In September 2020, SMEX conducted a [security analysis of the contact-tracing app, Ma3an,](https://smex.org/security-concerns-with-lebanons-new-contact-tracing-app/) to test the app’s security and safety. The[Ministry of Public Health in Lebanon (MoPH)](https://smex.org/security-concerns-with-lebanons-new-contact-tracing-app/) seemed responsive to our recommendations, but in reality, no changes were implemented.

This year, we are re-assessing Ma3an app’s security measures, among other governmental applications used in Lebanon. Below are the details of the new findings and recommendations, presented in comparison to last year’s security analysis.

**Applications Permission Analysis**

An app should not request permission to access third-party applications, unless they are essential for it to function. With that being said, most of the permissions found in the previous Ma3an application assessment were valid, but the app requests GPS permission (ACCESS\_FINE\_LOCATION) which allows it to access precise location using integrated GPS modules in devices.

**This means the app can identify every user’s location.** We understand the need for such a permission, since it works on a proximity algorithm to trace contacts via GPS and Bluetooth, but a more transparent approach to its use is required.

**Security Analysis**

“Cleartext traffic” configuration is enabled on the Ma3an app. In this case, data collected and sent to the Ma3an’s servers **is potentially exposed to interception if the content is not encrypted.** The app intends to use cleartext network traffic, such as HTTP, Download Manager and MediaPlayer for android. It is also important to note that this function is set by default and enabled for Android application programming interface (API) level 27 and lower. For Android versions working on API level above 28, cleartext traffic is disabled by default.

**Cleartext traffic should be avoided** as it lacks confidentiality. Interception might tamper with the traffic and transmit false information. For example, it allows a malicious actor to eavesdrop on the app’s data and modify it without being detected.

**Security Weakness**

**Remote WebView Debugging**

In our analysis, we noticed that nothing has changed in the Ma3an app[in comparison to last year’s analysis.](https://smex.org/security-concerns-with-lebanons-new-contact-tracing-app/)The Ma3an team did not implement any of our previous recommendations, despite public and private communication. T we noticed that nothing was changed in the Ma3an app according to our last year’s analysis, surprisingly! The Ma3an team did not implement anything of our previous recommendations, despite the public and private communication. **“Remote WebView debugging” is enabled**via the third-party library, [OneSignal](https://onesignal.com/). This feature is considered one of the top 10 mobile threats listed under the Open Web Application Security Project (Open Web Application Security Project ([OWASP)](https://owasp.org/www-project-mobile-top-10/),) which might be due to “improper use” of the platform.

Improper platform usage is when an application fails to make adequate use of a platform’s capabilities or rights. Features or permissions could have been implemented incorrectly or they aren’t being used.

This happens through two android features:

1. **INTENTS**: Android intents are used for messages requesting an action from another application component. Intents can be private or public, and they can be set to communicate only between applications by the same organization.

Public intents may compromise a user’s privacy and expose sensitive information.

1. **FILE PERMISSIONS**: If developers do not properly restrict access to program files, a malicious party may read or overwrite data. This allows a program to execute arbitrary code, a security flaw in software or hardware, where an attacker can execute system commands to a target machine to obtain special access or unauthorized information like credential theft on a local device.

**Ma3an developers [TedMob](https://tedmob.com/) should reassess the app’s need for OneSignal.** Remote WebView debugging enables attackers to inject malicious code and gain access over the android device.

**Information Storage**

In the [previous analysis](https://smex.org/security-concerns-with-lebanons-new-contact-tracing-app/), we also noticed the app uses Google’s [Firebase](https://firebase.google.com/) as a Software as a Service (SaaS) (SaaS) for back-end communication and information exchange.

However, the Firebase platform associated with the Ma3an app was not properly configured and needed to be checked.

The following URL [<https://moph-tracer.firebaseio.com/.json>] is not well-secured and might be exploited. We stopped our analysis at this point because it is beyond our legal authority to actively exploit the application, but we can state the following:

**For the previous URL two types of responses may appear:**

1. “Permission Denied” entails that the end-point is restricted, and the website is well-configured.
2. “Null,” or a JSON string, means that the database is public and read accesses are potentially possible. Read access is a feature in Firebase SaaS to read data from the document from which and on which Ma3an app reads and writes data. A malicious actor could attempt writing on Firebase endpoint, polluting data and possibly undermining its integrity.

Also the application uses SQLite database and executes raw SQL query. It is the local database/data-store used by android applications. The type of this database is SQLite and the communication “protocol” between SQLite and Android app is called “SQL query.” Untested users can inject raw SQL queries and can cause SQL injections leading to data leaks and data tampering on a user’s device locally. It might lead to the exposure of a user’s personal data within a Ma3an application to other untrusted applications. MOPH, through TedMob, should manage SQLite3 more securely. Sensitive information should be encrypted before being written to the database.

**Encryption**

The application is indeed using Secure Sockets Layer (SSL), that establishes authenticated and secure layers between servers and computers, on most of their end-points. As a result, application in-transit data is relatively secure.

**An important issue we found is about the application signature**. We noticed the application is signed with V1. signature scheme which makes it vulnerable to Janus vulnerabilities for Android versions lower than Android 7.0. Janus is a serious vulnerability in Android allowing attackers to inject a DEX file (file used to initialize and execute applications developed for the Android mobile OS) into an APK file (Android Package file) without affecting the signature. This facilitates modifying code in applications without affecting the digital integrity of the file. This vulnerability only affects Android versions lower than 7.0, but many people in Lebanon still use this version.

**Conclusion**

If we compare the listed problems with our previous assessment, even after our recommendations to Ma3an, we notice little to no evolution in the application’s vulnerability and user’s privacy from a technical standpoint. Ma3an did not put an effort into updating the application’s previously noted security issues and potential vulnerabilities.

**Based on our recent static analysis, we recommend and urge the Ministry of Public Health and the Ma3an app team to implement the following:**

1. Modify configuration on Firebase for <https://moph-tracer.firebaseio.com/.json> to return Access Denied instead of a read response of Null.
2. Abandon OneSignal because of its **Remote WebView debugging**abilities and find a more secure alternative.
3. Disable cleartext traffic in AndroidManifest.xml
4. Implement SQLite hardening on mobile devices.