KPMG

Security test potential Corona apps

Ministry of VWS
A2000020142

Final report
-
April 19, 2020

Table of contents

KPMG Advisory NV

PO Box 74500 1070 DB
Amsterdam

Laan van Langerhuize 1 1186
DS Amstelveen Telephone
(020) 656 7890

Confidential

Amstelveen, April 18, 2019

To the Board of the Ministry of VWS

Dear Mr. Roozendaal, Dear Ron,

In accordance with our proposal of April 17, 2020, with reference A2000020142, we hereby send you our report on our activities.

The purpose of the assignment as provided to us was to provide you with insight into the extent to which sufficient attention has been paid to the security and reliability of solutions as shown in the Appathon of 18 and 19 April last, consisting of one or more apps, backend systems and the communication with these systems.

Nature of the contract

The nature of the work means that we have not conducted an audit, review engagement or other assurance engagement. Therefore, no assurance regarding the fairness of financial or other information can be derived from this report.

Our advice in this report is based solely on the results of the agreed activities and the results thereof. If we had performed additional procedures, or performed an audit, review or assurance engagement, other matters that would have been eligible for reporting may have been identified. We will not adjust our reporting for future changes, changes in laws and regulations or changed legal and administrative interpretations of laws and regulations.

Responsibility of the management of the Ministry of VWS

For completeness, we note that you are responsible for the correctness and completeness of the information made available to us in the context of the above activities. We accept no responsibility for the quality, correctness or completeness of the information supplied to us.

Distribution of the report

The report is exclusively intended for you as a client. Without our express and prior written permission, it is not permitted to use this report, or parts of this report, for other purposes, to make it public and / or to provide it to third parties. KPMG accepts no liability for the use of this report other than for which it was drawn up and made available to you as the client.

We thank for the open and constructive cooperation in the performance of our work and the preparation of our report. We are happy to provide further information.

Yours sincerely, KPMG

Advisory NV

Ing. JAM Hermans RE Partner

Background research.

On 11 April last, the Ministry of Health, Welfare and Sport (hereinafter: VWS), with the Invitation to Smart Digital Solutions Corona (hereinafter: "the Invitation"), asked the market to submit proposals in the following four areas:

1. Smart digital solutions that can contribute to source and contact tracing;

2. Smart digital solutions that can contribute to self-monitoring and remote guidance;

3. Other digital solutions that can contribute to the transition strategy;

4. Preconditions under which such solutions can be deployed.

With regard to part 1, seven parties were asked to participate in a so-called Appathon (which takes place / has taken place on 18 and 19 April 2020), with the aim of allowing VWS to determine in a transparent manner whether and to what extent the solution meets the principles stated in the Invitation.

Parallel to this Appathon, VWS asked KPMG to investigate the solutions offered by the seven parties, with the aim of providing insight into the extent to which sufficient attention has been paid to the security and reliability of these solutions, consisting of one or more apps, back-end systems and communication with these systems.

The research that took place per supplier consists of two parts, namely:

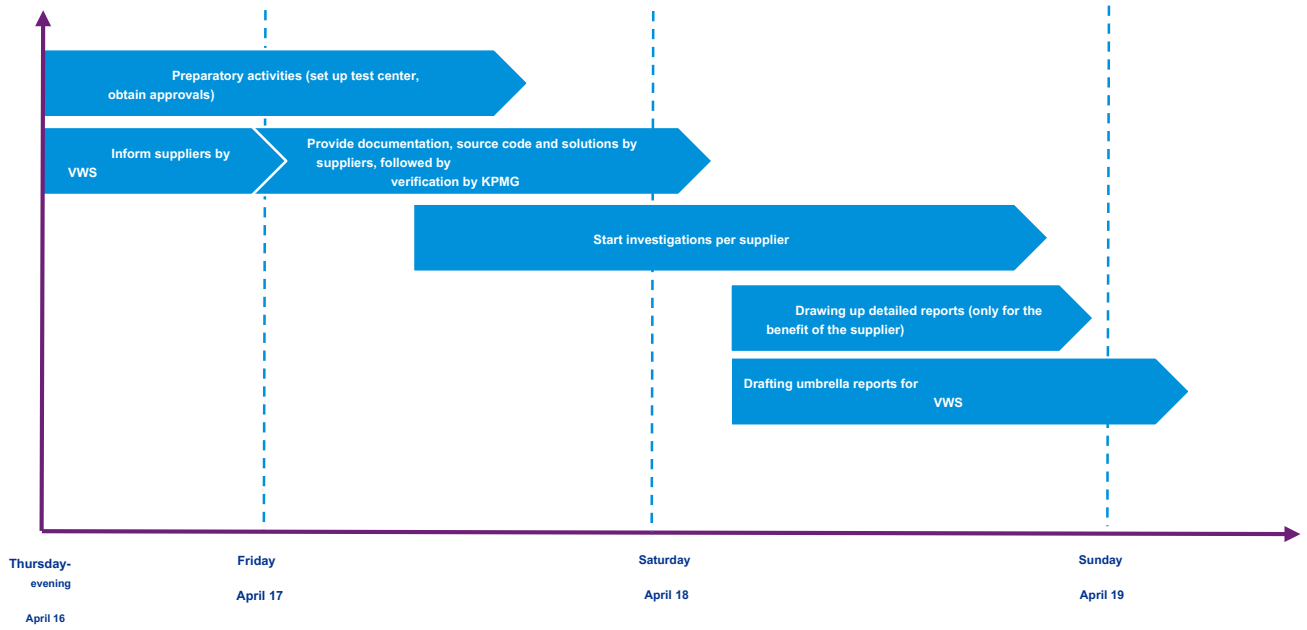
1. Limited initial penetration test.

2. Quick scan source code research.

During the research, a number of research questions were formulated for both parts, supported by a structured research approach, in order to ensure comparability of the aspects to be investigated per supplier. These research questions are further specified in this report.

Background research.

The study was characterized by the very short timelines, namely two days. The figure below shows the activities that took place during these two days.



In this report, we will present our findings from our investigation, grouping our findings into the following categories:

- General results
- Findings per research question

Research questions limited initial penetration test

KPMG has used the following research questions in the execution of the limited initial penetration test. These research questions are plotted on the test methodology used and consistently performed for all applications in scope, where possible.

1.	Can a malicious person acquire sensitive information by being able to communicate unauthorized with the underlying infrastructure of application X?	**Highest priority**
2.	Can a malicious person compromise or severely disrupt the underlying infrastructure of application X by injecting malicious code? As a result, the availability, integrity and confidentiality of the system and data cannot be guaranteed.	
3.	Can a malicious person enter incorrect data or delete correct data through misuse of the underlying infrastructure of application X? As a result, the data in the underlying infrastructure becomes less reliable and it is no longer possible to properly determine which users have COVID19 infection and which do not.	
4.	Can a malicious party intercept and / or misuse communication between application X and the underlying infrastructure (the channel itself or, for example, via the API)? This scenario has a higher chance of using public hotspots such as guest Wi-Fi and / or public hotspots.	
5.	Can a malicious person obtain sensitive data from the storage space of application X on the end user's mobile device? This scenario can occur if the mobile device is vulnerable and application X stores sensitive data (consciously or unconsciously) on the device. This can affect the privacy of the user as well as possible other users (if data is also stored about this because these users were nearby, for example).	
6.	Can malicious communication between application X and the underlying infrastructure compromise through misuse of channels other than the primary channel (the expected API interface)? Think of additional channels such as e-mail, SMS or other TCP / UDP network interfaces.	**Lowest priority**

Research questions limited initial penetration test

KPMG has used the following research questions when carrying out the quick scan source code research. These research questions have been converted into a “car wash” where automated and manual examinations are performed consistently for all applications in scope, where possible.

7.	What is the general picture of the (technical) quality of the source code of the application?
8.	Do automatic tooling in the “car wash” reveal any real vulnerabilities regarding the reliability and security in the source code?
9.	Are there other data outputs (including logging) in the source code that are not defined in the design?
10.	Does the supplied software (both front-end and back-end) depend on external libraries for proper operation? If so, are these libraries current, are they maintained and / or do they contain known vulnerabilities with regard to reliability and security?
11.	Is the source code set up in line with our expectations from the technical and functional documentation? (For example, have the described privacy mechanisms been implemented, does the scope fit the description, etc.)?

Limitations of the study

Partial research with limited depth

- During our investigation we were able to investigate the apps used, the backend as well as communication between the components with a limited depth (due to the short timelines).

- In addition, we would like to point out that the manner of implementation of the complete concept and compliance with the correct management and security measures are decisive for being able to meet the security and privacy requirements required for these concepts. We would then like to emphasize that an integral security and privacy investigation focused on the entire concept consisting of organization, processes and technology is important before proceeding to full implementation.

Ensuring the anonymity of the suppliers

- The results of the study are aimed at providing general insights to be used by the committee when questioning the suppliers. The results of the study are not intended as a selection tool. No references to specific suppliers will be made in this report.

Given the very short turnaround time of the study, we would like to point out the following limitations of the study:

- A limited number of detailed test activities have not been able to take place, including due to the late availability of the source code, the required test environment and the use of tools for this test. This means that not all issues may have been found. In our reports, situations where no detailed testing activities have taken place have been identified.

- For a limited number of findings, no additional in-depth further investigation has been possible, with the result that some of these findings could possibly be described as a so-called "false-positive" after further investigation.

- Detailed findings of the penetration tests and source code reviews have not been coordinated in detail with the supplier (so-called "rebuttal"). This may mean that a number of detailed findings need to be adjusted, based on this validation.

- We conducted the research based on the documentation, source code, apps and backend systems provided by the supplier. We have not been able to determine whether the supplied documentation and source code is complete. In addition, adjustments may have been made to the apps, backends or source code during our investigation. These changes have not been included in the results of our investigation.

Limitations of the study - continued

Limitations on the limited initial penetration tests:

- Testing iOS apps carries certain inherent limitations. For example, Apple has secured its operating system (iOS) in such a way that we (depending on the iOS version) cannot test everything. We can only test to a limited extent and therefore not cover all versions / equipment combinations.

- Mobile devices have certain inherent vulnerabilities that are not investigated when performing a penetration test on a mobile application. For example, certain unique identifiers of identified Bluetooth devices may be stored in the log files of mobile devices. Such inherent vulnerabilities are not investigated during an application penetration test.

- A security test can only demonstrate whether it is possible to breach existing security measures. It cannot be stated with certainty that implemented security measures cannot be breached. An attacker with unlimited budget, knowledge and time will almost always succeed in breaking through security. The activities of KPMG also involve a time-related effort, which means that not all possible weaknesses can be tested exhaustively.

- When performing a security test, the weaknesses and attack techniques known to KPMG at that time are used. The result of a security test is therefore always a snapshot regarding both knowledge of weaknesses and attack techniques, and security measures taken.

Limitations regarding the quick scan source code study:

- The quick scan source code investigation is limited to the source code supplied by the suppliers. Limited inquiries were made as to whether the software was complete and everything had been delivered.

- All apps use additional standard software components (such as operating systems, database and web server software), network facilities and protocols. Vulnerabilities in these components have not been identified.

- There is a time-related effort and the tooling used is not the same for every software platform. This means that not all issues may have been found and that not all findings could be fully validated.

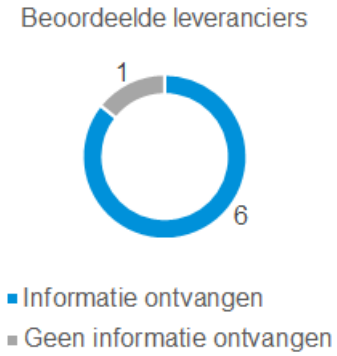
General findings and observations

General findings - participating suppliers

Suppliers included in the study

On the evening of April 16, seven participants of the Appathon were informed by VWS about their participation and the information needs of KPMG. KPMG has conducted an intake with the various suppliers and validated whether the required information (consisting of technical documentation, source code, app (s) and other required information) has been supplied by the supplier.

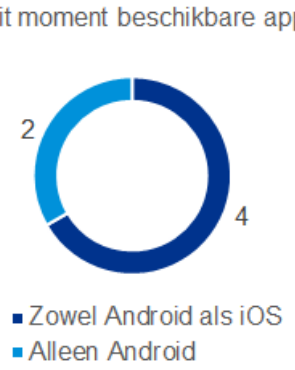
Based on the validation in which it was established that the necessary information could not be provided, it was decided to exclude one of the suppliers from further investigation.



Therefore, no findings with regard to this supplier are presented in the remainder of this report.

Available apps related to Operating System

During the validation of data, it has also been found that not all suppliers currently have an app on both the Android and iOS operating systems.

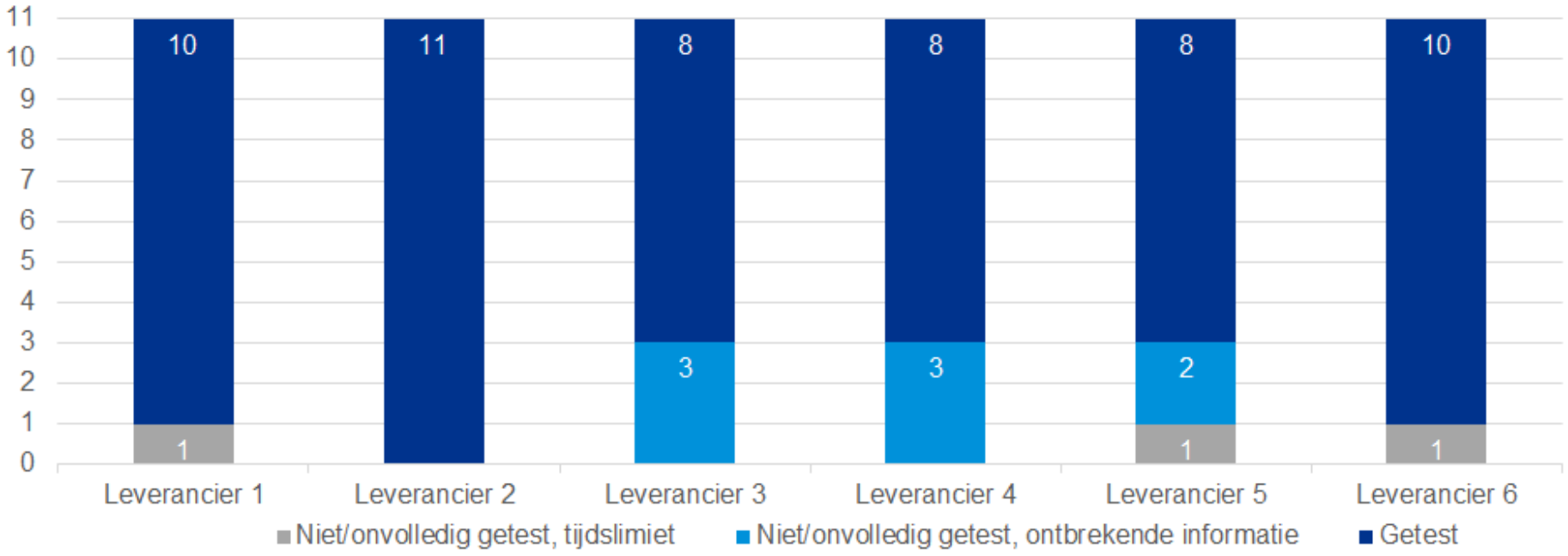


There were no vendors that only developed iOS-based apps.

General findings - research questions

As already explained in this report, we conducted research based on 11 research questions. During the investigation, we were unable to perform all test activities for answering the research questions at a number of suppliers, due to the unavailability of all relevant information, the availability of apps, partly due to the partial unavailability of test infrastructure and test tools, as well as lack of time of the testers.

The figure below shows for each supplier which research questions we have or have not been able to answer.



** The numbering of suppliers used in this report does NOT correspond to the list of suppliers as published by VWS*

General observations

General observations of the limited initial pen tests

- The developers of the applications generally have not applied "secure coding" principles, as a result of which well-known and therefore expected security measures have not been implemented. This also applies to the underlying infrastructure (misconfigurations, outdated software, exposed management interfaces, etc.). This creates (serious) vulnerabilities that could easily have been prevented.

- In addition to the above, we would like to specifically point out that apps use "hard-coded" passwords, which are therefore legibly included in the legible source code. It turned out that these passwords enabled access to databases and / or the underlying infrastructure (including datasets outside the domain of the COVID application to be tested).

- The underlying infrastructure (including the API) can often be exploited without the necessary identification / authentication. Where a security has been applied, it is often easy to circumvent due to a vulnerable implementation, or the failure or validation of, for example, security certificates, or the use of self-signed certificates. This allows frequent access to confidential data and / or incorrect data can be entered.

- Applications are not always based on the principle of storing as little sensitive data as possible on the end user's phone. Data that is stored is also stored without encryption.

- Due to the desired functionality (track / trace), Android phones require a relatively large number of access permissions, which also include some seemingly strange requests, such as access to the microphone or photos. The desired permissions can be explained from the desired functionality, but the end user is likely to distrust and not accept this.

General observations - continued

General observations of the quick scan source code study

- The supplied software is quite diverse in the degree to which they are "ready for use": however, none of the systems are "finished". This also prevents a good comparative assessment of the findings; for example, where a moderate hashing algorithm is used in one application component, this functionality is lacking in other solutions as a whole.

- Technical documentation of the components has been largely absent. The source code researchers spent a lot of time figuring out how the components were put together, (if necessary) compiling and working.

- General measures to promote code quality have been found to a limited extent. We saw deviations from coding standards and "magic literals" (an anti-pattern where numbers and strings are used directly in the code). Furthermore, few unit tests and limited inline documentation were found in the source code.

Detail findings per research question

Research question 1

1. Can a malicious person acquire sensitive information by being able to communicate unauthorized with the underlying infrastructure of application X?

Supplier	Test
1	✓
2	✓
3	-
4	-
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- The developers of the applications generally have not applied "secure coding" principles, as a result of which well-known and therefore expected security measures have not been implemented. This also applies to the underlying infrastructure (misconfigurations, outdated software, exposed management interfaces, etc.). This creates (serious) vulnerabilities that could easily have been prevented.
- The use of "hard-coded" passwords that are therefore legibly included in the legible source code. This allowed access to sensitive data.
- The underlying infrastructure (including the API) can often be misused without the necessary identification / authentication. Where a security has been applied, it is often easy to circumvent due to a vulnerable implementation, or the failure or validation of, for example, security certificates, or the use of self-signed certificates. This allows frequent access to confidential data and / or incorrect data can be entered.
- A malicious party can manipulate the underlying infrastructure because there is no or very limited control of the data that is sent to / from this infrastructure.

Research question 2

2. Can a malicious person compromise or severely disrupt the underlying infrastructure of application X by injecting malicious code? As a result, the availability, integrity and confidentiality of the system and data cannot be guaranteed.

Supplier	Test
1	X
2	✓
3	✓
4	-
5	-
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- The underlying infrastructure contains various (serious) vulnerabilities that can be exploited, giving an attacker access to the data (read and / or manipulation) and also possible access to the underlying operating system. These include:
  - Insecure configuration of operating systems, databases and (application) services
  - Outdated software
  - Management interfaces exposed to the internet (admin portals, databases, etc.)
- A malicious party can also manipulate the underlying infrastructure because there is no or very limited control of the data that is sent to / from this infrastructure.

Research question 3

3. Can a malicious person enter incorrect data or delete correct data through misuse of the underlying infrastructure of application X? As a result, the data in the underlying infrastructure becomes less reliable and it is no longer possible to properly determine which users have COVID19 infection and which do not.

Supplier	Test
1	✓
2	✓
3	-
4	-
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

- Not / incomplete tested, missing information

- The underlying infrastructure (including the API) can often be misused without the necessary identification / authentication. Where a security has been applied, it is often easy to circumvent due to a vulnerable implementation, or the incorrect or incorrect validation of, for example, security certificates. This allows access to confidential data and / or incorrect data can be entered.

- The developers of the applications generally have not applied "secure coding" principles, as a result of which well-known and therefore expected security measures have not been implemented. This also applies to the underlying infrastructure (misconfigurations, outdated software, exposed management interfaces, etc.). This creates (serious) vulnerabilities that could easily have been prevented.

Research question 4

4. Can a malicious party intercept and / or misuse communication between application X and the underlying infrastructure (the channel itself or, for example, via the API)? This scenario has a higher chance of using public hotspots such as guest Wi-Fi and / or public hotspots.

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	X
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- Application and the underlying infrastructure do no or limited checks on identification and authentication of the sender / receiver (not even in the case where certificates are used because so-called certificate pinning is not applied). This allows an attacker to easily access communication through a man-in-the-middle attack. These kinds of attacks are easy to set up at public hotspots.
- Weak encryption is used in the communication between the app and the underlying infrastructure.

Research question 5

5. Can a malicious person obtain sensitive data from the storage space of application X on the end user's mobile device? This scenario can occur if the mobile device is vulnerable and application X stores sensitive data (consciously or unconsciously) on the device. This can affect the privacy of the user as well as possible other users (if data is also stored about this because these users were nearby, for example).

Supplier	Test
1	✓
2	✓
3	-
4	✓
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

- Not / incomplete tested, missing information

- Applications are not always based on the principle of storing as little sensitive data as possible on the end user's phone. Data that is stored is also stored without encryption.
- Also, not all applications are designed / implemented with an adequate password policy.

Research question 6

6. Can malicious communication between application X and the underlying infrastructure compromise through misuse of channels other than the primary channel (the expected API interface)? Think of additional channels such as e-mail, SMS or other TCP / UDP network interfaces.

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	-
6	X

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- In a few cases, the applications and the underlying infrastructure use additional channels (in addition to the primary channel, the API). By using these additional channels, there is a risk of additional vulnerabilities as well as the risk that another party may correlate the data to groups or even individuals. For example, by tracing identification data of the telephone (SMS, Bluetooth, UUID, telephone numbers, etc.) or other traceable / correlable information.

Research question 7

7. What is the general picture of the (technical) quality of the source code of the application?

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- The applications are generally small and have only limited technical debt (the expected effort required to implement a solution to the findings).
- With one exception, we were able to analyze the delivered application components with tools, which meant that the researchers could also compile the source code where necessary.
- Few measures have been found that promote code quality. As a result, quite a few deviations, including magic literals, from the coding standards have been found. Also, limited use is made of unit testing. In almost all solutions, the inline documentation found “Todo's” and “Fixme's”, which indicate that the solution is not finished. The difference in technical quality between prototypes and almost mature implementations is clearly noticeable.

Research question 8

8. Do automatic tooling in the “car wash” reveal any real vulnerabilities regarding the reliability and security in the source code?

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- Vulnerabilities have been identified in all solutions that require attention. The vulnerabilities found in the source code have been documented and reported to the suppliers.
- We note that in a number of cases no (or limited) authentication was implemented in the communication with the back-end application and that in some cases other security measures were missing in the source code.

Research question 9

9. Are there other data outputs (including logging) in the source code that are not defined in the design?

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- No data outputs were found that were not expected.
- There are, in particular, location data available in the apps, and sometimes even passed on to the back-end system, for which nothing is done (yet); this can damage the trust of users in the app. In addition, in theory it is possible that the supplier will process this data in the future without the user being informed - after all, the permission to make the data accessible from the app is already given at the initial installation.
- A point of attention is that in some solutions the error handling is not properly implemented; there are therefore risks that error messages with sensitive data may be displayed on screens, for example.

Research question 10

10. Does the supplied software (both front-end and back-end) depend on external libraries for proper operation? If so, are these libraries current, are they maintained and / or do they contain known vulnerabilities with regard to reliability and security?

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- Most solutions use frameworks and libraries. A single solution uses the latest versions and automated methods (dependency managers) to keep it that way. Other solutions use outdated versions; vulnerabilities are also known in a number of them.

Research question 11

11. Is the source code set up in line with our expectations from the technical and functional documentation? (For example, have the described privacy mechanisms been implemented, does the scope fit the description, etc.)?

Supplier	Test
1	✓
2	✓
3	✓
4	✓
5	✓
6	✓

✓ Tested

X Not / incomplete tested, time limit

\- Not / incomplete tested, missing information

- In general, very limited documentation has been found; in general, the rough functionality has been found as understood from the pitches. However, some of the solutions have not yet implemented all the listed functionality.
- The expected roll-out of the API for exchanging Bluetooth identifiers by Apple and Google is in some cases the reason for the (still) lack of functionality.
- With the exception of one solution, which works with a telephone number, the user's data is intended to be passed on anonymously. Specific privacy mechanisms have not been found.

Appendices

1. Scope
2. Approach

Scope

as described in our proposal dated April 17 with reference number A2000020142

Scope

Scope of the limited initial penetration test

For the limited initial penetration test, we use a scenario-based approach in which we investigate vulnerabilities based on the OWASP Mobile Top 10 (version 2016) and the OWASP Top 10 (version 2017).
The scope of the initial penetration test consists of the following three components:

- COVID-19 app. In doing so, we will investigate risks 'M2: Insecure Data Storage', and 'M5: Insufficient Cryptography', to determine to what extent sensitive information is stored in an unsafe manner.

- Communication between app and backend. We will investigate risks such as 'M3: Insecure Communication' and 'M5: Insufficient Cryptography'. We focus on the scenario that an attacker can access the end-user's local network, access the end-user's mobile device, or act as Man-in-The-Middle.

- Backend interface. We will investigate risks such as 'M10: Extraneous Functionality', 'A1: Injection', and 'A3: Sensitive Data Exposure'. We focus on unauthorized unlocking of data from the backend, unauthorized modification of data in the backend, and making the backend unavailable (with the exception of Distributed Denial-of-Service (DDoS) attacks).

Scope of the quick scan source code study

The quick scan source code investigation will focus on the source code supplied by the supplier and other artifacts including documentation. In view of the desired timeframe, no inquiries will be made about apparently missing elements; these elements will be reported as a detailed finding. The quality of underlying software components, such as operating systems, database and web server software, is not part of the scope.

Approach

as described in our proposal dated April 17 with reference number A2000020142

Approach

We carry one ***limited initial penetration test*** off, with a maximum given lead time of 24 hours, on the COVID-19 app, the associated backend and the communication between the app and the backend according to the "white box" principle. This means, among other things, that we obtain extensive access to documentation, system configuration, and other requested information in advance. We will not conduct detailed system configuration reviews, but will use this information to efficiently perform our tests.

At the same time, we get a picture of what an attacker could possibly do if this information is not available to him. Based on this information, we determine which scenarios are most relevant to test during the penetration test. Based on these scenarios, we go through the following three phases of our penetration test:

1. *Identification scan phase:* perform different identification scans on the backend environment. With this step we obtain detailed information about which ports and services are active.

2. *Vulnerability scan:* we use efficient tools that identify known vulnerabilities in systems. These tools and scans are applied to the backend environment. It also manually scans for any vulnerabilities, both at infrastructure and application level, that deal with the app, the backend and the communication between the app and the backend.

3. *Exploitation:* we perform manual checks to determine whether the vulnerabilities obtained in the previous steps are actually present (the so-called "false positive" verification). We also manually test for vulnerabilities and exploit them where possible. In this way we can determine the impact of the vulnerabilities. In the ***quick scan source code research,*** with a maximum given lead time of 24 hours, with emphasis on Reliability, we will inspect the delivered source code of both the front and (if applicable) backend application. Using automatic tooling we will determine commonly used software metrics such as the number of lines of code (LOC), Complexity and Duplication. From available tooling for the specific development platform, we will, if possible, obtain findings on Reliability, Security and Maintainability. From the focus of this research we will focus on the findings *Reliability* and *Security* manual override.

We will also check whether the software depends on external libraries, whether these libraries are current and whether there are any known reliability and security findings for these libraries.

Finally, we will check if the source code is in line with our expectations based on the technical and functional documentation and if there are data outputs or interfaces (including log files) other than those specified. We will pay attention to whether the described privacy mechanisms have been implemented. If cryptographic algorithms are used (within the supplied software), it will be checked whether these algorithms are frequently used and tested.

Ing. JAM Hermans RE

Partner
KPMG Advisory NV
Tel: + 31 20 656 8394 Mob: +
31 6 51 366 389
hermans.john@kpmg.nl

ir. R. Heil MSC CISSP GICSP CISA

Partner KPMG Advisory
NV
Tel: +31 20 656 8033 Mob:
+31 6 51 369 785
heil.ronald@kpmg.nl

Drs. JMA Koedijk CISA CISM

Partner
KPMG Advisory NV
Tel: + 31 20 656 8251 Mob: +
31 6 22 903 688
koedijk.joost@kpmg.nl

KPMG on social media

KPMG app