149 New Montgomery Street
4th Floor
San Francisco, CA 94105

COVID-19 Android Apps: Spain

App Analysis Report

April 18, 2020

www.appcensus.io

Table of Contents

Acknowledgments

This report was done in collaboration with Prof. Juan Tapiador, Universidad Carlos III de Madrid. AppCensus, Inc. would like to acknowledge his valuable contributions.

Disclaimer

This report was produced independently by AppCensus, without the knowledge, sponsorship, endorsement or participation of any software provider discussed herein. At this time, AppCensus has not reported the results of its analysis, or sought input or comment, from any software provider discussed herein.

The results in this report are limited to the particular software versions and experimental conditions described. Please be aware that results could differ in other environments. Analysis of mobile phone apps is complex, and although we believe we have made reasonable efforts to provide an accurate report, we expressly disclaim all warranties, including without limitation the warranty of merchantability and fitness for a particular purpose, and we do not warrant that the contents of this report, or the findings or conclusions herein, are complete or error-free.

Introduction

On April 3rd 2019, the World Health Organization (WHO) published a press release announcing the results of a coordinated effort led by the WHO Digital Health Technical Advisory Group to develop technical solutions that could trace the development of the COVID-19 infection, perform population screening, and more efficiently allocate limited medical resources.[1]

In the early days of the COVID-19 pandemic outbreak, countries like Singapore, South Korea, Taiwan, Italy and Spain released—either at a regional or national level—Android apps to assist their citizens by providing them information, or allowing them to self-diagnose their symptoms so that they can alleviate the load on emergency telephone numbers. This, however, has raised numerous concerns from privacy advocates about the potential privacy implications of this type of software, if not implemented correctly.[2]

In this report, we present the results of an analysis of the following COVID-19 Android apps released by regional and national public health services in Spain (as of April 18, 2020):

1. CoronaMadrid (versions 1.0.7 and 1.0.8, the latter released on the 2nd of April, 2020), published by the regional government of Madrid (Comunidad de Madrid).
2. COVID-19.eus (versions 1.0 and 1.2, the latter released on the 4th of April, 2020), published by the public health service of Euskadi (Osakidetza).
3. STOP COVID19 CAT (version 1.0.1, released on the 20th of March, 2020), published by the Regional Government of Catalunya (Generalitat de Catalunya).
4. Asistencia COVID-19 (version 1.0.0, released on the 6th of April, 2020), published by the Ministry of Economy and Digital Transformation (MINECO).

These apps have been designed by public authorities with three clear objectives in mind: (1) to assist users in identifying COVID-19-related symptoms and performing a self-diagnosis in order to minimize the load on emergency services; (2) to provide actionable information and recommendations to the public; and (3) to collect information about the spread of the virus, including potential COVID-19-positive subjects.

We examined how these apps made use of permission requests, which regulate access to sensitive data (e.g., personal information), and to whom they transmitted the resulting data. We specifically examined what permissions were requested by the apps and whether those permissions were actually used at runtime. We also examined the third-party

[1] https://www.who.int/news-room/detail/03-04-2020-digital-technology-for-covid-19-response

[2] https://www.nature.com/articles/d41586-020-00740-y

SDKs that were bundled with the apps, which could be used for exfiltrating personal information. Finally, we examined the apps' network traffic to detect the potential exfiltration of personal information (e.g., known persistent identifiers, contact information, location data, etc.).

We performed our analysis on mobile phones physically located in Spain, in a heavily instrumented analysis environment. Each app was installed on a test phone and executed by a human operator while the environment recorded whenever the app tried to perform an action that required a "dangerous" permission. The permissions that an app may use are known *a priori*; during our testing, we determined which of these were actually used during app execution.

Additionally, we monitored all network traffic going in and out of the device, including traffic encrypted using TLS and "pinned" certificates. We searched this traffic for persistent identifiers, location information, and other sensitive data that may be exfiltrated. In order to better understand and contextualize our empirical observations, we analyzed the code of each app.

While none of the tested apps implements contact-tracing technologies, like the South Korean and Taiwanese apps, we have noticed that STOP COVID19 CAT seems to integrate geo-fencing capabilities that could have serious implications for users' privacy rights.

In the next section, we describe the different types of personal information that we examined in more detail. We then describe our methodology, as well as how the analysis component of this report is structured to understand the results.

We caution that just because we observed that a permission was not used, or that a transmission of data did not occur, does not mean that it would not occur under different circumstances. Certain functionality may not have been triggered during the testing period. Moreover, there are means of evading our monitoring system that may have been put in place by app developers. In particular, any apps for which we observed no transmissions of sensitive data may need to be further scrutinized to determine if such techniques are in place (by inspecting the raw data associated with this report).

Summary

In this report, we analyzed the privacy behaviors and risks of four Android apps released by the Spanish national government and three regional health agencies in response to the COVID-19 pandemic. We observed the following:

1. As it is common in most mobile apps, the four governmental apps depend on cloud services like Google Cloud (CoronaMadrid and Asistencia COVID-19), Mubiquo's push notifications and geo-fencing services (STOP COVID19 CAT), and Amazon Web Services (STOP COVID19 CAT), possibly to speed-up the development process and scale-up their backend infrastructure. As a result, sensitive data from citizens, including national ID numbers, geolocation data, phone numbers, chronic medical conditions, and COVID-19 symptoms are hosted on services offered by non-European companies, thus also subject to foreign jurisdictions.

2. The lack of meaningful information provided in X.509 certificates on the instances deployed in cloud providers makes difficult attributing their deployment to specific stakeholders (government agencies, or to third parties like partners, or providers). As a result, it is difficult to identify the actual organizations operating these services and responsible for collecting sensitive data and their role as a first-party or as a third-party service from a technical standpoint (STOP COVID19 CAT, COVID-19.eus).

3. The app publishers are not sufficiently transparent about the presence of third-party SDKs like Crashlytics, Google Firebase, or Google's Mobile Services (GMS).

4. Both CoronaMadrid and Asistencia COVID-19 seem to upload data to the same database hosted by Google Cloud. We cannot tell whether they act as individual or joint data controllers from a technical standpoint. According to their respective privacy policies, each relevant competent public authorities is a sole data controller, rather than joint data controllers.

5. The four apps implement authentication mechanisms based on SMS text messages (two-factor authentication) and national ID numbers (or social security numbers). However, these technologies can be easily circumvented with fake information (*e.g.,* ID number generators, and disposable SMS services). This lack of control over the authenticity of the users can have a negative effect on the quality of the data to control the pandemic.

Methodology

Our approach to analyzing mobile apps relies on a combination of "static" and "dynamic" analysis. Static analysis refers to analyzing programs without running them, in order to detect the presence of specific instructions that *may* be executed when the program is run. For example, static analysis can be used to answer the question, "does the program include a particular function?" Static analysis is quick, because it does not involve interactively running the program. However, it is prone to false positives, as not all program code is reached during execution (i.e., "dead code"); code detected through static analysis may never get executed in practice (e.g., due to libraries that provide multiple functions, but only a subset of which are used by the given app).

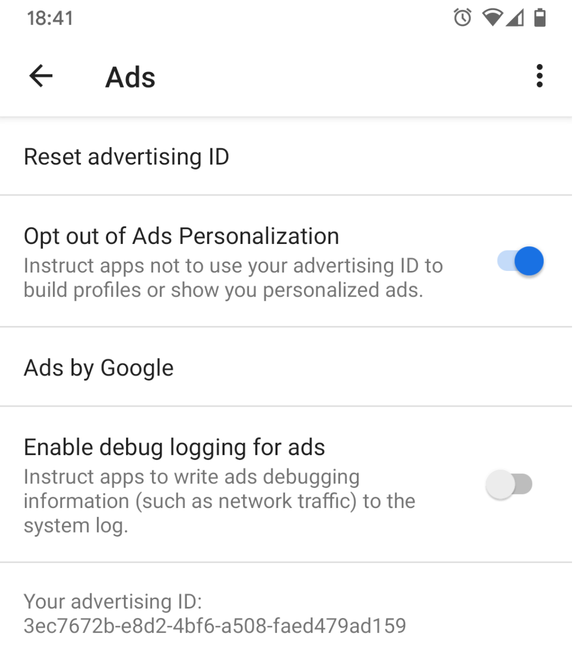
Dynamic analysis, on the other hand, refers to running programs to directly observe their behaviors. For example, dynamic analysis can answer the question, "what functions does the program actually use?" Dynamic analysis more realistically models program behavior, as the program is executed in a testing environment that is designed to model real-world usage. However, it is prone to false negatives: program code not executed during the testing period may be executed under different conditions and stimuli. Dynamic analysis is also desirable because it does not yield false positives: conclusions are observations of *actual* program behavior. Thus, by performing both static and dynamic analysis, we can determine reasonable upper and lower bounds for the types of privacy-invasive behaviors that a user is likely to encounter under normal usage.

We performed dynamic analysis on the apps to examine whether personal information would be transmitted over the Internet during the course of realistic app usage. During testing, human testers installed apps on smartphones and then interacted with them to simulate real-world usage. We examined the network traffic generated by the apps in order to detect the exfiltration of personal information (discussed in detail in the next section of this report). Additionally, we ran the apps with our modified version of the operating system, which included additional instrumentation to monitor how apps attempt to access sensitive data stored on the device, including usage of the Android permissions system. Finally, we performed static analysis of the apps to identify bundled third-party software development kits (SDKs) and the use of various privacy-related functions and settings.

The Android test devices were all Google Nexus 5X smartphones running a modified version of AOSP 7.1 (The Android Open Source Project, or AOSP, is an open source branch of the Android operating system). While our testing capabilities are also ported to Android 9, most carrier-locked devices only support system updates for a year or two after release, and therefore most Android users have older versions, unless they opt to buy new hardware. For this reason, we perform our testing on Android versions that most consumers are actually using, rather than the most recent version. The testing devices were physically located in Madrid, Spain, and accessed the Internet through a major residential ISP.

We modified the operating system by instrumenting the permission-checking APIs. What this means is that using our modified version of the operating system, whenever an unmodified Android app attempts to access a resource protected by Android's permission system, our instrumentation logs this information, so that we can understand which apps have accessed protected user data. This allows us to monitor the execution of individual apps without having to modify them. In testing Android apps, we also use bespoke instrumentation to monitor the payloads of network traffic, allowing us to examine even TLS-encrypted traffic (regardless of whether certificate "pinning" is used).

Our testing procedure for Android is to first install an app by downloading its newest version from the Google Play Store. Once the app is installed, it is launched. No other Play Store apps were installed on the device to prevent our observations from being confounded. The testing device was logged into a valid Google account that was associated with that device (as is the norm for Android). We manually interacted with each app while collecting data on the app's behaviors while it was being used extensively for several hours. We also used real phone numbers to register on services that required them. Finally, the logs generated during testing were downloaded from each phone for analysis.



18:41

Ads

Reset advertising ID

Opt out of Ads Personalization ●
Instruct apps not to use your advertising ID to build profiles or show you personalized ads.

Ads by Google

Enable debug logging for ads ○
Instruct apps to write ads debugging information (such as network traffic) to the system log.

Your advertising ID:
3ec7672b-e8d2-4bf6-a508-faed479ad159

Figure 1: System-wide ad tracking/personalization settings in Android.

Android offers users the option to opt out of tracking using a system-wide setting (Figure 1). When accessing the AAID, apps are supposed to follow a policy that requires them to check the status of this setting and then elect to handle the AAID accordingly.[3] In practice,

[3] https://play.google.com/about/monetization-ads/ads

this often means transmitting an additional parameter to indicate to data recipients that the user has opted out of tracking. During testing, the tracking opt-out was disabled (i.e., the default setting) and all permissions were granted in the first run. Then, we tested app behavior by modifying the permission controls in order to identify potential differences in the data collection, and whether the app honors users' privacy choices. Nevertheless, we did not observe the exfiltration of the AAID parameter in any of the apps. This could be caused by the fact that none of the tested apps leverage advertising-based monetization models.

We use MaxMind's GeoIP2 Precision Enterprise API to geo-locate the approximate physical locations of the servers contacted by each mobile app. The accuracy of our IP geolocation data is, as a result, limited to MaxMind's accuracy.[4]

[4] https://dl.acm.org/doi/abs/10.1145/1971162.1971171

Personal Information

Throughout the results section of this report, we refer to transmissions of personal information with reference to a data type. We also look for other data introduced directly by the user through the User Interface (UI) of each app, including information such as previous medical conditions, symptoms related to COVID-19, the user's physical address, or date of birth.

In this section, we define each of these data types, describe what they are, what they look like, what purpose they serve, and the degree to which they can be easily changed by the user (e.g., to preserve privacy). For simplicity, we categorize these data types amongst three main categories: persistent identifiers that allow a user to be tracked over time and across services, sensitive personal information that reveals a user's contact information or personal details, and fine-grained location data that allows a user to be physically located with a high degree of precision.

Persistent Identifiers

Persistent identifiers are unique numbers that allow an individual to be tracked over time and across services. Some of them can be reset to prevent this type of tracking (e.g., the "AAID"), whereas others are hardware-based and cannot be easily reset by the user. In other cases, developers create custom IDs by applying hash functions to one or more unique identifiers. For instance, the MD5 hash[5] of the unique IMEI—see definition below—`353626076259204` is `895a9bddd9949ec194c6ba97f6823950`. However, a hash value of a unique identifier is still a unique identifier, and therefore does not protect privacy, as it still identifies a user uniquely and persistently. As a poor mechanism for protecting privacy, it is also vulnerable to de-anonymization attacks, such as through the use of lookup tables.[6]

AAID: This is the Android advertising identifier (provided in our raw data as "`aaid`"). It has the format of a Java-style UUID, such as `4e398b96-c1b7-4937-942c-1ab8d4e34b3a`. It is a resettable identifier, meaning that users can go through the system settings to reset it to a new value (this is the mobile tracking equivalent of clearing web browser cookies to prevent tracking across websites). Google's developer guidelines require that this is the only identifier that is used for behavioral advertising and other tracking/profiling purposes (see documentation).

Android ID: The Android ID (provided in our raw data as "`androidid`") is a semi-persistent identifier, meaning that it can be reset, but only through an extraordinary operation: a factory reset of the mobile phone. As such, the Android ID remains the same even when the AAID changes. Apps that send both together are therefore able to bridge changes to the AAID (i.e., resetting the AAID has no privacy-preserving effect). It has the format of a 16-digit hexadecimal number, such as `a553362398083b36`.

[5] https://en.wikipedia.org/wiki/MD5

[6] http://blog.dasient.com/2011/07/hashing-imei-numbers-does-not-protect.html

GSF ID: This is the Google Services Framework Identifier (provided in our raw data as "gsfid"). This is an identifier that ties the user's account to Google Services. It has the format of a 16-digit hexadecimal number, such as `37ee4f071b362be6`.

IMEI: This is the International Mobile Equipment Identity (provided in our raw data as "`imei`"). This is used when connecting to cellular networks and to blacklist stolen phones. It is a unique identifier that cannot be changed. It is illegal to change the IMEI in some jurisdictions, including the United Kingdom. It does not change with factory resets and therefore remains the same even for refurbished mobile phones. It has the format of a 15-digit decimal number, such as `353626076259204`.

Serial #: This is the serial number of the phone (provided in our raw data as "`hwid`"). It is semi-persistent in that it requires a factory reset and a custom operating system to change. It has the format of a 16-digit hexadecimal number, such as `02597e74305f4802`.

SIM ID: the identifier tied to the SIM card that is installed in the phone (provided in our raw data as "`simid`"). As with the telephone number, the SIM ID can only change if the user removes the SIM card and installs a new one.

WiFi MAC: the MAC address of the phone's networking hardware (provided in our raw data as "`wifimac`"). The MAC address is typically persistent, as it does not change with factory resets or other operations. It can be temporarily changed with non-trivial technical effort, but can only be persistently changed by rooting the phone. It has the format of 6 octets separated with a colon, such as `a8:b8:6e:4b:d0:a2`.

Sensitive Personal Information

We define sensitive personal information to include personal contact information, demographic data, and other information that speaks directly to an individual's preferences.

Name: The full name associated with the Google Account that was provided when configuring the mobile phone (provided in our raw data as "`name`"). This corresponds to the owner of the phone.

Email Address: The email address that was used when first configuring the mobile phone (provided in our raw data as "`email`"). This corresponds to the owner of the phone's email address.

Phone #: This is the phone number associated with the SIM card that is installed on the phone (provided in our raw data as "`phone`"). The phone number can change, but it requires a new SIM card in order to do so.

Package Dump: This refers to an app recording and transmitting the names of all of the other installed apps ("packages") on the device (provided in our raw data as "`package_dump`"). This data is generally used for behavioral profiling purposes, as it can both be used to fingerprint a device and infer the user's interests based on the other apps that they use.

Username: This corresponds to the username of the user's app profile, which may be used to identify the user's profile.

User ID: This corresponds to a numerical identifier that identifies the user's app profile.

Government ID: This corresponds to an alphanumerical identifier issued by the government. This information, if leaked by a mobile app, would most likely have been introduced by the user through the user interface of the app.

Medical information: In the context of health apps, users might introduce medical records (*e.g.,* previous chronic conditions) and symptoms (*e.g.,* fever, weakness) that could be disseminated to various data recipients.

Location Information

Location information consists of either fine-grained GPS coordinates or other information that would allow the recipient to infer the user's location with a high degree of accuracy (e.g., SSID or BSSID). This does not include coarse-grained location data (e.g., city, state, or country) or data that would allow a recipient to infer the user's location on a coarse-grained level (e.g., IP address).

GPS Location: This means transmission of both the latitude and longitude (provided in our raw data as "`geolatlon`") in the same network transmission with more than 3 decimal places of accuracy, which is roughly within 100m. This can be considered a precise location, as it provides street-level accuracy.

SSID: This is the name of the WiFi router that the phone is either connected to or can perceive in the vicinity (provided in our raw data as "`routerssid`"). The SSID is the human-friendly WiFi name, which is not guaranteed to be globally unique, although many routers have unique default SSIDs. The SSID does not uniquely identify a user, as it may change throughout the day and the router may have more than one user; it does, however, give coarse-grained location data of a comparable accuracy to GPS.

BSSID: This is the MAC address of the WiFi router and is a globally unique identifier (provided in our raw data as "`routermac`"). The BSSID does not uniquely identify a user, as it may change throughout the day and the router may have more than one user; it does, however, give coarse-grained location data of a comparable accuracy to GPS.

How to Read This Report

The analysis section follows, which first presents an overview of our most significant findings, across all apps tested. The remainder of the analysis section is split into subsections for each app tested. For each of the apps tested, we present the following data:

- User Authentication
- Android Permissions
- Third-Party SDKs
- Traffic Analysis and Backend Infrastructure
- Data Exfiltration

In the User Authentication subsection, we analyze the mechanisms implemented by each app to authenticate users, including the presence of two-factor authentication mechanisms. We also analyze the mechanisms implemented to identify and deny access to minors.

The Android Permissions subsection includes a table containing the list of permissions classified as "dangerous" by Google (see Google's permission overview), such as those protecting access to sensitive resources. For each dangerous permission that was requested by the app, we indicate whether we observed the app actually make use of the permission at runtime. Due to our methodology, not all code branches for each app will execute during our testing, and so there may be functionality that makes use of a permission that we simply did not examine during our testing.

If an app contains any identifiable third-party SDKs, a table is present in the Third-Party SDKs subsection that contains the list of these SDKs that we identified as being present in that app. We also provide a short description of those SDKs that could be used for long-term tracking, profiling, and behavioral advertising, if present. These are the SDKs that we were able to positively identify; it is possible that in some cases additional obfuscated or otherwise unidentified SDKs may exist.

The Traffic Analysis and Backend Infrastructure section contains a table describing the different hostnames that the app connected to during our manual analysis. We report on their X.509 certificates—a proxy to attribute these services to their operators—and hosting infrastructure. We also report on the use of TLS/SSL encryption per hostname (i.e., whether personal data was properly encrypted while in transit).

The table in the Data Exfiltration subsection contains all observed transmissions of the personal information described in the Personal Information section of this report, if any were detected during testing. This table, if present, has three columns for each transmission: the hostname or IP address that received the transmission, the type of data that was transmitted, and whether TLS (encryption) was used to secure the transmission. The Data Exfiltration subsection also provides some high-level observations about the observed behaviors.

Analysis

5.1 Overview

For the purposes of this report, we tested the following app(s):

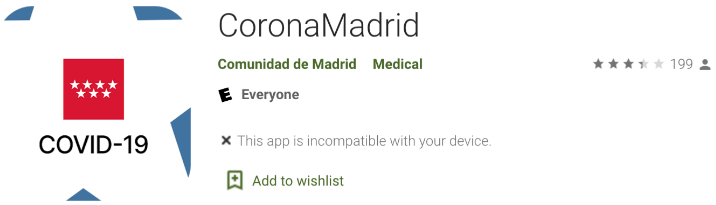
App Name	Package Name	Version	Date Published[7]
CoronaMadrid	org.madrid.CoronaMadrid	1.0.7	28 March 2020
CoronaMadrid	org.madrid.CoronaMadrid	1.0.8	2 April 2020
COVID-19.eus	com.erictelm2m.colabora	1.0	27 March 2020
COVID-19.eus	com.erictelm2m.colabora	1.2	3 April 2020
STOP COVID19 CAT	cat.gencat.mobi.StopCovid19Cat	1.0.1	20 March 2020
Asistencia COVID-19	es.gob.asistenciacovid19	1.0.0	6 April 2020

The app versions tested were the latest releases at the time of testing. Yet, as of today, new versions have been released for the following apps:

- STOP COVID19 CAT, version 1.0.2 (2nd April 2020). According to the developer's "What's new" report on their Google Play profile, this new version contains minor improvements.
- Asistencia COVID-19, version 1.0.1 (8th of April 2020). According to the developer's "What's new" report on their Google Play profile, this new version now offers visual hints to inform users about the last time they used the app, plus several bug fixes and performance improvements.

[7] Release dates for older versions are approximated based on available information.

5.2 CoronaMadrid



CoronaMadrid

Comunidad de Madrid Medical

199

Everyone

This app is incompatible with your device.

Package Name	org.madrid.CoronaMadrid
Versions	1.0.7 and 1.0.8
MD5	748c1cc91677c17175b3a37906e9417b (1.0.7) 187dc228652073cce103e322c77834bc (1.0.8)
Developer Information	ForceManager (`support@forcemanager.net`)
App Signature	Owner: O=Comunidad de Madrid, L=Madrid, ST=Madrid, C=ES; Issuer: O=Comunidad de Madrid, L=Madrid, ST=Madrid, C=ES; Serial number: 5331a586
Certificate Fingerprint (MD5)	03:D3:F3:22:04:63:BF:01:66:30:45:58:70:62:90:2C
Installs	50,000+
Privacy Policy	`https://www.coronamadrid.com/proteccion-de-datos`
Data Controller	Viceconsejería de Asistencia Sanitaria de la Consejería de Sanidad de la Comunidad de Madrid ("CSCM").

This mobile app was launched by the regional government of Madrid, first as a web-based solution, and then as a mobile app on the 24th of March. According to an FAQ available on their website,[8] the app was made possible thanks to the altruistic contributions of different individuals and companies, including Carto, ForceManager, and Mendesaltaren, with the support and collaboration of Google, Telefonica, Ferrovial, and Gogoo network. From a technical perspective, the app was developed using React Native, which eases the development of cross-platform applications by leveraging JavaScript.

While the app was initially developed for the regional government of Madrid, it seems to have been designed for release at a national level. The analysis of the traffic generated by

[8] `https://www.coronamadrid.com/preguntas-frecuentes`

this app (Example 1) indicates that it could be easily extended to provide recommendations and guidance for users from other regions. The tested app versions do not implement contact-tracing and geofencing.

This app is released by a public health service and is free of charge. As a result, it is likely expected that it does not engage with any advertising or tracking services. Based on our analysis, this app appears to be meeting this expectation. **We observed no significant differences in terms of personal data collection between versions 1.0.7 and 1.0.8.**

Example 1: Sample of the downlink traffic intercepted by AppCensus between the service `coronamadrid.comunidad.madrid` (https://coronamadrid.comunidad.madrid/protocols_metadata.json) and the CoronaMadrid app over HTTPS. It suggests that the app could offer recommendations and guidelines for users from other regions.

```
    "default_protocol": "comun-nacional-v1",
    "regions": [
        {
            "id": "01",
            "name": "Andalucia",
            "custom_protocol": ""
        },
        {
            "id": "02",
            "name": "Aragon",
            "custom_protocol": ""
        },
        {
            "id": "05",
            "name": "Canarias",
            "custom_protocol": ""
        },
        {
            "id": "06",
            "name": "Cantabria",
            "custom_protocol": ""
        },
        {
            "id": "08",
            "name": "Castilla␣-␣La␣Mancha",
            "custom_protocol": ""
        },
        {
            "id": "07",
            "name": "Castilla␣y␣Leon",
            "custom_protocol": ""
        },
]
```

Potential Security Issue:

While the Google Play profile contact email belongs to ForceManager, the data controller is the Comunidad de Madrid, according to the privacy policy, and the app is signed with a certificate issued by the Comunidad de Madrid. This raises a number of security concerns. Particularly, in the case that the third-party company (ForceManager) has developed and signed the software using the root certificate of the public health agency. In the unfortunate scenario that the root certificate is not well managed and protected by the third-party responsible for the software development, a malicious actor could sign and publish software impersonating the public service.

5.2.1 User Authentication

The app requires users to validate a phone number (done through Firebase two-factor authentication services), as well as provide a valid ID number (either DNI or NIE) to register for the service.

While the privacy policy and terms of service of the app state that the user must provide real data when registering, it is trivial to register for the service using fake information provided by disposable SMS services[9] and ID generators. Nevertheless, we cannot discount the possibility of the service providers performing offline verification and data sanitization on the server-side. In fact, the privacy policy states that the service provider can disable the account if they identify a violation of their terms of service. If not, the lack of verification mechanisms may allow malicious actors to intentionally pollute the data collection or otherwise degrade the utility of the service.

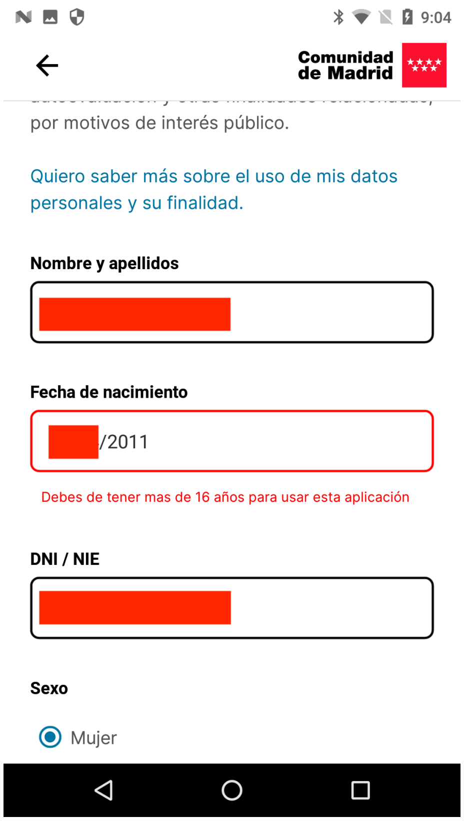
Age control: CoronaMadrid's privacy policy states that the user must be over 16 years of age or have the consent of a parent or legal guardian. We ran a separate test, pretending to be a 9 year old boy to study the behavior of the app in this scenario. When introducing the date of birth, the app denied access as shown in Figure 2. However, if the user introduces a valid date of birth, the app will operate as normal without performing any robust age verification.

5.2.2 Android Permissions

The table below depicts the normal and "dangerous" permissions—those requiring user approval—that the Android app had requested, as well as whether the app was observed making use of those permissions (e.g., to access sensitive user data) during dynamic analysis. In blue, we highlight "custom permissions" associated with third-party services.[10]

[9] E.g., `https://smsreceivefree.com/`

[10] Android allows app developers to define their own permissions. By defining custom permissions, an app can share its resources and capabilities with other apps.



por motivos de interés público.

Quiero saber más sobre el uso de mis datos personales y su finalidad.

Nombre y apellidos

[redacted]

Fecha de nacimiento

[redacted]/2011

Debes de tener mas de 16 años para usar esta aplicación

DNI / NIE

[redacted]

Sexo

● Mujer

Figure 2: UI notification informing a 9 year-old minor about the age limit for using the service.

Permission	Protection Level
`android.permission.INTERNET`	Normal
`android.permission.ACCESS_NETWORK_STATE`	Normal
`android.permission.ACCESS_WIFI_STATE`	Normal
`android.permission.ACCESS_COARSE_LOCATION`	Dangerous
`android.permission.ACCESS_FINE_LOCATION`[11]	Dangerous
`com.google.android.c2dm.permission.RECEIVE`	Custom permission: Push notifications
`com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE`	Custom permission: Firebase-related

[11] As of Android 10, gaining access to the BSSID (*i.e.,* MAC Address of the WiFi access point) also requires access to the `ACCESS_FINE_LOCATION` permission. This change in Android's permission model was motivated by our prior published research demonstrating that several apps used this information to infer users' geolocation without requesting the location permission. (See `https://www.usenix.org/system/files/sec19-reardon.pdf`.) Users running lower Android versions are still vulnerable to this side-channel attack.

According to the Firebase documentation, the "custom permissions" requested by its SDK are required to report events to the server, such as installation events or scheduling tasks, possibly for analytics purposes. (Firebase is owned by Google.) We also observed the custom permission `com.google.android.c2dm.permission.RECEIVE` which is required by Google's push notification service (*i.e.*, server-to-client notifications and communications).

5.2.3 Third-Party SDKs

The app's code contains various third-party libraries. Many of them are open-source libraries offering development support, including those maintained and developed by large companies like Facebook and Google. For example, there are several Facebook libraries present for easing the development and deployment of cross-platform mobile apps, such as the React Native SDK,[12] Facebook's Yoga[13] and Facebook's Hermes.[14]

The third-party libraries identified in the source code are listed below:

SDK	Provider	Package	Type
Android Support v4	Google	android/support/v4	Development Support
Google Core Libraries for Java 6+	Google	com/google/common	Development support
Google Core Libraries	Google	com/google/thirdparty	Development support
Google Gson	Google	com/google/gson	Development support
OkHttp	OkHttp	com/squareup/okhttp	Development support
Hermes	Facebook	com/facebook	Development support
Yoga	Facebook	com/facebook	Development support
Fabric[15]	Google	io/fabric/sdk/android	Development support
Bolts Base Library	Parse/Facebook	bolts	Development support
Firebase	Google	com/google/firebase	Development support, cloud integration, analytics, A/B testing, push notifications, Crashlytics, and authentication.
Google Mobile Service	Google	com/google/android/gms	Development support, advertising services, mapping, push notifications, and analytics.

[12] `https://reactnative.dev/`
[13] `https://yogalayout.com/`
[14] `https://github.com/facebook/hermes`
[15] Deprecated: now known as Firebase.

The third-party libraries highlighted in red are known to collect personal information from app users. This is the case for Google's Firebase suite[16] (which also includes Google's Crashlytics bug reporting and analytics solution)[17] and Google's Mobile Services.[18] The latter is a library offering APIs to integrate various Google services, including Google Maps, in mobile apps.

5.2.4 Traffic Analysis and Backend Infrastructure

During the manual execution of CoronaMadrid using the AppCensus platform, we identified connections to different cloud services. As can be seen, a number of thems are associated with the third-party services collecting personal and behavioral data that were previously identified by our static analysis: Crashlytics and Google's Geocoding API.[19]

Domain	IP Address	Server X.509 Certificate	Hosting / Description
settings.crashlytics.com	172.217.17.3[20]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = google.com	Google Crashlytics
reports.crashlytics.com	54.243.191.123[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = crashlytics.com	Google Crashlytics
europe-west1-covid19madrid.cloudfunctions.net	216.239.36.54[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = *.google.com	Service hosted in Google Cloud Services
maps.google.com	172.217.17.14[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = *.google.com	Google Maps / Geocoding API
firestore.googleapis.com	172.217.168.170[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = *.googleapis.com	Google Cloud Services
play.google.com	216.58.211.46[42]	Owner: CN=*.google.com, O=Google LLC, L=Mountain View, ST=California, C=US Issuer: CN=GTS CA 1O1, O=Google Trust Services, C=US	Google

[16] https://firebase.google.com/

[17] https://firebase.google.com/docs/crashlytics

[18] https://www.android.com/gms/

[19] https://developers.google.com/maps/documentation/geocoding/start

[20] Approximate location based on MaxMind's GeoIP2 service: U.S.

coronamadrid.comunidad.madrid	13.224.106.129[42]	Organization: AGENCIA PARA LA ADMINISTRACION DIGITAL DE LA COMUNIDAD DE MADRID Org. Unit: SUB-DIRECCIÓN GENERAL DE SERVICIOS Y GESTIÓN DE APLICACIONES Location: MADRID, ES	Amazon Cloudfront CDN

This analysis reveals that the online infrastructure supporting this app is deployed on Google Cloud Services, including its database. Therefore, while these services are owned by Google, they seem to act as a first-party service deployed by the developer on Google's infrastructure. The only exception is the domain `coronamadrid.comunidad.madrid`, which is hosted by Amazon's CloudFront CDN. Manual inspection of the traffic payload shows that this domain sends instructions and medical recommendations for the users after completing a self-assesment test for COVID-19. We have not observed this hostname performing any personal data collection.

The app also uses Google Maps' Geocoding API for what seems to be improving and normalizing user-introduced geolocation information, as shown in the flow captured below (Example 2). Part of the response obtained from Google Maps' API is later forwarded to the database hosted by `firestore.googleapis.com`, excluding parameters such as the place ID—a unique identifier for a physical location generated by Google Maps—and location type. For instance, the address "Puerta Sol" will be corrected as "Puerta del Sol".

Example 2: Google Maps Geocoding API response to CoronaMadrid. Corrected location information and metadata—marked in red—are anonymized by us for this report.

```
"results" : [
   {
      "address_components" : [
         {
            "long_name" : "195",
            "short_name" : "195",
            "types" : [ "street_number" ]
         },
         {
            "long_name" : "STREET_NAME_ANONYMIZED",
            "short_name" : "STREET_NAME_ANONYMIZED",
            "types" : [ "route" ]
         },
         {
            "long_name" : "Madrid",
            "short_name" : "Madrid",
            "types" : [ "locality", "political" ]
         },
         {
            "long_name" : "Madrid",
            "short_name" : "M",
            "types" : [ "administrative_area_level_2", "political" ]
         },
         {
            "long_name" : "Comunidad␣de␣Madrid",
               "short_name" : "Comunidad␣de␣Madrid",
               "types" : [ "administrative_area_level_1", "political" ]
            },
            {
               "long_name" : "Spain",
               "short_name" : "ES",
               "types" : [ "country", "political" ]
            },
            {
               "long_name" : "28026",
               "short_name" : "28026",
               "types" : [ "postal_code" ]
            }
         ],
         "formatted_address" : "Calle␣de␣STREET_NAME_ANONYMIZED, 195, 28026 Madrid, Spain
         "geometry" : {
            "location" : {
               "lat" : LATITUDE_ANONYMIZED,
               "lng" : LONGITUDE_ANONYMIZED
            },
            "location_type" : "ROOFTOP",
            "viewport" : {
               "northeast" : {
                  "lat" : LATITUDE_ANONYMIZED,
                  "lng" : LONGITUDE_ANONYMIZED
               },
               "southwest" : {
                  "lat" : LATITUDE_ANONYMIZED,
                  "lng" : LONGITUDE_ANONYMIZED
               }
            }
         },
         "place_id" : "PLACE_ID_ANONYMIZED",
         "plus_code" : {
            "compound_code" : "CODE_ANONYMIZED Madrid, Spain",
            "global_code" : "GLOBAL_CODE_ANONYMIZED+MP"
         },
         "types" : [ "street_address" ]
      }
   ],
   "status" : "OK"
```

TLS Encryption: We observed that during testing, the app always used HTTPS for connecting with online services, regardless of the domain name contacted.

Unused Code: We note that using static analysis methods to extract URL strings from the code might be possible to find many domains associated with other third-party services, as listed in the table below. As we can see, the app contains URLs associated with Google Fitness services. Yet, we did not observe any communication with them across multiple app executions. This demonstrates that the presence of an SDK in an app's code does not guarantee that it is ultimately invoked by the actual app at runtime.

Example 3: Google-related URLs (i.e., API endpoints) found in the app during static analysis. However, our AppCensus instrumentation did not detect any data flows to these services.

```
APP_STATE =
        "https://www.googleapis.com/auth/appstate";
CLOUD_SAVE =
        "https://www.googleapis.com/auth/datastoremobile";
DRIVE_APPFOLDER =
        "https://www.googleapis.com/auth/drive.appdata";
DRIVE_APPS =
        "https://www.googleapis.com/auth/drive.apps";
DRIVE_FILE =
        "https://www.googleapis.com/auth/drive.file";
DRIVE_FULL =
        "https://www.googleapis.com/auth/drive";
FITNESS_ACTIVITY_READ =
        "https://www.googleapis.com/auth/fitness.activity.read";
FITNESS_ACTIVITY_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.activity.write";
FITNESS_BLOOD_GLUCOSE_READ =
        "https://www.googleapis.com/auth/fitness.blood_glucose.read";
FITNESS_BLOOD_GLUCOSE_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.blood_glucose.write";
FITNESS_BLOOD_PRESSURE_READ =
        "https://www.googleapis.com/auth/fitness.blood_pressure.read";
FITNESS_BLOOD_PRESSURE_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.blood_pressure.write";
FITNESS_BODY_READ =
        "https://www.googleapis.com/auth/fitness.body.read";
FITNESS_BODY_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.body.write";
FITNESS_BODY_TEMPERATURE_READ =
        "https://www.googleapis.com/auth/fitness.body_temperature.read";
FITNESS_BODY_TEMPERATURE_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.body_temperature.write";
FITNESS_LOCATION_READ =
        "https://www.googleapis.com/auth/fitness.location.read";
FITNESS_LOCATION_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.location.write";
FITNESS_NUTRITION_READ =
        "https://www.googleapis.com/auth/fitness.nutrition.read";
FITNESS_NUTRITION_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.nutrition.write";
FITNESS_OXYGEN_SATURATION_READ =
        "https://www.googleapis.com/auth/fitness.oxygen_saturation.read";
FITNESS_OXYGEN_SATURATION_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.oxygen_saturation.write";
FITNESS_REPRODUCTIVE_HEALTH_READ =
        "https://www.googleapis.com/auth/fitness.reproductive_health.read";
FITNESS_REPRODUCTIVE_HEALTH_READ_WRITE =
        "https://www.googleapis.com/auth/fitness.reproductive_health.write";
GAMES =
        "https://www.googleapis.com/auth/games";
GAMES_LITE =
        "https://www.googleapis.com/auth/games_lite";
PLUS_LOGIN =
        "https://www.googleapis.com/auth/plus.login";
PLUS_ME =
        "https://www.googleapis.com/auth/plus.me";
```

5.2.5 Data Exfiltration

When the app was granted all the requested permissions, we observed the following types of personal data transmitted to the following hostnames:

Domain	Personal Information
settings.crashlytics.com reports.crashlytics.com	• Installation ID (X-CRASHLYTICS-INSTALLATION-ID) • City
europe-west1-covid19madrid.cloudfunctions.net	• National ID number
maps.google.com	• User-introduced location (country, region, postal code, street name, street number)
firestore.googleapis.com	• Full name • User-introduced location (Country, region, postal code, street name, street number) • Corrected user-introduced location (as provided by Google Maps Geocoding) • National ID number / INE • Phone number • Device information • Gender • Device token • GPS location (latitude / longitude) • Local IP address • App-generated UID • COVID-19 symptoms • Self-assessment • Last report

The privacy policy did not inform users about the presence of Crashlytics at the time of running the tests. We note that none of the different custom UIDs generated by different app components seem to be shared with other domains and services.

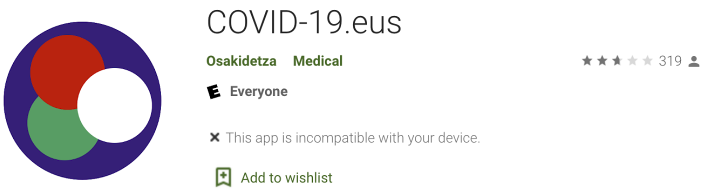
Minors: We note that the phone number and the user ID generated by Firebase during the two-factor authentication attempt (delegated to Google's servers at `securetoken.google.com`) are transmitted before completing the age verification mechanism, as shown in the following listing:

Example 4: Phone number and synthetic user ID being uploaded to firestore.googleapis.com as a result of the 2-factor authentication mechanism

```
{"iss":"https://securetoken.google.com/covid19madrid","aud":"covid19madrid","
auth_time":1586307646, "user_id":"krIuIVzFEthA07zzewqLoquDnfk1",
"sub":"krIuIVzFEthA07zzewqLoquDnfk1","iat":1586307646,"exp":1586311246,
"phone_number":"ANONYMIZED","firebase":{"identities":{"phone":["+ANONYMIZED"]},
"sign_in_provider":"phone"}}
```

Denying geolocation permissions: We did not observe geolocation data being transmitted if the user denies access to geolocation permissions.

5.3 COVID-19.eus



COVID-19.eus

Osakidetza Medical

319

Everyone

This app is incompatible with your device.

Add to wishlist

Package Name	com.erictelm2m.colabora
Version	1.0, and 1.2 (4th of April)
MD5	cda75d9052e44b674ceb86c9bedd325e (1.0) 502b0c5745acdde05ac13e72f8b6085c (1.2)
Developer Information	Osakidetza (Basque public health system)
App Signature	Owner: CN=Android, OU=Android, O=Google Inc., L=Mountain View, ST=California, C=US Issuer: CN=Android, OU=Android, O=Google Inc., L=Mountain View, ST=California, C=US Serial number: 77f2b36d7f123db053d3cd4e50e0f2a9c1bcdd7e
Certificate Fingerprint (MD5)	48:3D:73:92:65:53:85:5A:6D:9F:40:94:CF:99:42:85
Installs	10,000+
Privacy Policy	`https://colaboro.erictel.com/privacy/`
Data Controller	DIRECCION DE SALUD PÚBLICA Y ADICCIONES DEL DEPARTAMENTO DE SALUD

This app was released by the Basque Health System (Osakidetza) in collaboration with the company EricTel.[21] The app is free of charge and, as a public service, it is likely expected that it does not engage with any advertising or tracking services. Furthermore, it does not seem to implement any geo-fencing and contact-tracing capabilities. According to their website,[22] this app was developed with three goals in mind:

1. Prevention and auto-diagnosis;

[21] The reason why the app is signed with a Google certificate might be a consequence of the developers using Google Play's App Signing service.

[22] `https://www.euskadi.eus/coronavirus-app-covid-eus/web01-a2korona/es/`

2. Tracking and assisting subjects with symptoms in their homes—in fact, it offers an IM service to enable direct access to medical services—;
3. Analyzing the concentration of positive cases. Additionally, users can track the health state of people in their social network (known as "circles") by requesting access to the address book.

We tested versions 1.0 and 1.2, the latter was released on the 4th of April and was the latest one at the time of writing this report. We have not observed any significant change in terms of personal data collection between these two versions. The developers indicate in their Google Play profile that version 1.2 incorporates a user settings panel that allows users to edit their profiles (*e.g.,* setting their ages, names, or whether they could be considered more vulnerable to COVID-19, and reporting whether the symptoms remain), and delete contacts added to their "circles".

5.3.1 User Authentication

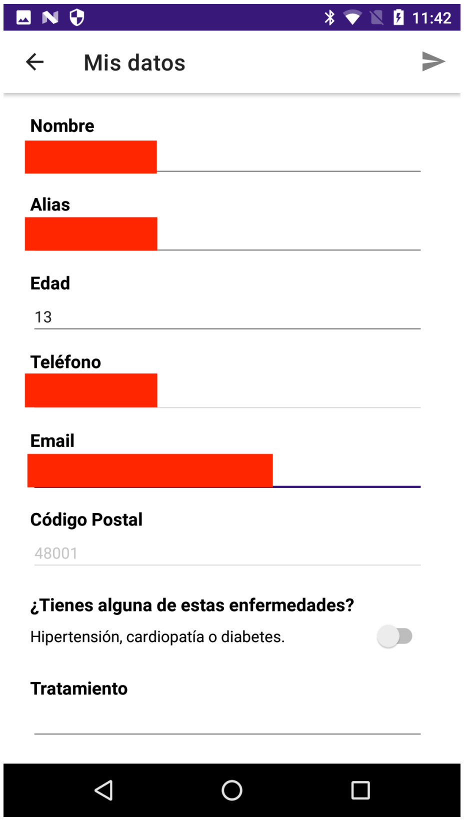
For registration, the user must provide an email address, phone number and postal code. As in the case of CoronaMadrid, this app also implements an SMS-based authentication mechanism to verify user accounts. We note that this mechanism is easy to circumvent using disposable text messages, thus potentially compromising data integrity and quality.

Age control: The privacy policy indicates that users under the age of 14 must have the authorization of their parents or legal guardian before using the app. The privacy policy states that they may verify subjects' ages after collecting the data, but does not state how they will treat it in that case. As with the other apps studied here, the software running on the client does not contain any age verification mechanisms.

5.3.2 Android Permissions

The table below depicts the normal and "dangerous" permissions—those requiring user approval, highlighted in red—that the Android app had requested, as well as whether the app was observed making use of those permissions (e.g., to access sensitive user data) during dynamic analysis. In blue, we highlight "custom permissions" associated with third-party services.[23]

[23] Android allows app developers to define their own permissions. By defining custom permissions, an app can share its resources and capabilities with other apps.



Mis datos

Nombre

[redacted]

Alias

[redacted]

Edad

13

Teléfono

[redacted]

Email

[redacted]

Código Postal

48001

¿Tienes alguna de estas enfermedades?

Hipertensión, cardiopatía o diabetes.

Tratamiento

Figure 3: Lack of UI notifications and age-limit controls when minors register in the service.

Permission	Protection Level
`android.permission.INTERNET`	Normal
`android.permission.ACCESS_NETWORK_STATE`	Normal
`android.permission.WRITE_EXTERNAL_STORAGE`	Normal
`android.permission.WAKE_LOCK`	Normal
`android.permission.RECEIVE_BOOT_COMPLETED`	Normal
`android.permission.FOREGROUND_SERVICE`	Normal
`android.permission.ACCESS_COARSE_LOCATION`	Dangerous
`android.permission.ACCESS_FINE_LOCATION`	Dangerous
`android.permission.CAMERA`	Dangerous
`android.permission.READ_CONTACTS`	Dangerous

The list of permissions requested by this app differs from that of other apps in this report. The need for gaining access to users' contacts might be justified by the social awareness features developed by the app. The camera permission might be requested to add a user profile picture. During our manual testing, we did not observe any notifications regarding access to user geolocation data.

5.3.3 Third-party SDKs

The libraries identified in the source code are listed below:

SDK	Provider	Package	Type
Android Support v4	Google	android/support/v4	Development Support
Google Gson	Google	com/google/gson	Development support
EventBus			Development support
Google Mobile Services	Google	com/google/android/gms	Development support, advertising services, mapping, push noti-ficatios and analytics.
Firebase	Google	com/google/firebase	Development support, cloud integration, analytics, A/B testing, push notifications, crashlytics, and authentication

We note that the AndroidManifest.xml file also declares various services related to Google's Firebase. The privacy policy does not specifically list and Google-related services.

5.3.4 Traffic Analysis and Backend Infrastructure

During the manual dynamic execution of the app using the AppCensus testing platform, we identified connections to:

Domain	IP Address	X.509 Certificate	Hosting Provider
covid.erictel.com	185.118.56.139[24]	OU = Domain Control Validated, OU = EssentialSSL Wildcard, CN = *.erictelm2m.com	Blackslot S.L.
www.euskadi.eus	62.99.63.78[24]	EAEko Herri Administrazioen CA - CA AAPP Vascas (2) SANs: URI:http://www.izenpe.com, email:info@izenpe.com, DirName: O = IZENPE S.A. - CIF A01337260-RMerc.Vitoria-Gasteiz T1055 F62 S8, street = Avda del Mediterraneo Etorbidea 14 - 01010 Vitoria-Gasteiz Organization: IZENPE S.A. Org. Unit: AZZ Ziurtagiri publikoa - Certificado publico SCA Location: ES	Euskaltel
www.geo.euskadi.eus	212.55.11.73[24]	Common name: EAEko Herri Administrazioen CA - CA AAPP Vascas (2) SANs: URI:http://www.izenpe.com, email:info@izenpe.com, DirName: O = IZENPE S.A. - CIF A01337260-RMerc.Vitoria-Gasteiz T1055 F62 S8, street = Avda del Mediterraneo Etorbidea 14 - 01010 Vitoria-Gasteiz Organization: IZENPE S.A. Org. Unit: AZZ Ziurtagiri publikoa - Certificado publico SCA Location: ES	Euskaltel
firebaseinstallations.googleapis.com	66.102.1.95[25]	Common name: *.storage.googleapis.com SANs: *.storage.googleapis.com, *.appspot.com.storage.googleapis.com, *.commondatastorage.googleapis.com, *.content-storage-download.googleapis.com, *.content-storage-upload.googleapis.com, *.content-storage.googleapis.com, *.googleapis.com, *.storage-download.googleapis.com, *.storage-upload.googleapis.com, *.storage.select.googleapis.com, commondatastorage.googleapis.com, storage.googleapis.com, storage.select.googleapis.com, unfiltered.news Organization: Google LLC Location: Mountain View, California, US	Google Cloud

TLS Encryption: The app always used HTTPS for connecting with the aforementioned online services, regardless of the hostname.

5.3.5 Personal Data Dissemination

The table below reports the information that has been disseminated by the app to online services as captured by our device instrumentation. This test has been performed with all the permissions being granted to the app.

[24] Approximate location based on MaxMind's GeoIP2 service: Spain
[25] Approximate location based on MaxMind's GeoIP2 service: U.S.

Domain	PII
covid.erictel.com	• Alias • Name and Family Name • Email • Phone number • Postal code • Push ID (Hash of IMEI, and device Serial Number) • Convid symptoms and diagnosis • Company ID • Whether a household member is consider vulnerable

The code snippet below shows how the app creates a unique identifier labeled as `PushID`. It is generated by invoking a method in the class: `com.erictelm2m.colabora.main.MyFirebaseMessagingService`.

This ID is generated by applying a hash function to unique identifiers like the IMEI and device serial number. We note that a hash of a unique ID will still be unique, and is therefore itself another identifier. The code suggests that this value is also used for Google's push notification service.

Example 5: Push ID generation (com.erictelm2m.colabora.main.MyFirebaseMessagingService)

```
private void updatePushId(String str, String str2, final String str3) {
        HashMap hashMap = new HashMap();
        hashMap.put("op", "updatePushid");
        hashMap.put("serial", str);
        hashMap.put("imei", str2);
        hashMap.put(Constants.pushid, str3);
        HyperLog.i(TAG, "parametros:" + hashMap);
        MySingleton.getInstance(getApplicationContext()).addToRequestQueue(new
                JsonObjectRequest(1, "https://covid.erictel.com:443/mobileController.m2m",
                new JSONObject(hashMap), new Response.Listener<JSONObject>() {
            public void onResponse(JSONObject jSONObject) {
                try {
                    if (jSONObject.getString("code").equals(Message.TEXT)) {
                        MyFirebaseMessagingService.this.storeRegistrationId(str3);
                        HyperLog.i(MyFirebaseMessagingService.TAG,
                                "UPDATE_PUSH_ID:␣se␣ha␣actualizado␣correctamente");
                        return;
                    }
                    HyperLog.i(MyFirebaseMessagingService.TAG,
                        "UPDATE_PUSH_ID:␣NO␣se␣ha␣actualizado␣pushID:␣" + jSONObject.get("text"))
                } catch (Exception e) {
                    HyperLog.e(MyFirebaseMessagingService.TAG,
                        "UPDATE_PUSH_ID:␣Exception:␣" + e.getMessage());
                }
            }
        }, new Response.ErrorListener() {
            public void onErrorResponse(VolleyError volleyError) {
                HyperLog.e(MyFirebaseMessagingService.TAG, "VolleyError:updatePushId:" +
                        volleyError.getMessage());
            }
        }));
    }
```

We could also identify the declaration of the `User` class, which contains the alias, company, country, postal code, phone number, picture, and medical conditions among many other variables. Not all of the parameters seem to be declared and used in the app version 1.0.

Example 6: User class.

```
public class User {
    private int age;
    private String alias;
    private String company;
    private String country;
    private String cp;
    private String email;

    /* renamed from: id  reason: collision with root package name */
    private String f14id;
    private boolean isConfirmed;
    private boolean isRisk;
    private String name;
    private String phone;
    private String photo;
    private boolean sensible;
    private int state;
    private String treatment;
    private long ts;
```

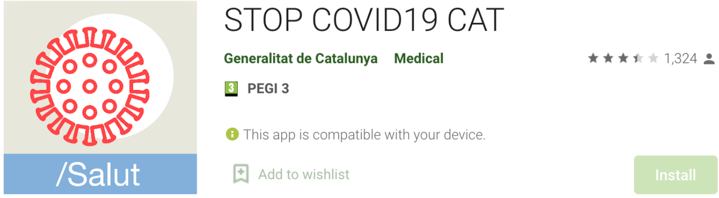
Minors: The app does not request the user's age when accessing the service for the first time, and it is only introduced after registration if the user voluntarily sets up their profile. Even when a minor introduces their age, the app does not provide any visual notification informing the user about the age limit and still, personal data is being uploaded to `covid.erictel.com` as shown below:

Example 7: Data of a 13 year-old user uploaded to `covid.erictel.com` over HTTPS

```
Content-Type: application/json
User-Agent: Dalvik/2.1.0 (Linux; U; Android 7.1.2; AOSP on BullHead Build/BUILD)
Host: covid.erictel.com
Connection: Keep-Alive
Accept-Encoding: gzip
Content-Length: 131

{"alias":"ANONYMOUS","medicine":"","age":13,"sensible":true,"risk":false,"name":"NAME","email":"ANONYMOUS@gmail.com"}
```

5.4 STOP COVID19 CAT


STOP COVID19 CAT

Generalitat de Catalunya Medical

PEGI 3

This app is compatible with your device.

Package Name	cat.gencat.mobi.StopCovid19Cat
Version	1.0.1 (20th of March, 2020)
MD5	8e3160ee3e75d1f529cae4c3cc165114
Developer Information	Generalitat de Catalunya (mobilitat.ctti@gencat.cat)
App Signature	Owner: CN=Ricard Mateu, OU=Direccio General d'Atencio Ciutadana i Difusio, O=Generalitat de Catalunya, L=Barcelona, ST=Catalunya, C=ES Issuer: CN=Ricard Mateu, OU=Direccio General d'Atencio Ciutadana i Difusio, O=Generalitat de Catalunya, L=Barcelona, ST=Catalunya, C=ES Serial number: 50053646
Certificate Fingerprint (MD5)	4C:85:AF:66:30:CF:5E:D6:A3:B9:26:D0:AC:3D:28:72
Installs	500,000+
Privacy Policy	http://sem.gencat.cat/ca/061CatSalutRespon/apps-mobils/STOPCOVID19/condicions-seguretat
Data Controller	Servei Català de la Salut

STOP COVID19 CAT was released by the Catalan Health Service (CatSalut) on the 20th of March. We tested version v1.0.1, released on the 20th of March. According to the Catalan health service, this app was developed with the objectives of:

1. Alleviating the load in emergency call centers and hospitals by helping citizens diagnose themselves based on their symptoms.
2. Identifying COVID-19 patients and tracking their response.
3. Detecting and monitoring areas with higher infection rates.

This app was released by a public health service free of charge. As a result, it does not directly engage in any advertising-based monetization. However, as we will see in this report, it integrates services offered by Mubiquo,[26] a company that specializes in mobile marketing solutions.[27]

5.4.1 User Authentication

The primary method for authentication is the official card ID number issued by the Catalan health system. In the case that users do not have a valid card (*e.g.,* expats or temporary residents not covered by Catalan social security), the app allows users to register with their national ID number, passport number, or NIE. In a second phase, the app requires users to introduce their phone number. As with the previous apps, these authentication mechanisms are easy to circumvent with disposable SMS numbers and fake ID generators. This might allow malicious actors to launch campaigns to pollute the data.

5.4.2 Android Permissions

The table below depicts normal and "dangerous" permissions—those requiring user approval—that the Android app had requested, as well as whether the app was observed making use of those permissions (e.g., to access sensitive user data) during dynamic analysis. In blue, we highlight "custom permissions" associated with third-party services.[28]

Permission	**Protection Level**
`android.permission.INTERNET`	Normal
`android.permission.ACCESS_NETWORK_STATE`	Normal
`android.permission.READ_EXTERNAL_STORAGE`	Normal
`android.permission.WRITE_EXTERNAL_STORAGE`	Normal
`android.permission.WAKE_LOCK`	Normal
`android.permission.VIBRATE`	Normal
`android.permission.RECEIVE_BOOT_COMPLETED`	Normal
`android.permission.RECEIVE_BOOT_COMPLETED`	Normal
`android.permission.ACCESS_COARSE_LOCATION`	Dangerous
`android.permission.ACCESS_FINE_LOCATION`	Dangerous
`android.permission.READ_PHONE_STATE`	Dangerous
`com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE`	Custom permission: Firebase-related

[26] https://www.mubiquo.com/

[27] https://www.crunchbase.com/organization/mubiquo

[28] Android allows app developers to define their own permissions. By defining custom permissions, an app can share its resources and capabilities with other apps.

The permission READ_PHONE_STATE allows reading sensitive user data such as the phone number of the device, current cellular network information, the status of any ongoing calls, and unique identifiers like the IMEI.

The permission BIND_GET_INSTALL_REFERRER_SERVICE is required by Google's Firebase Analytics for the reporting of install referrer campaign information. It is a requirement imposed by Google's Play Install Referrer API.

5.4.3 Third-Party SDKs

The third-party libraries identified in the app's source-code are listed below:

SDK	Provider	Package	Type
OKHTTP3.0	OKHTTP	com/squareup/okhttp	Development support
Google Gson	Google	com/google/gson	Development support
AndroidAnnotations	Open Source	org/androidannotations	Development support
Google Mobile Services	Google	com/google/android/gms	Development support, advertising services, mapping, push notifications and analytics.

The third-party libraries highlighted in red have been known to collect personal information. In this case, the app included Google's Mobile Services. Their presence is not mentioned in the privacy policy of the app, though they were not observed collecting personal information during testing.

5.4.4 Traffic Analysis and Backend Infrastructure

The whole online infrastructure supporting this app is hosted by Amazon Web Services. However, at the hostname level, we can observe the presence of a domain associated with a Spanish company offering mobile marketing solutions: Mubiquo.

Domain	IP Address	X.509 Certificate (Trust chain)	Hosting / Description
api.backendcovid19.net	13.33.235.42[29]	Common name: *.backendcovid19.net SANs: *.backendcovid19.net Valid from March 14, 2020 to April 15, 2021 Serial Number: 0bf71283e6a15950cdb9ca099977518f Signature Algorithm: sha256WithRSAEncryption Issuer: Amazon	Amazon CloudFront
location.backendcovid19.net	54.76.67.216[30]	depth=4 C = US, O = "Starfield Technologies, Inc.", OU = Starfield Class 2 Certification Authority; depth=3 C = US, ST = Arizona, L = Scottsdale, O = "Starfield Technologies, Inc.", CN = Starfield Services Root Certificate Authority - G2; depth=2 C = US, O = Amazon, CN = Amazon Root CA 1; depth=1 C = US, O = Amazon, OU = Server CA 1B, CN = Amazon; depth=0 CN = *.execute-api.eu-west-1.amazonaws.com	Amazon Web Services
mmm.mubiquo.com	52.0.78.156[29]	depth=4 C = US, O = "Starfield Technologies, Inc.", OU = Starfield Class 2 Certification Authority; depth=3 C = US, ST = Arizona, L = Scottsdale, O = "Starfield Technologies, Inc.", CN = Starfield Services Root Certificate Authority - G2; depth=2 C = US, O = Amazon, CN = Amazon Root CA 1; depth=1 C = US, O = Amazon, OU = Server CA 1B, CN = Amazon; depth=0 CN = mubiquo.com	Amazon AWS
s3.amazonaws.com	52.216.165.253[29]	depth=2 C = IE, O = Baltimore, OU = CyberTrust, CN = Baltimore CyberTrust Root; depth=1 C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert Baltimore CA-2 G2; depth=0 C = US, ST = Washington, L = Seattle, O = "Amazon.com, Inc.", CN = s3.amazonaws.com	Amazon AWS

With the information available, including DNS WHOIS and IP blocks, we cannot conclude whether the `*.backendcovid19.net` hostnames are a third-party or a first-party service. To improve the transparency of the service, we would recommend using complete and more meaningful X.509 certificates, issued by a trustworthy Certificate Authority (CA).

The traffic observed for the `s3.amazonaws.com` domain indicates that this host is used for delivering static content such as questionnaires, privacy policies, and medical guidelines. Mubiquo's service also notifies users periodically, encouraging them to re-check their symptoms.

TLS Encryption: The app always uses HTTPS for connecting with online services, regardless of the hostname.

[29] Approximate location based on MaxMind's GeoIP2 service: US

[30] Approximate location based on MaxMind's GeoIP2 service: Ireland

5.4.5 Data Exfiltration

When the app is granted all the requested permissions, we observed the following types of personal data transmitted to the following hostnames:

Domain	PII
api.backendcovid19.net	• NIE/DNI/Passport number • Age • Phone number • Symptoms • Chronic and previous medical conditions • Location (Latitude/Longitude) • MID (UUID)
location.backendcovid19.net	• Location (Latitude/Longitude) • MID (UUID)
mmm.mubiquo.com	• Location (Latitude/Longitude) • MID (UUID)

The privacy policy does not list the personal data types collected by the app and the role of the different services. The `MID` variable is a hash of a random user ID appended to the package name of the app. It is generated by the client and shared across three different services. Sharing IDs across services operated by different organizations eases database linkage. This is concerning from a privacy standpoint as this UID is uploaded alongside other persistent unique identifiers such as the national ID number, location, and phone number of the user, and also increases the risks of potential data breaches.

Example 8: JSON uploaded to `api.backendcovid19.net` containing medical information, national ID number, phone number, age, geolocation, symptoms and location over HTTP2. The MID identifier is highlighted in red.

```
"page1":{"dificultad\_respiratoria":true, "fiebre":true, "malestar\_general":true, "tos\
    _persistente":true},
"page2":{"edad":25, "enfermedad\_cancer":false, "enfermedad\_cardiovascular":false, "enfermedad\
    _diabetes": true, "enfermedad\_hepatica\_cronica":false, "enfermedad\_inmunodeficiencia":
    false,
"enfermedad\_neuro\_cronica":false, "enfermedad\_pulmonar\_cronica":false, "enfermedad\_renal\
    _cronica" :false,
"genero":2, "sintoma\_dispnea":false, "sintoma\_dolor\_costado":true, "sintoma\_hemoptisi":true,
"situacion\_embarazada":false,  "situacion\_lactacia" false, "situacion\_medicado\
    _inmunodepresores":false, "situacion\_postparto":true},
"page3":{"actual\_dificultad\_respiratoria":1, "actual\_malestar\_general":2, "actual\_tos":true
    , "actual\_temperatura":1},
"page4": {"estado\_actividades\_basicas":true, "form\_version":"1.0.0", "language":"en", "
    location\_latitude":40.XYZ, "location\_longitude":-3.XYZ,
"mid":"1e59c0d9459959d46e3005e1646d7e43a25af606", "modified":0, "phone":"602545766"}
```

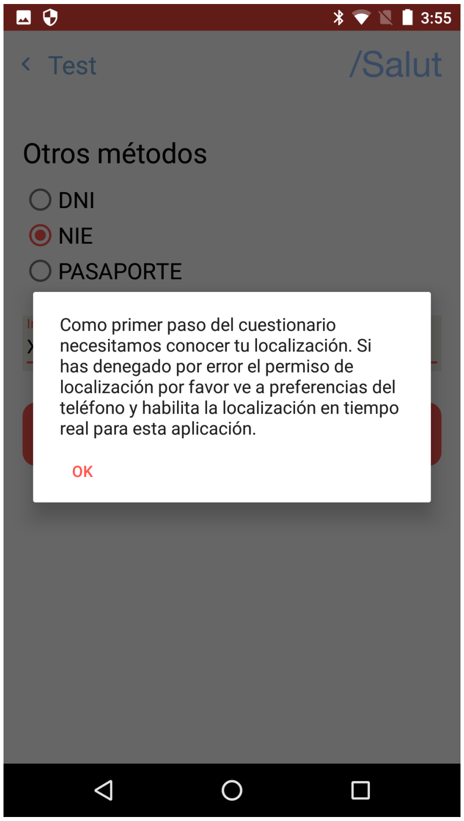
Example 9: Network flow to `location.backendcovid19.net`. The MID identifier and geolocation data are highlighted in red.

```
PRI * HTTP/2.0

SM

..............................._.........‘u...b..i1.C...A....Lz..d.Z.!..@..EO.X \\
...!’.._..u.b&=LtA..P....OZ.V{\.173P....z.?Y...ew..........{"ts":1585783121,␣\\
"key":"7527220e1392bdae689768629daaafdc9a6f99844a7b546c9b117ae82789f3cd",␣\\
"mid":"1e59c0d9459959d46e3005e1646d7e43a25af606","lat":40.XYZ,"lon":-3.XYZ}
```

Denying geolocation permissions: According to the privacy policy, the collection of some data types are optional. Likewise, users can deny the app's access to sensitive data by configuring their privacy settings. However, STOP COVID19 CAT does not allow users to proceed and access the service if geolocation is disabled, as shown in Figure 4:



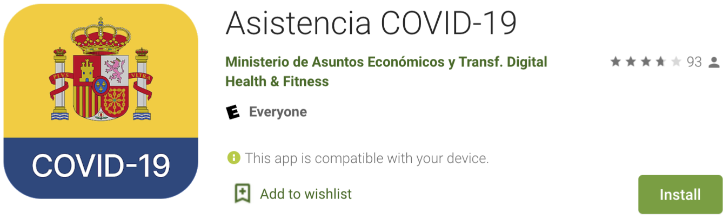
Otros métodos

○ DNI
● NIE
○ PASAPORTE

Como primer paso del cuestionario necesitamos conocer tu localización. Si has denegado por error el permiso de localización por favor ve a preferencias del teléfono y habilita la localización en tiempo real para esta aplicación.

Figure 4: Notification informing users that they need to enable geolocation to use the app.

5.5 Asistencia COVID-19



Asistencia COVID-19

Ministerio de Asuntos Económicos y Transf. Digital

Health & Fitness

★★★★☆ 93

Everyone

This app is compatible with your device.

Package Name	es.gob.asistenciacovid19
Versions	1.0.0 (6th of April)
MD5	5f200b75fa049f6ca278c7b88d066be4
Developer Information	contacto@covid19.gob.es
App Signature	Owner: CN=Asistencia COVID-19, OU=IT, O=Espana, L=Espana, ST=Espana, C=ES Issuer: CN=Asistencia COVID-19, OU=IT, O=Espana, L=Espana, ST=Espana, C=ES Serial number: bf3a7f1
Certificate Fingerprint (MD5)	5F:96:9D:4C:93:20:C8:E3:40:3B:C3:E9:59:67:3C:01
Installs	10,000+
Privacy Policy	https://asistencia.covid19.gob.es/politica-de-privacidad
Data Controller	Ministerio de Sanidad

The Asistencia COVID-19 app was published by the national government of Spain on the 6th of April following the national plan to alleviate the impact of COVID-19 (Orden SND/297/2020 de 27 de marzo). According to media sources, this app is based on the technology developed for the CoronaMadrid app and it is only offered to citizens of specific Spanish regions.

There is evidence to suggest that both apps were developed by the same organizations. The UI (see Figure 5) and the features in Asistencia COVID-19 are almost identical to those in CoronaMadrid. The traffic behavior, most of the code (including permission requests and embedded third-party SDKs), and the types of personal data transmitted to the cloud are also almost identical between the two apps. Asistencia COVID-19's privacy policy follows a structure similar to CoronaMadrid's, and also does not report the presence of the Crashlytics SDK.

As opposed to CoronaMadrid, this app is not signed with an official government certificate. In fact, it does not provide any app developer identity. This impedes accurately identifying the publisher, which could raise trust concerns for certain users.

5.5.1 User Authentication

The user authentication method is identical to the one implemented in CoronaMadrid. However, the app suggests that users located in Madrid install the CoronaMadrid app.

The app requires users to validate a phone number (done through Firebase two-factor authentication services), as well as provide a valid ID number (either DNI or NIE) to register for the service. While the privacy policy and terms of service of the app state that the user must provide real data when registering, it is trivial to register for the service using fake information provided by disposable SMS services[31] and ID generators. Nevertheless, we cannot discount the possibility of the service providers performing offline verification and data sanitization on the server-side. In fact, the privacy policy states that the service provider can disable the account if they identify a violation of their terms of service. If not, the lack of verification mechanisms may allow malicious actors to intentionally pollute the data collection or otherwise degrade the utility of the service.

Age control: As in the case of CoronaMadrid, Asistencia COVID-19's privacy policy states that the user must be over 16 years of age or have the consent of a parent or legal guardian. We ran a separate test, pretending to be a 9 year old boy to study the behavior of the app in this scenario. When introducing the date of birth, the app denied access. However, if the user introduces a valid date of birth, the app will operate as normal without performing any robust age verification.

5.5.2 Android Permissions

The table below depicts the normal and "dangerous" permissions—those requiring user approval—that the Android app had requested, as well as whether the app was observed making use of those permissions (e.g., to access sensitive user data) during dynamic analysis. In blue, we highlight "custom permissions" associated with third-party services.[32]

[31] E.g., `https://smsreceivefree.com/`

[32] Android allows app developers to define their own permissions. By defining custom permissions, an app can share its resources and capabilities with other apps.

Permission	Protection Level
android.permission.INTERNET	Normal
android.permission.ACCESS_NETWORK_STATE	Normal
android.permission.ACCESS_WIFI_STATE	Normal
android.permission.ACCESS_COARSE_LOCATION	Dangerous
android.permission.ACCESS_FINE_LOCATION[33]	Dangerous
com.google.android.c2dm.permission.RECEIVE	Custom permission: Push notifications
com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	Custom permission: Firebase-related

According to the Firebase documentation, the "custom permissions" requested by its SDK are required to report events to the server, such as installation events or scheduling tasks, possibly for analytics purposes. (Firebase is owned by Google.) We also observed the custom permission `com.google.android.c2dm.permission.RECEIVE` which is required by Google's push notification service (*i.e.*, server-to-client notifications and communications).

5.5.3 Third-Party SDKs

The app's code contains various third-party libraries. Many of them are open-source libraries offering development support, including those maintained and developed by large companies like Facebook and Google. For example, there are several Facebook libraries present for easing the development and deployment of cross-platform mobile apps, such as the React Native SDK,[34] Facebook's Yoga[35] and Facebook's Hermes.[36]

The third-party libraries identified in the source code are listed below:

SDK	Provider	Package	Type
Android Support v4	Google	android/support/v4	Development Support
Google Core Libraries for Java 6+	Google	com/google/common	Development support
Google Core Libraries	Google	com/google/thirdparty	Development support
Google Gson	Google	com/google/gson	Development support

[33] As of Android 10, gaining access to the BSSID (*i.e.*, MAC Address of the WiFi access point) also requires access to the `ACCESS_FINE_LOCATION` permission. This change in Android's permission model was motivated by our prior published research demonstrating that several apps used this information to infer users' geolocation without requesting the location permission. (See https://www.usenix.org/system/files/sec19-reardon.pdf.) Users running lower Android versions are still vulnerable to this side-channel attack.

[34] https://reactnative.dev/

[35] https://yogalayout.com/

[36] https://github.com/facebook/hermes

OkHttp	OkHttp	com/squareup/okhttp	Development support
Hermes	Facebook	com/facebook	Development support
Yoga	Facebook	com/facebook	Development support
Fabric[37]	Google	io/fabric/sdk/android	Development support
Bolts Base Library	Parse/Facebook	bolts	Development support
Firebase	Google	com/google/firebase	Development support, cloud integration, analytics, A/B testing, push notifications, Crashlytics, and authentication.
Google Mobile Service	Google	com/google/android/gms	Development support, advertising services, mapping, push notifications, and analytics.

The third-party libraries highlighted in red are known to collect personal information from app users. This is the case for Google's Firebase suite[38] (which also includes Google's Crashlytics bug reporting and analytics solution)[39] and Google's Mobile Services.[40] The latter is a library offering APIs to integrate various Google services, including Google Maps, in mobile apps.

5.5.4 Traffic Analysis and Backend Infrastructure

During the manual execution of this app using the AppCensus platform, we identified connections to different cloud services. As can be seen, a number of thems are associated with the third-party services collecting personal and behavioral data that were previously identified by our static analysis: Crashlytics and Google's Geocoding API.[41]

Domain	IP Address	Server X.509 Certificate	Hosting / Description
settings.crashlytics.com	172.217.17.3[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = google.com	Google Crashlytics
europe-west1-covid19madrid.cloudfunctions.net	216.239.36.54[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = *.google.com	Service hosted in Google Cloud Services

[37] Deprecated: now known as Firebase.
[38] https://firebase.google.com/
[39] https://firebase.google.com/docs/crashlytics
[40] https://www.android.com/gms/
[41] https://developers.google.com/maps/documentation/geocoding/start
[42] Approximate location based on MaxMind's GeoIP2 service: U.S.

maps.google.com	172.217.17.14[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = *.google.com	Google Maps / Geocoding API
firestore.googleapis.com	216.58.209.74[42]	C = US, ST = California, L = Mountain View, O = Google LLC, CN = *.googleapis.com	Google Cloud Services
play.google.com	216.58.211.46[42]	Owner: CN=*.google.com, O=Google LLC, L=Mountain View, ST=California, C=US Issuer: CN=GTS CA 1O1, O=Google Trust Services, C=US	Google
webapp.covid19.gob.es	35.190.74.6[42]	Organization: ENTIDAD PUBLICA EMPRESARIAL RED.ES M.P Unit: Sistemas Location: MADRID, State/Province: Comunidad de Madrid Country: ES	Google Cloud

This analysis reveals that the online infrastructure is identical to the one used by CoronaMadrid with one exception. In Asistencia COVID-19, we can observe that the hostname `coronamadrid.comunidad.madrid` has been replaced by `webapp.covid19.gob.es` (IP 35.190.74.6), which is hosted on Google Cloud Services. This new hostname is responsible for offering health information and recommendations specific to the region of the user (if the region is supported). We refer the reader to the CoronaMadrid report for further information.

TLS Encryption: We observed that during testing, the app always used HTTPS for connecting with online services, regardless of the domain name contacted.

5.5.5 Data Exfiltration

When the app was granted all the requested permissions, we observed the following types of personal data transmitted to the following hostnames:

Domain	Personal Information
settings.crashlytics.com	• Installation ID (X-CRASHLYTICS-INSTALLATION-ID) • City
europe-west1-covid19madrid.cloudfunctions.net	• National ID number
maps.google.com	• User-introduced location (country, region, postal code, street name, street number)
firestore.googleapis.com	• Full name • User-introduced location (Country, region, postal code, street name, street number) • Corrected user-introduced location (as provided by Google Maps Geocoding) • National ID number / INE • Phone number • Device information • Gender • Device token • GPS location (latitude / longitude) • Local IP address • App-generated UID • COVID-19 symptoms • Self-assessment • Last report

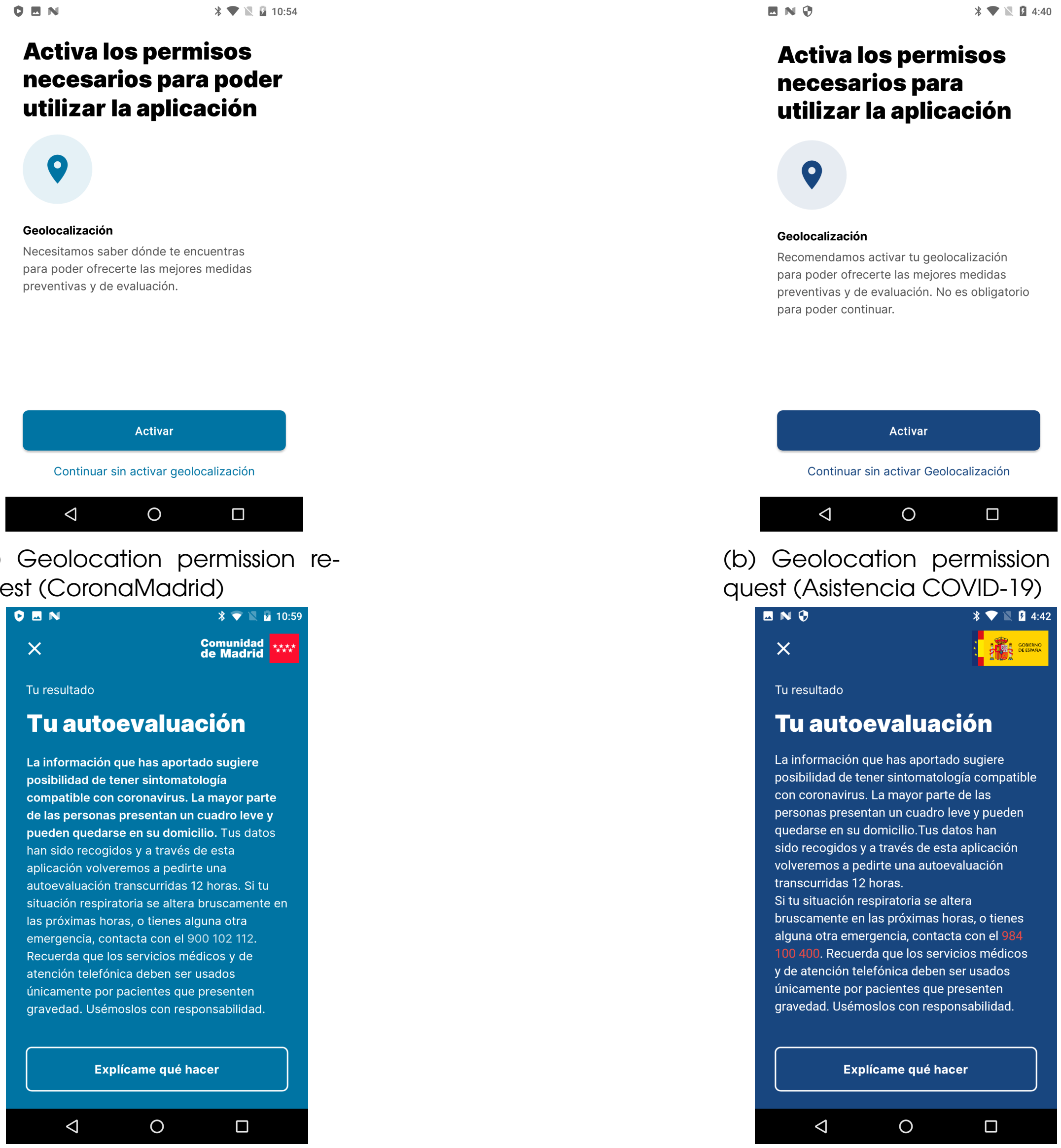
Asistencia COVID-19 uploads data to the same hostname and URI [43] that provides backend support to CoronaMadrid. This suggests that both apps (and their respective data controllers, according to their privacy policies) might share the database initially deployed for CoronaMadrid. This is confirmed by other configuration parameters present in the source code. However, the API Keys for Crashlytics, Firebase and Google Services are different. This suggests that each app developer might have separate access to their app-specific Crashlytics dashboard and Google Services.

Considering the technical information extracted from both apps, we cannot conclude whether the data controllers for CoronaMadrid and Asistencia Covid-19 are, as a result, independent or joint data controllers. There is no mention whatsoever to any joint controllership regarding any data processing in the information available in their respective privacy policies.

Finally, the privacy policy of this app did not inform users about the presence of Crashlytics—or any other Google-related service—at the time of running the tests. We note that none

[43] projects/covid19madrid/databases/

of the different custom UIDs generated by different app components seem to be shared with other domains and services. We refer the reader to CoronaMadrid´s report for further information.



10:54

Activa los permisos necesarios para poder utilizar la aplicación

Geolocalización

Necesitamos saber dónde te encuentras para poder ofrecerte las mejores medidas preventivas y de evaluación.

Activar

Continuar sin activar geolocalización

(a) Geolocation permission request (CoronaMadrid)

4:40

Activa los permisos necesarios para utilizar la aplicación

Geolocalización

Recomendamos activar tu geolocalización para poder ofrecerte las mejores medidas preventivas y de evaluación. No es obligatorio para poder continuar.

Activar

Continuar sin activar Geolocalización

(b) Geolocation permission request (Asistencia COVID-19)

10:59

Comunidad de Madrid

Tu resultado

Tu autoevaluación

La información que has aportado sugiere posibilidad de tener sintomatología compatible con coronavirus. La mayor parte de las personas presentan un cuadro leve y pueden quedarse en su domicilio. Tus datos han sido recogidos y a través de esta aplicación volveremos a pedirte una autoevaluación transcurridas 12 horas. Si tu situación respiratoria se altera bruscamente en las próximas horas, o tienes alguna otra emergencia, contacta con el 900 102 112. Recuerda que los servicios médicos y de atención telefónica deben ser usados únicamente por pacientes que presenten gravedad. Usémoslos con responsabilidad.

Explícame qué hacer

(c) Covid-19 assesment report (CoronaMadrid)

4:42

GOBIERNO DE ESPAÑA

Tu resultado

Tu autoevaluación

La información que has aportado sugiere posibilidad de tener sintomatología compatible con coronavirus. La mayor parte de las personas presentan un cuadro leve y pueden quedarse en su domicilio.Tus datos han sido recogidos y a través de esta aplicación volveremos a pedirte una autoevaluación transcurridas 12 horas.
Si tu situación respiratoria se altera bruscamente en las próximas horas, o tienes alguna otra emergencia, contacta con el 984 100 400. Recuerda que los servicios médicos y de atención telefónica deben ser usados únicamente por pacientes que presenten gravedad. Usémoslos con responsabilidad.

Explícame qué hacer

(d) Geolocation permission request (Asistencia COVID-19)

Figure 5: User interface comparison between CoronaMadrid and Asistencia COVID-19 apps.