including the popular BLU R1 HD. In response to this disclosure, Amazon removed Blu products from their Prime Exclusive line-up [2]. A company named Shanghai Adups Technology Co. Ltd. was pinpointed as responsible for this incident. The same report also discussed the case of how vulnerable core system services (*e.g.*, the widely deployed MTKLogger component developed by the chipset manufacturer MediaTek) could be abused by co-located apps. The infamous Triada trojan has also been recently found embedded in the firmware of several low-cost Android smartphones [77], [66]. Other cases of malware found pre-installed include Loki (spyware and adware) and Slocker (ransomware), which were spotted in the firmware of various high-end phones [6].

Logging on Android: Two Questions

1. Are developers following Google’s guidelines or does sensitive information end up in the logs?

2. Given the lack of supply chain controls, are there apps that can access the logs?

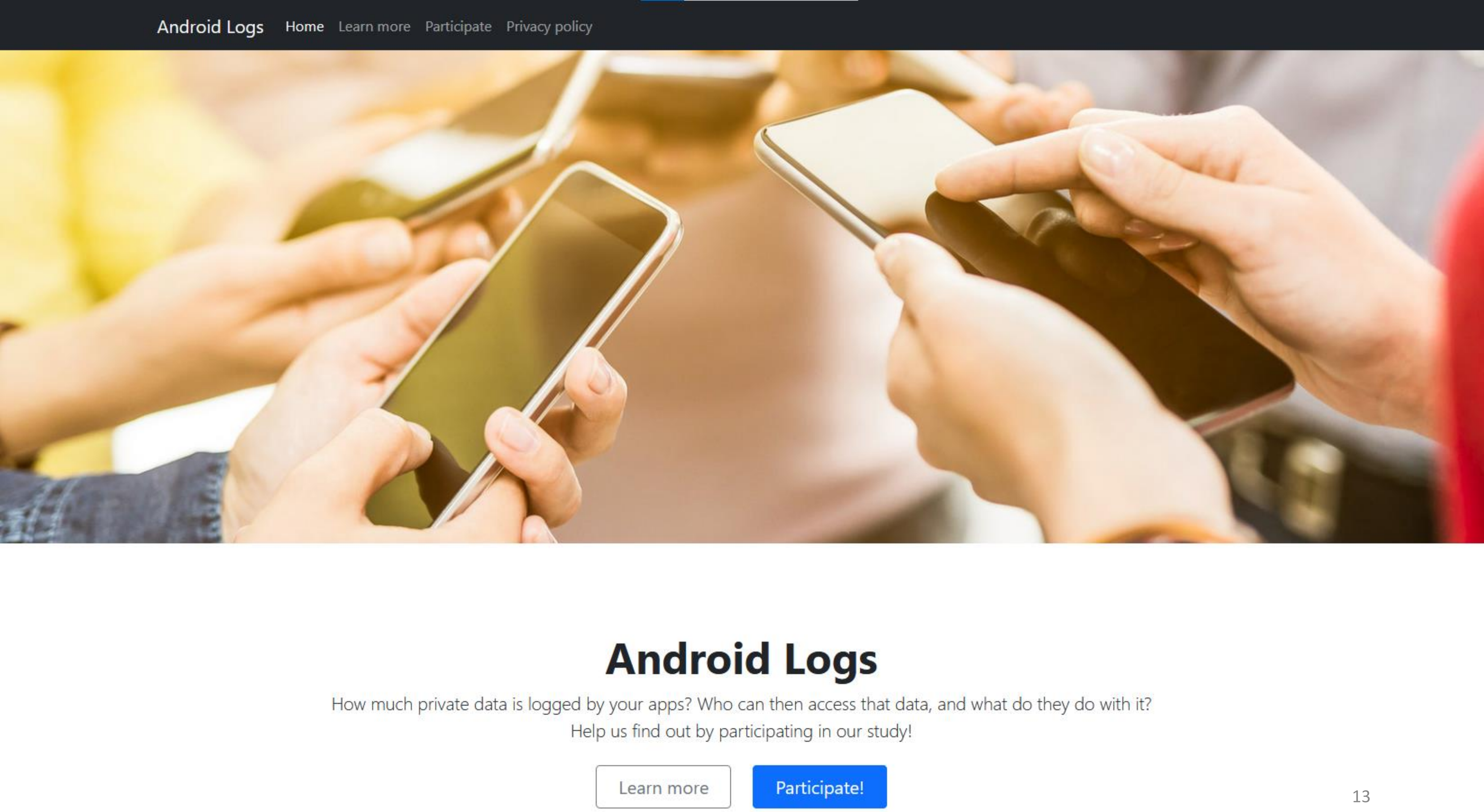
#1 : Is Sensitive Information in the Logs?

Personally Identifying Information (PII)

- Direct Identifiers
    - Email Address
    - Phone Number
    - User Name
- Indirect Identifiers
    - Android ID
    - MAC Address
    - IMEI
    - Serial Number
- User Location
    - GPS Coordinates
    - Nearby WiFi and Bluetooth devices

Logging by Default

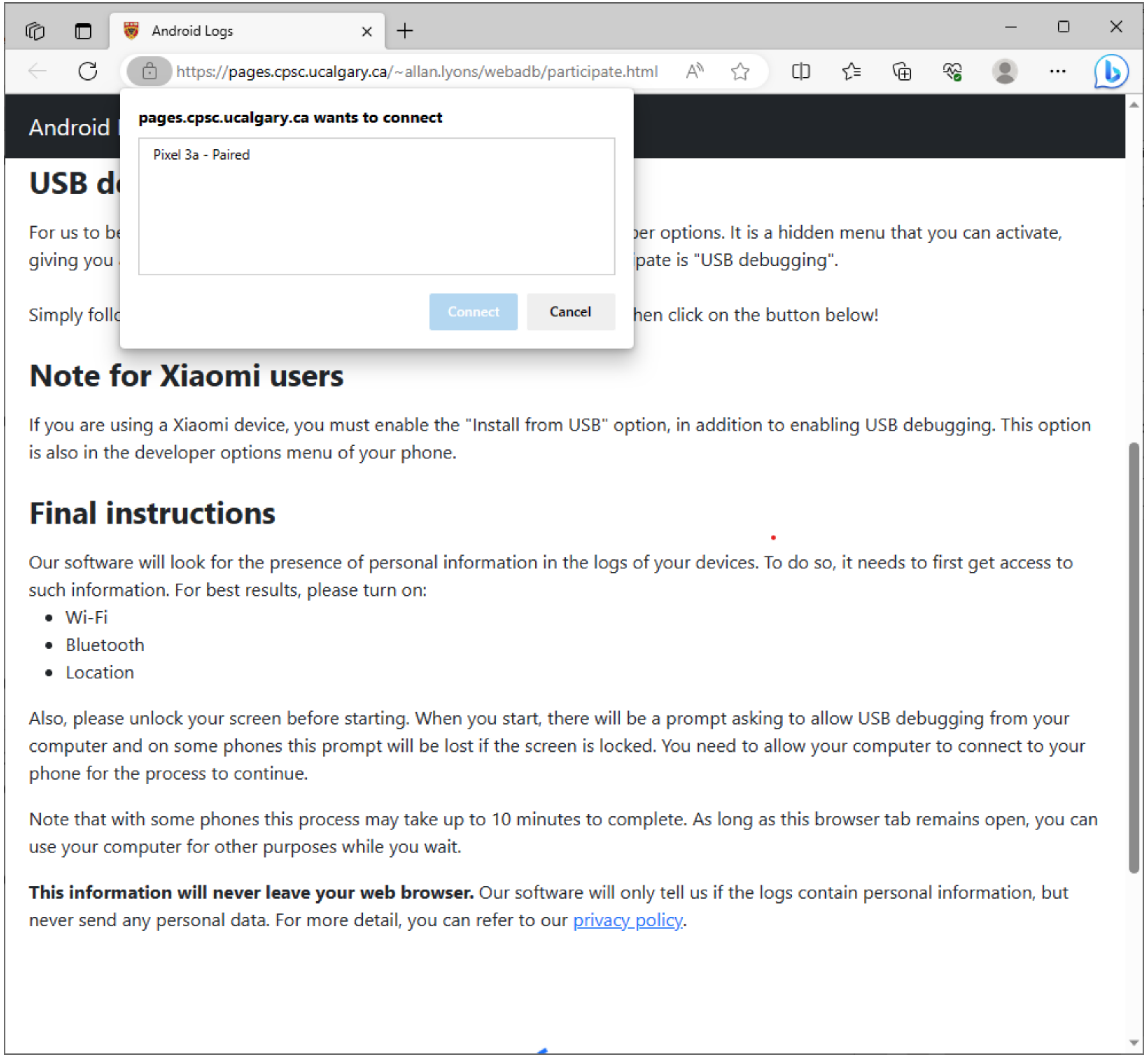
Phone			Identifier								Proximate Data					
Make	**Model**	**Android Version**	**SSID**	**BSSID**	**BT MAC**	**WiFi MAC**	**IMEI**	**Serial**	**Phone Num**	**Email Address**	**GPS**	**Nearby SSIDs**	**Nearby BSSIDs**	**Nearby BT MACs**	**Bluetooth Payloads**	**Read Logs**
Blu	Studio Mini	9	✓	✓	✓	✓				✓		✓		✓		5
Cubot	Note 7	10	✓	✓			✓		✓	✓		✓	✓			4
Google	Pixel 3a	9	✓	✓		✓	✓		✓	✓						6
Google	Pixel 3a	12	✓	✓					✓	✓		✓	✓			6
Huawei	Nova 5T	9	✓	✓	✓	✓				✓		✓	✓			58
LG	K51	12	✓	✓	✓	✓		✓		✓						58
Motorola	G Play	10	✓	✓	✓	✓	✓	✓		✓	✓					34
Motorola	One 5G Ace	10	✓	✓	✓	✓	✓	✓	✓	✓	✓					34
Nokia	3.4	10	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	5
Nokia	3.4	12	✓	✓	✓	✓		✓		✓	✓	✓	✓	✓	✓	22
Samsung	Galaxy A12	10	✓	✓	✓			✓		✓		✓	✓			14
Samsung	Galaxy A21S	10	✓	✓						✓						83
Samsung	Galaxy A21S	12	✓	✓						✓						95
uleFone	Note 11P	11	✓	✓	✓					✓						34
ZTE	Blade A5 2020	9	✓	✓		✓		✓		✓		✓	✓			4
Total			15	15	8	9	5	6	4	15	4	8	7	3	2	



Android Logs

How much private data is logged by your apps? Who can then access that data, and what do they do with it?
Help us find out by participating in our study!

PII in the Wild Experiment



Android Logs
https://pages.cpsc.ucalgary.ca/~allan.lyons/webadb/participate.html
Android

pages.cpsc.ucalgary.ca wants to connect

Pixel 3a - Paired

USB d

For us to be ... per options. It is a hidden menu that you can activate, giving you ... ipate is "USB debugging".

Simply follo ... hen click on the button below!

Note for Xiaomi users

If you are using a Xiaomi device, you must enable the "Install from USB" option, in addition to enabling USB debugging. This option is also in the developer options menu of your phone.

Final instructions

Our software will look for the presence of personal information in the logs of your devices. To do so, it needs to first get access to such information. For best results, please turn on:

- Wi-Fi
- Bluetooth
- Location

Also, please unlock your screen before starting. When you start, there will be a prompt asking to allow USB debugging from your computer and on some phones this prompt will be lost if the screen is locked. You need to allow your computer to connect to your phone for the process to continue.

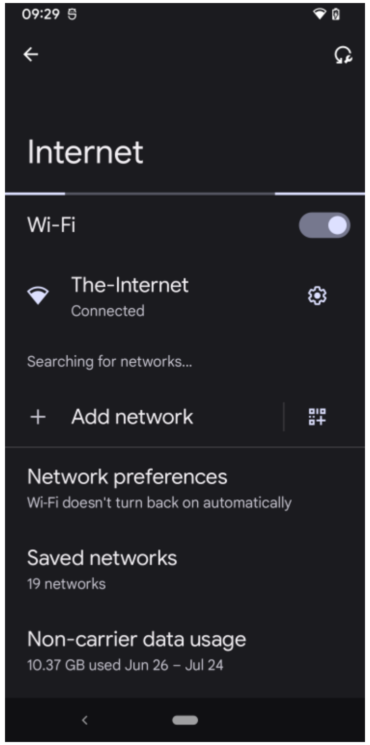
Note that with some phones this process may take up to 10 minutes to complete. As long as this browser tab remains open, you can use your computer for other purposes while you wait.

This information will never leave your web browser. Our software will only tell us if the logs contain personal information, but never send any personal data. For more detail, you can refer to our privacy policy.

PII Broadly Found on Phones

PII Type	PII Class	Prevalence
Email Address	Direct ID	16%
Phone Number	Direct ID	3%
Bluetooth Scan MAC	Location ID	2%
Bluetooth Scan SSID	Location ID	2%
Coarse Location	Location ID	24%
Fine Location	Location ID	22%
WiFi Router MAC	Location ID	67%
WiFi Router SSID	Location ID	68%
WiFi Scan MAC	Location ID	14%
WiFi Scan SSID	Location ID	39%
Android ID	Non-resetable ID	8%
Bluetooth MAC	Non-resetable ID	11%
IMEI	Non-resetable ID	6%
Serial Number	Non-resetable ID	4%
Bluetooth Name	Other ID	69%
WiFi Randomized MAC	Other ID	78%
Any PII Detected		**94%**

Google Pixel 3a Connecting to WiFi



09:29

Internet

Wi-Fi

The-Internet
Connected

Searching for networks...

Add network

Network preferences
Wi-Fi doesn't turn back on automatically

Saved networks
19 networks

Non-carrier data usage
10.37 GB used Jun 26 – Jul 24

Google Pixel 3a Connecting to WiFi

```
D wpa_supplicant: wlan0: Own MAC address: aa:52:0f:bc:55:60
D wpa_supplicant: wlan0: BSS: Add new id 2 BSSID
  a8:70:5d:84:2a:de SSID 'ShawMobileHotspot' freq 5765 HESSID
  a8:70:5d:84:2a:de
I wpa_supplicant: wlan0: RX-ANQP a8:70:5d:84:2a:de 3GPP
  Cellular Network information
D WifiClientModeImpl[wlan0]: ConnectedMacRandomization
  SSID(The-Internet). setMacAddress(aa:52:0f:bc:55:60) from
  02:c3:72:e5:15:17 = true
I WifiClientModeImpl[wlan0]: Connecting with aa:52:0f:bc:55:60
  as the mac address
I wpa_supplicant: wlan0: Trying to associate with SSID 'The-
  Internet'
I wpa_supplicant: wlan0: Associated with 08:9e:08:e4:2b:a0
```

wpa_supplicant Logging

wpa_supplicant.c

```
wpa_dbg(wpa_s, MSG_DEBUG, "Own MAC address: " MACSTR,
      MAC2STR(wpa_s->own_addr));
```

wpa_debug.h

```
#ifdef CONFIG_NO_STDOUT_DEBUG
#define wpa_dbg(args...) do { } while (0)
#else /* CONFIG_NO_STDOUT_DEBUG */
#define wpa_dbg(args...) wpa_msg(args)
#endif /* CONFIG_NO_STDOUT_DEBUG */
```

Xiaomi Redmi Note 9 (Android 11)

```
I [BIP]   :  [BIP NL] IPv6: *:*:*:*:*:*:*:*
I [BIP]   :  [BIP NL] addr state is 3, ipv4=*.*.*.*, ipv6=*:*:*:*:*:*:*:*
I WifiHAL : data: version=1, cur_rssi=-66 BSSID=12:0c:*:*:*:d9
```

Sample Volley Log Entry

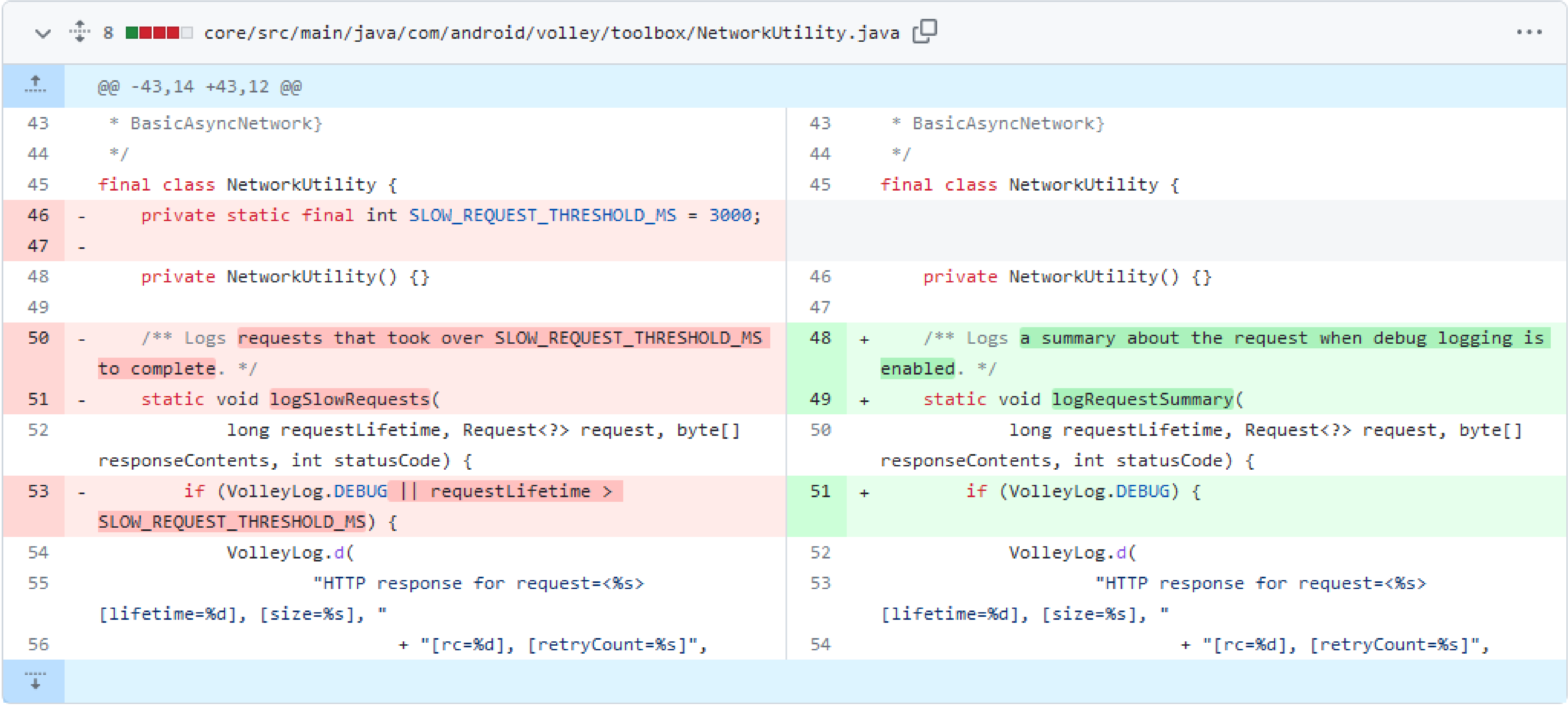
```
D Volley  : [919] BasicNetwork.logSlowRequests: HTTP
    response for request=<[ ] https://apis.netmarble.
    com/mobileauth/v2/players/063DFBE41A1342449E74C89BF
    2757786/deviceKeys/3CBCDA0D7F054EA5964CDAAD3353C651
    /accessToken?nmDeviceKey=d8b1df4dbf6926b2&country
    Code=CA&adId=7f9e4fac-c211-498e-804c-6befc76d39530
    xac67c518 IMMEDIATE 3> [lifetime=445], [size=851],
    [rc=200], [retryCount=0]
```

com.netmarble.war

Volley Logging Code

```
private static final int SLOW_REQUEST_THRESHOLD_MS = 3000;

private NetworkUtility() {}

/** Logs requests that took over SLOW_REQUEST_THRESHOLD_MS to complete. */
static void logSlowRequests(long requestLifetime, Request<?> request, byte[]
responseContents, int statusCode) {
        if (VolleyLog.DEBUG || requestLifetime > SLOW_REQUEST_THRESHOLD_MS) {
                VolleyLog.d("HTTP response for request=<%s> [lifetime=%d],
[size=%s], "
                + "[rc=%d], [retryCount=%s]", request, requestLifetime,
                responseContents != null ? responseContents.length : "null",
                statusCode, request.getRetryPolicy().getCurrentRetryCount());
        }
}
```

Volley Has Changed

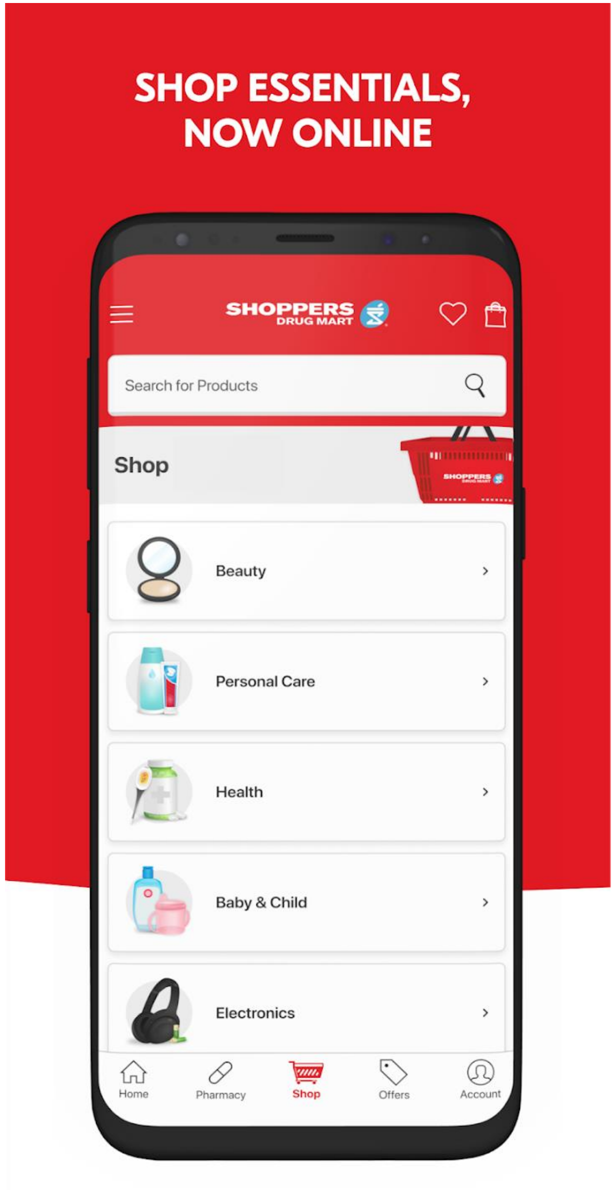


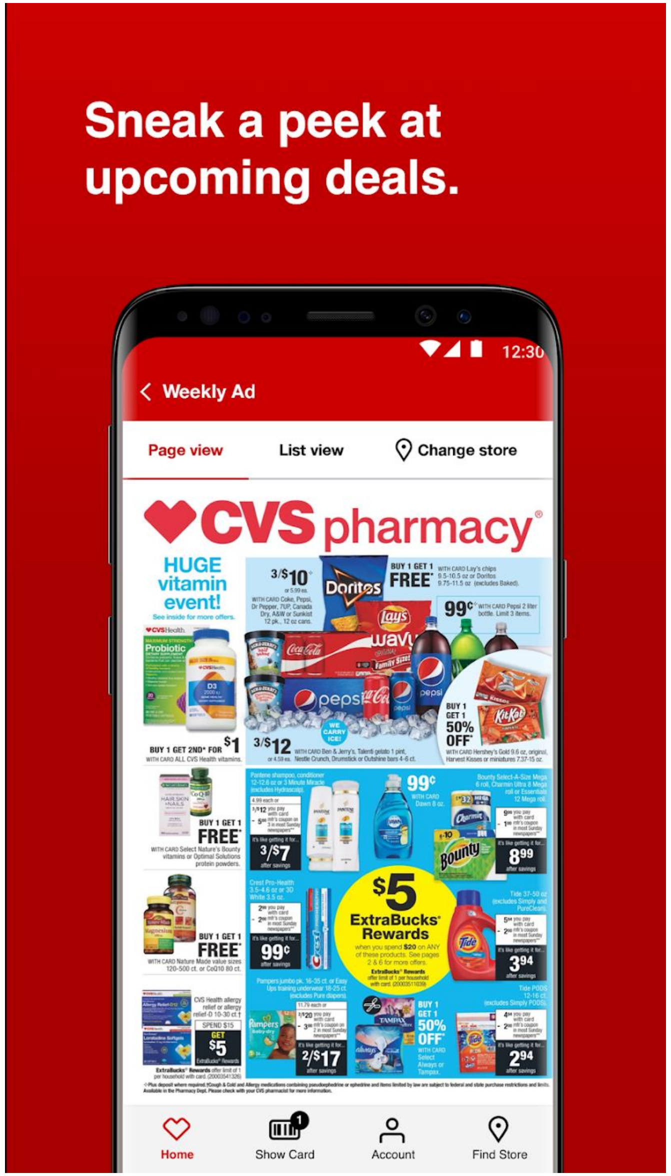
8 core/src/main/java/com/android/volley/toolbox/NetworkUtility.java

```
@@ -43,14 +43,12 @@
```

```diff
      * BasicAsyncNetwork}
      */
     final class NetworkUtility {
-        private static final int SLOW_REQUEST_THRESHOLD_MS = 3000;
-
         private NetworkUtility() {}

-        /** Logs requests that took over SLOW_REQUEST_THRESHOLD_MS to complete. */
-        static void logSlowRequests(
                 long requestLifetime, Request<?> request, byte[] responseContents, int statusCode) {
-            if (VolleyLog.DEBUG || requestLifetime > SLOW_REQUEST_THRESHOLD_MS) {
                 VolleyLog.d(
                         "HTTP response for request=<%s> [lifetime=%d], [size=%s], "
                                 + "[rc=%d], [retryCount=%s]",
```

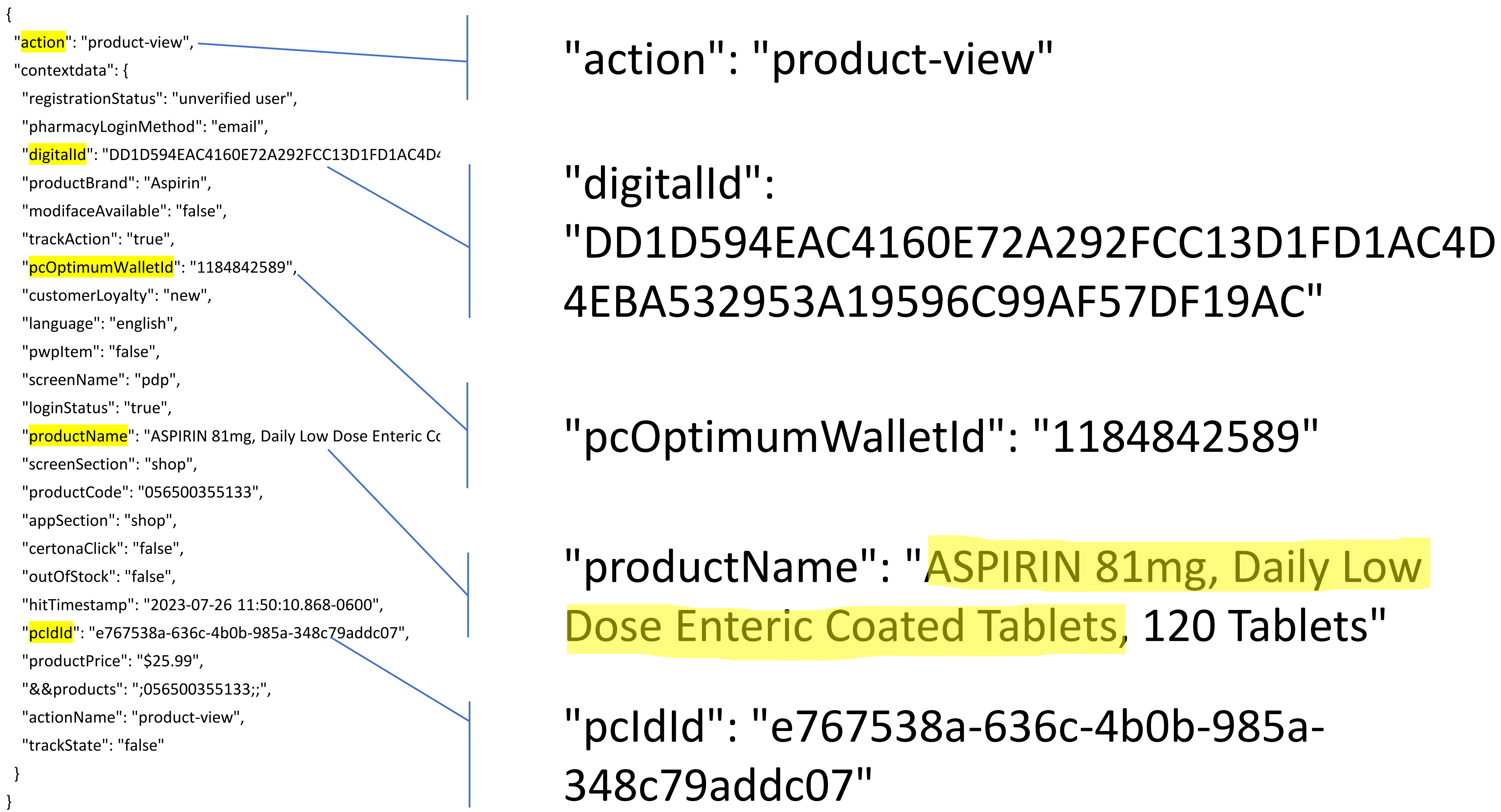
```diff
      * BasicAsyncNetwork}
      */
     final class NetworkUtility {
         private NetworkUtility() {}

+        /** Logs a summary about the request when debug logging is enabled. */
+        static void logRequestSummary(
                 long requestLifetime, Request<?> request, byte[] responseContents, int statusCode) {
+            if (VolleyLog.DEBUG) {
                 VolleyLog.d(
                         "HTTP response for request=<%s> [lifetime=%d], [size=%s], "
                                 + "[rc=%d], [retryCount=%s]",
```


SHOP ESSENTIALS, NOW ONLINE
SHOPPERS DRUG MART
Search for Products
Shop
Beauty
Personal Care
Health
Baby & Child
Electronics
Home
Pharmacy
Shop
Offers
Account


Sneak a peek at upcoming deals.
12:30
Weekly Ad
Page view
List view
Change store
CVS pharmacy
HUGE vitamin event!
$5 ExtraBucks Rewards
Home
Show Card
Account
Find Store

Sample Log Entry From Shopper’s App

```
D AdobeExperienceSDK: Rules Engine - Original EventData for Event #424:
{"action":"product-view","contextdata":{"registrationStatus":"unverified
user","pharmacyLoginMethod":"email","digitalId":"DD1D594EAC4160E72A2
92FCC13D1FD1AC4D4EBA532953A19596C99AF57DF19AC","productBrand":"
Aspirin","modifaceAvailable":"false","trackAction":"true","pcOptimumWallet
Id":"1184842589","customerLoyalty":"new","language":"english","pwpItem"
:"false","screenName":"pdp","loginStatus":"true","productName":"ASPIRIN
81mg, Daily Low Dose Enteric Coated Tablets, 180 Tablets", "screenSection":
"shop","productCode":"056500355133","appSection":"shop","certonaClick":
"false","outOfStock":"false","hitTimestamp":"2023-07-26 11:50:10.868-
0600","pcIdId":"e767538a-636c-4b0b-985a-348c79addc07",
"productPrice":"$25.99","&&products":";056500355133;;","actionName":"p
roduct-view","trackState":"false"}}
```



```
{
  "action": "product-view",
  "contextdata": {
   "registrationStatus": "unverified user",
   "pharmacyLoginMethod": "email",
   "digitalId": "DD1D594EAC4160E72A292FCC13D1FD1AC4D4
   "productBrand": "Aspirin",
   "modifaceAvailable": "false",
   "trackAction": "true",
   "pcOptimumWalletId": "1184842589",
   "customerLoyalty": "new",
   "language": "english",
   "pwpItem": "false",
   "screenName": "pdp",
   "loginStatus": "true",
   "productName": "ASPIRIN 81mg, Daily Low Dose Enteric C
   "screenSection": "shop",
   "productCode": "056500355133",
   "appSection": "shop",
   "certonaClick": "false",
   "outOfStock": "false",
   "hitTimestamp": "2023-07-26 11:50:10.868-0600",
   "pcIdId": "e767538a-636c-4b0b-985a-348c79addc07",
   "productPrice": "$25.99",
   "&&products": ";056500355133;;",
   "actionName": "product-view",
   "trackState": "false"
  }
}
```

"action": "product-view"

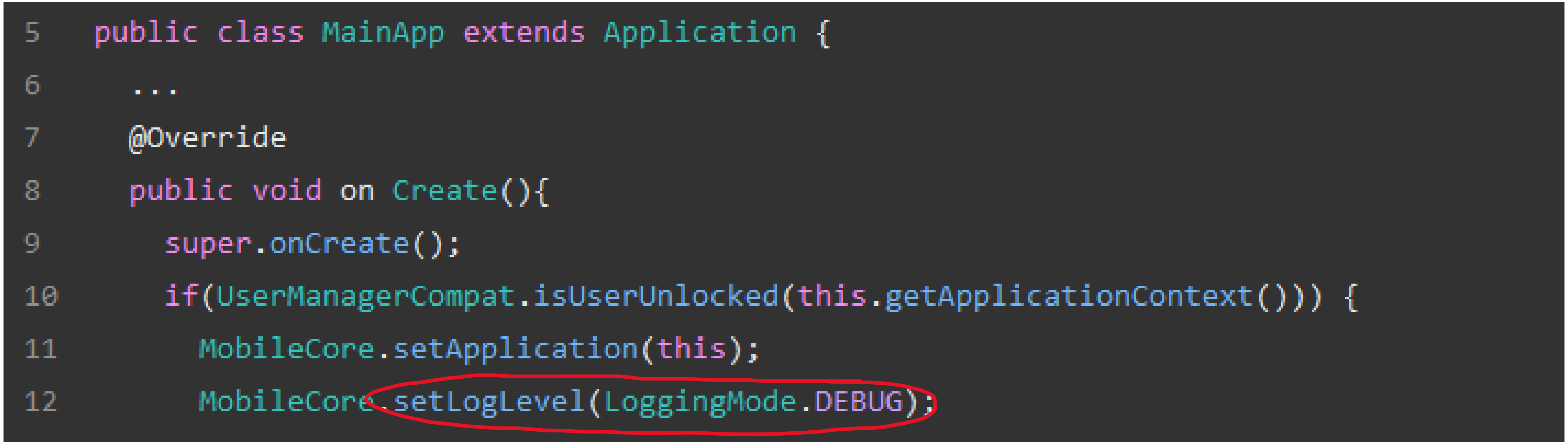
"digitalId": "DD1D594EAC4160E72A292FCC13D1FD1AC4D4EBA532953A19596C99AF57DF19AC"

"pcOptimumWalletId": "1184842589"

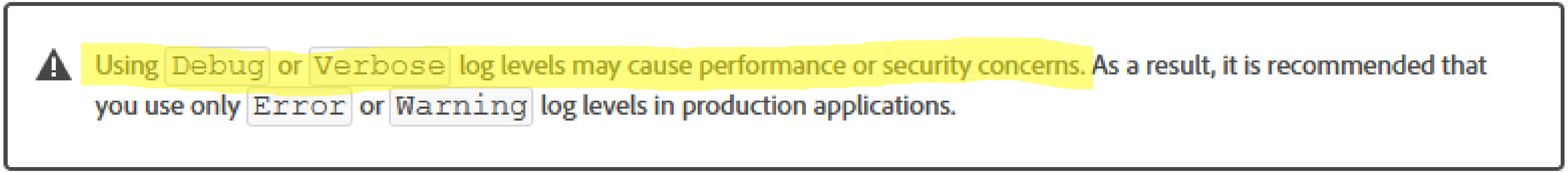
"productName": "ASPIRIN 81mg, Daily Low Dose Enteric Coated Tablets, 120 Tablets"

"pcIdId": "e767538a-636c-4b0b-985a-348c79addc07"

Adobe Experience Documentation

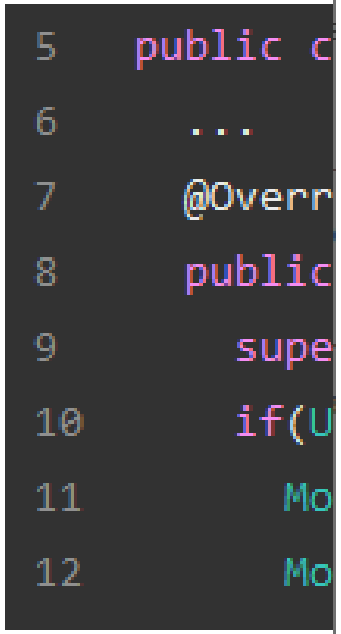


```
public class MainApp extends Application {
  ...
  @Override
  public void on Create(){
    super.onCreate();
    if(UserManagerCompat.isUserUnlocked(this.getApplicationContext())) {
      MobileCore.setApplication(this);
      MobileCore.setLogLevel(LoggingMode.DEBUG);
```



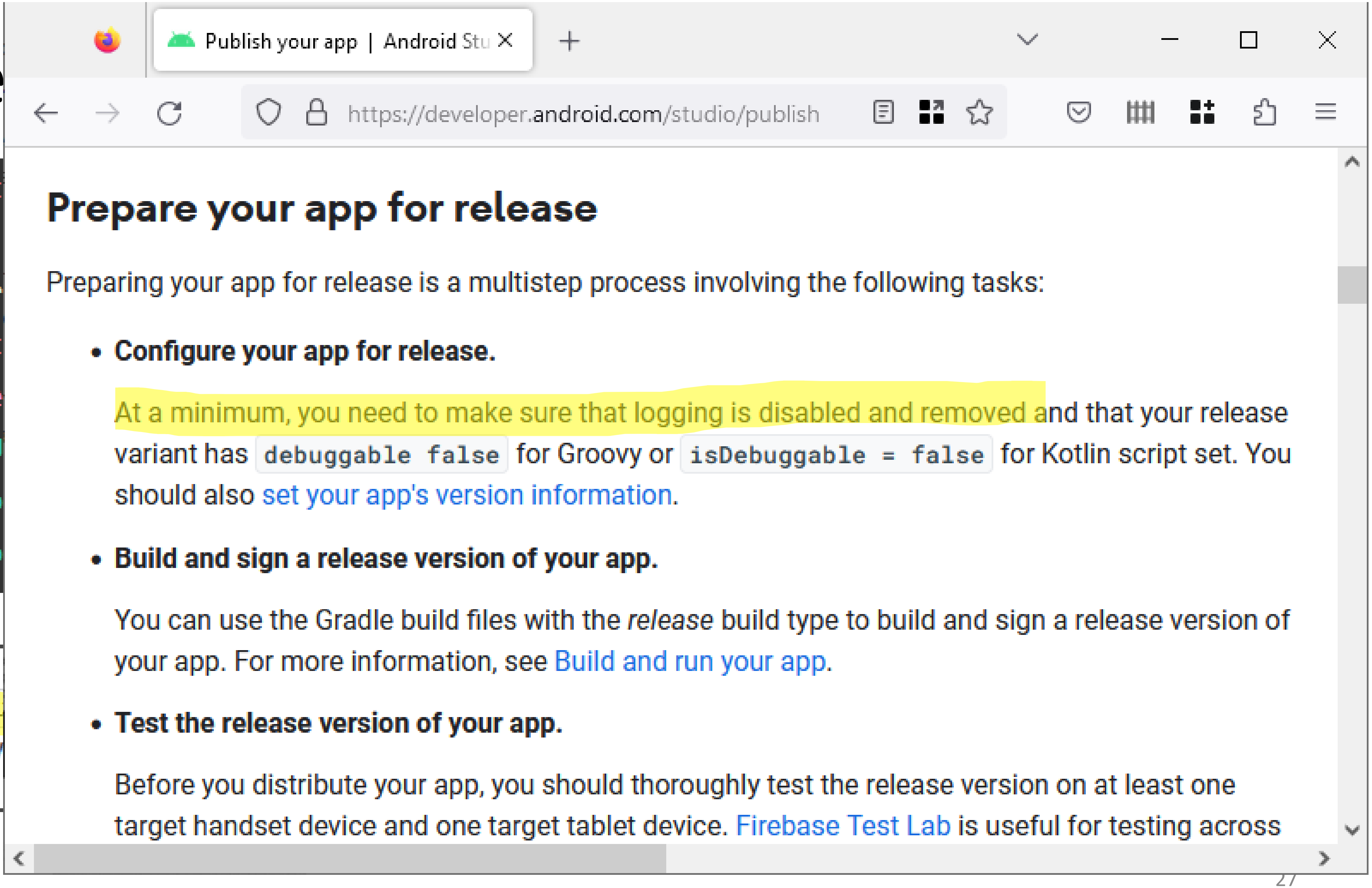
⚠ Using `Debug` or `Verbose` log levels may cause performance or security concerns. As a result, it is recommended that you use only `Error` or `Warning` log levels in production applications.

Adobe



```
public c
  ...
  @Overr
  public
    supe
    if(U
      Mo
      Mo
```

⚠ Using Debu
you use only



Publish your app | Android Stu
https://developer.android.com/studio/publish

Prepare your app for release

Preparing your app for release is a multistep process involving the following tasks:

- **Configure your app for release.**

  At a minimum, you need to make sure that logging is disabled and removed and that your release variant has `debuggable false` for Groovy or `isDebuggable = false` for Kotlin script set. You should also set your app's version information.

- **Build and sign a release version of your app.**

  You can use the Gradle build files with the *release* build type to build and sign a release version of your app. For more information, see Build and run your app.

- **Test the release version of your app.**

  Before you distribute your app, you should thoroughly test the release version on at least one target handset device and one target tablet device. Firebase Test Lab is useful for testing across

#2 : Are Apps Able to Access the Logs?

Do Apps Read Logs?

- Linking our field study to the dataset collected by Gamba et al.
    - 1,319 apps with READ_LOGS permission
    - 63 apps run logcat as a shell command
    - 15 of these have code to save the logs to the SD card
    - 9 apps post raw logs to the Internet

READ_LOGS Permission: What is a “**third-party**”?

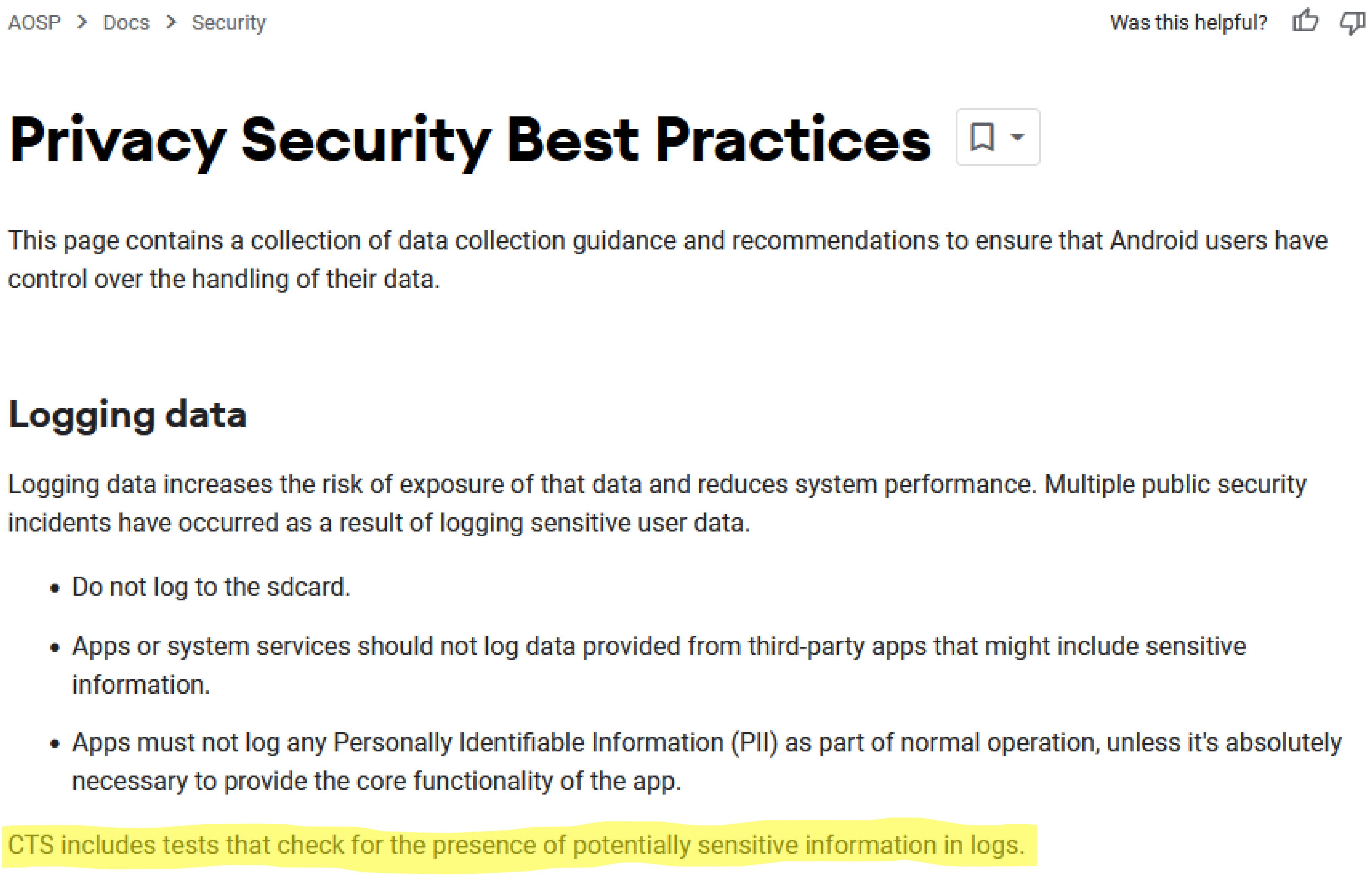
“Not for use by **third-party** applications, because Log entries can contain the user's private information.”

Apps with READ_LOGS. Third-party or Not?

- Apps by Mobile Network Operators like Verizon, AT&T, Telefonica
- Apps by large companies like Amazon, Baidu, Microsoft, Tencent
- Analytics services like Digital Turbine
- Utility apps such as “device cleaners”
- Parental control apps
- Anti-virus software

- Note that any SDKs used by the above inherit the permissions of the apps that include them

Mitigations



Privacy Security Best Practices

This page contains a collection of data collection guidance and recommendations to ensure that Android users have control over the handling of their data.

Logging data

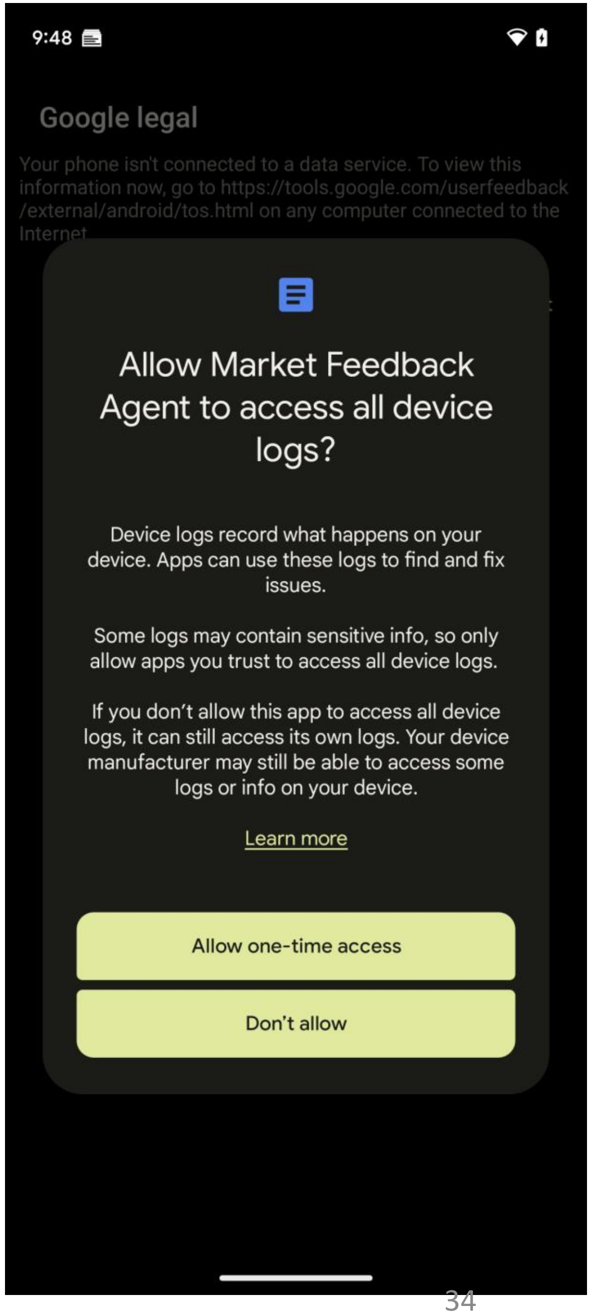
Logging data increases the risk of exposure of that data and reduces system performance. Multiple public security incidents have occurred as a result of logging sensitive user data.

- Do not log to the sdcard.
- Apps or system services should not log data provided from third-party apps that might include sensitive information.
- Apps must not log any Personally Identifiable Information (PII) as part of normal operation, unless it's absolutely necessary to provide the core functionality of the app.

CTS includes tests that check for the presence of potentially sensitive information in logs.

Recent Changes to Android

- “On Android 13, if an app tries to access all device logs for approved use cases such as app feedback or bug reporting, the system will ask you if you want to provide the app with one-time access to this more expansive set of logs.”
- Mitigation
  - If an app in the foreground with READ_LOGS requests access to the device logs, the system prompts the user to approve or deny the request.
  - An app running in the background is automatically denied unless the app
    - Shares the system UID
    - Uses a native system process (UID < APP_UID)
    - (and a few other cases listed in the documentation)



Google legal

Your phone isn't connected to a data service. To view this information now, go to https://tools.google.com/userfeedback/external/android/tos.html on any computer connected to the Internet.

Allow Market Feedback Agent to access all device logs?

Device logs record what happens on your device. Apps can use these logs to find and fix issues.

Some logs may contain sensitive info, so only allow apps you trust to access all device logs.

If you don’t allow this app to access all device logs, it can still access its own logs. Your device manufacturer may still be able to access some logs or info on your device.

Conclusion

- Logging of potentially sensitive information is prevalent despite Google's recommendations to protect end users
  - System services log sensitive information
  - Misconfigured libraries and SDKs
  - "Debugging during deployment"

- Many preinstalled apps can read the logs

- Impact: Our work has led to a change in Android log access notification

Log: It's Big, It's Heavy, It's Filled with Personal Data!
Measuring the Logging of Sensitive Information in the Android Ecosystem

Allan Lyons
University of Calgary

Julien Gamba
IMDEA Networks Institute / Universidad Carlos III de Madrid

Austin Shawaga
University of Calgary

Joel Reardon
University of Calgary / AppCensus, Inc.

Juan Tapiador
Universidad Carlos III de Madrid

Serge Egelman
ICSI / UC Berkeley / AppCensus, Inc.

Narseo Vallina-Rodríguez
IMDEA Networks Institute / AppCensus, Inc.

Abstract

Android offers a shared system that multiplexes all logged data from all system components, including both the operating system and the console output of apps that run on it. A security mechanism ensures that user-space apps can only read the log entries that they create, though many "privileged" apps are exempt from this restriction. This includes preloaded system apps provided by Google, the phone manufacturer, the cellular carrier, as well as those sharing the same signature. Consequently, Google advises developers to not log sensitive information to the system log.

In this work, we examined the logging of sensitive data in the Android ecosystem. Using a field study, we show that most devices log some amount of user-identifying information. We show that the logging of "activity" names can inadvertently reveal information about users through their app usage. We also tested whether different smartphones log personal identifiers by default, examined preinstalled apps that access the system logs, and analyzed the privacy policies of manufacturers that report collecting system logs.

1 Introduction

Printing diagnostics and informative output messages is fundamental to software development [45]. Meaningful, verbose logging allows one to monitor runtime behaviour and helps to quickly find and diagnose bugs without needing to replicate the issue in a debugger. Continual logging of software in production facilitates monitoring and remediation of runtime crashes. It is therefore typical that software systems log their behaviour in some manner after deployment.

At the same time, logs often contain sensitive and personal information, particularly when logs are unified across a diverse collection of software components. To that point, in 2010 Rosenberg introduced a Linux kernel patch to provide a build option to make access to the kernel ring buffer, i.e., the output of `dmesg`, effectively require root access (`CAP_SYS_ADMIN`); this option was subsequently modified to require `CAP_SYSLOG` and is now widely enabled. In deploying this patch, Rosenberg noted both the futility of attempting to sanitize thousands of `printk` statements along with the resulting loss of functional utility of these sanitized logs [53].

Analogous to Linux, the Android platform also offers a shared system log that collects all logging statements from apps, system services, and drivers [9]. Developers can write to this log using a family of logging functions, differentiated by verbosity, that provide both a log tag and message. A single system log multiplexes the output of all the log statements that occur along with metadata, which includes thread and process identifiers, a log level ("verbosity"), log tag, and timestamp.

On Android, access to the logs has never been part of the published Software Development Kit (SDK) [40], but early versions allowed full access based on requesting a permission (i.e., READ_LOGS). Without it, apps can only access their own log messages (but not logs from other apps or operating system components); with the permission, apps have access to *all* log data. Android 4.1 changed the READ_LOGS permission to become "privileged," meaning that the permission itself is only available to system apps and apps that come preinstalled (e.g., by manufacturers, carriers, etc.) [14, 40, 41].

Many such privileged apps exist. Android is an open platform that allows any manufacturer to create its own custom version of the Operating System (OS) with preinstalled software that they determine. This means that apps from the phone's manufacturer (OEM) and other key actors in the supply chain, such as Mobile Network Operators (MNOs), OEM partners, and Google, are eligible for privileged access to the system log. Gamba et al. examined preinstalled software in the Android ecosystem [35] and found a vast supply chain with many privileged third parties able to access this data. They noted that many of these apps also included additional SDKs, provided by ad networks, analytics services, or social networks, that inherit the permissions of the embedding app [34] (i.e., these third-party components can read the logs).

The Android Feedback app (`com.google.android.feedback`) is an example of a privileged, preinstalled app. It can launch after another app crashes and gives the user the

1