SPARTA

D5.4 Appendix I

ATE – Tests – Vertical 2 – Mobile Scenario

Project number	830892
Project acronym	SPARTA
Project title	Strategic programs for advanced research and technology in Europe
Start date of the project	1st February, 2019
Duration	36 months
Programme	H2020-SU-ICT-2018-2020

Deliverable type	Report
Deliverable reference number	SU-ICT-03-830892 / D5.4 / V1.0 / Appendix G
Work package contributing to the deliverable	WP5
Due date	Jan 2022 – M36
Actual submission date	2nd February, 2022

Responsible organisation	CINI
Editor	Luca Verderame
Dissemination level	PU
Revision	V1.0

Abstract	This document provides a description of the test procedure and report for the mobile Scenario (also known as Scenario for the CIE ID APP) of the "Complex System Assessment Including Large Software and Open-Source Environments, targeting e-Government Services" vertical (also known as e-Government services vertical or Vertical 2). The document demonstrates how the CAPE tools APPROVER and TSOpen contribute to secure the CIE ID APP scenario.
Keywords	Vulnerability Assessment, Mobile Security, CIE ID

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 830892.

Editor

Luca Verderame (CINI)

Contributors

Roberto Carbone, Andrea Bisegna (CINI)

Mirko Malacario, Claudio Porretti (LEO)

Jordan Samhi, Jacques Klein (UNILU)

Reviewers

Maximilian Tschirschnitz (TUM)

Rimantas Zylius (L3CE)

Table of Content

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Document Overview

This document provides a description of the test procedure and report for the mobile Scenario (also known as Scenario for the CIE ID APP) of the “Complex System Assessment Including Large Software and Open-Source Environments, targeting e-Government Services” vertical (also known as e-Government services vertical or Vertical 2).
We will show how the CAPE tools APPROVER and TSOpen contribute to secure the CIE ID APP scenario. In particular, we will show how:

- we properly integrated in the development process of the CIE ID APP the continuous integration techniques developed in the context of Task 5.3 for Approver and TSOpen, and
- APPROVER and TSOpen perform a security assessment of the CIE ID APP, providing a security report to the security analyst.

The structure of the document is organized as follows:

- **Chapter 1 Introduction**, is the current section presenting the objectives, scope and structure of the document.
- **Chapter 2 Test preparations,** presents the hardware and software used for testing.
- **Chapter 3 Test descriptions**, details the different test cases to be executed and their results.
- **Chapter 4 Test Summary Coverage,** shows the completeness of tests coverage.

Chapter 2 Test preparations

2.1 System overview

The e-Government services vertical (Vertical 2) has been fully described in D5.2 [1]. In this section, we provide an overview of the case study description, focusing on CIE ID APP scenario.

The demonstration scenario of the vertical 2 involves the development and testing environments managed by FBK (one of the institutions of the SPARTA partner CINI), where the preliminary versions of the CIE ID mobile app is developed and tested, before being migrated on the Italian Ministry of the Interior servers.

CIE ID is the app developed by the Italian National Mint and Printing House for accessing the services of the Italian Public Administrations, by leveraging the electronic identity card (CIE 3.0).

As mentioned in D5.3 [2], the app has been developed in Kotlin programming language, and it required Android 6.0 and later (API level > 23) equipped with NFC interface. The app is available in the official Google Play Store and currently has more than 100.000 installations. In the context of the SPARTA project, CINI has extended the Gitlab environment in such a way to use the continuous integration functionalities offered by Gitlab to automatically build the APK file after each commit in the repository.

The system under test used for SPARTA consists of the source code of the CIE ID mobile app and its integration in the DevSecOps pipeline.

To avoid any risk to disclose sensitive information concerning the official version of the CIE ID App, the tests are performed on an old version of the source code of the app. Indeed, the purpose of the tests is to show that the CAPE tools are properly integrated in the development process and are indeed helpful to spot relevant vulnerabilities.

2.1.1 Hardware preparation

APPROVER and TSOpen are provided as SaaS so there is no needed hardware preparation for using the tool. The VM used to host the tools is an Azure Standard DS3 v2 equipped with a processor Intel Xeon E5-2673 v4 2.29 GHz 4 Cores, 14 GB of RAM and two drives, one hard disk drive of 30 GB used by the OS and 10GB of solid-state drive used by the tools.

2.1.2 Software preparation

The testing environment consists of a Gitlab platform hosted by FBK, and cloud-hosted Azure virtual machines controlled by FBK.
Gitlab provides:

- a version control system (Git-repository), storing the source code of CIE ID APP;
- Issues tracking and continuous integration and deployment pipeline.

The Azure virtual machines, running Linux distributions (Ubuntu 20.04) and supporting the Docker technology, are used to run the APPROVER and TSOpen tools.
In the virtual machines hosting the tools, we installed the GitLab Runner application, which works with GitLab CI/CD to run jobs in a pipeline.

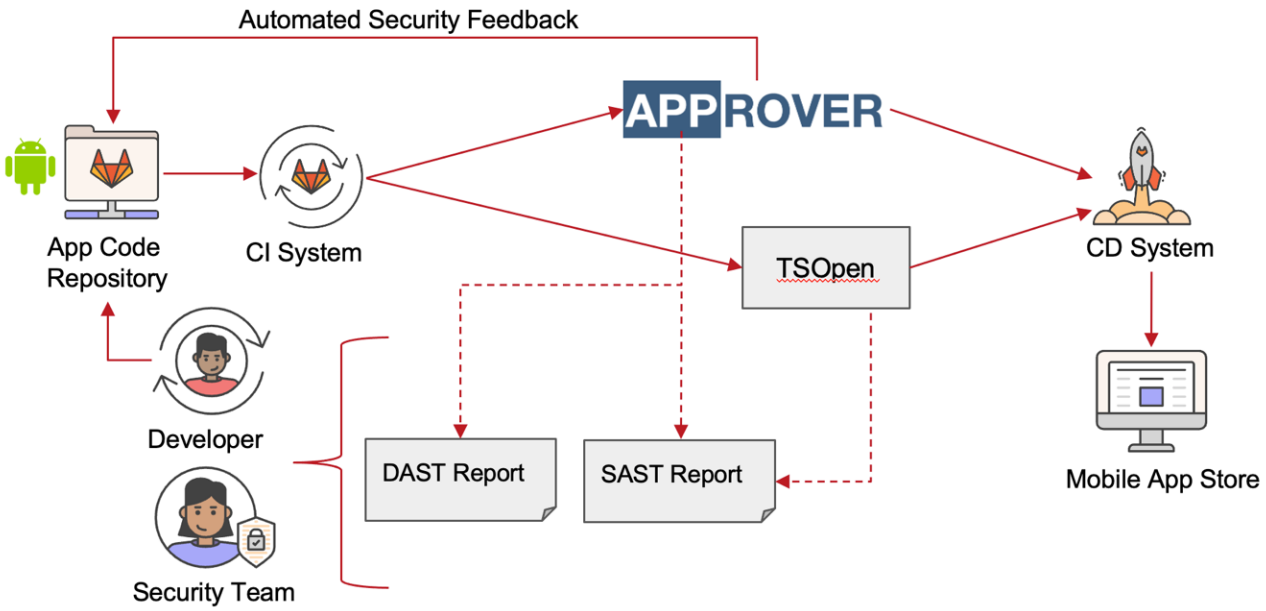

Figure 1: Vertical 2 - Mobile Scenario

As mentioned in D5.3, to assess the security of the CIE ID APP, we deployed the DevSecOps scenario depicted in Figure 1. The APPROVER and TSOpen CAPE tools are used to evaluate the security and risk requirements for the CIE ID mobile app.
The pipeline has been designed as a set of integration scripts that are attached to the GitLab repository hosting the mobile app source code. The tools are either hosted in the Azure VM environment as remote services (SaaS) exposing REST APIs.

Chapter 3 Test descriptions

Table 1 shows the Security Requirements (SRs) for the CIE ID Android Apps. In the next sections we describe the test descriptions that have been elaborated to support the test of these requirements.

Security Req (ID)	Short Description
MSTG-STORAGE-2	No sensitive data should be stored outside of the CIE ID-App container or system credential storage facilities.
MSTG-STORAGE-3	No sensitive data is written to CIE ID-App logs.
MSTG-STORAGE-4	No sensitive data is shared with third parties unless it is a necessary part of the architecture.
MSTG-STORAGE-8	No sensitive data is included in backups generated by the mobile operating system.
MSTG-STORAGE-9	The CIE ID-App removes sensitive data from views when moved to the background.
MSTG-CRYPTO-1	The CIE ID-App does not rely on symmetric cryptography with hardcoded keys as a sole method of encryption.
MSTG-CRYPTO-2	The CIE ID-App uses proven implementations of cryptographic primitives.
MSTG-CRYPTO-3	The CIE ID-App uses cryptographic primitives that are appropriate for the particular use-case, configured with parameters that adhere to industry best practices.
MSTG-CRYPTO-4	Taking into account the constraints posed by the cryptographic algorithms supported by the card (CIE), the CIE ID-App does not use cryptographic protocols or algorithms that are widely considered deprecated for security purposes.
MSTG-CRYPTO-6	All random values used in the CIE ID-App are generated using a sufficiently secure random number generator.
MSTG-NETWORK-1	Data is encrypted on the network using TLS. The secure channel is used consistently throughout the CIE ID-App.
MSTG-NETWORK-2	The TLS settings are in line with current best practices, or as close as possible if the mobile operating system does not support the recommended standards.
MSTG-NETWORK-3	The CIE ID-App verifies the X.509 certificate of the CIE ID-Server when the secure channel is established. Only certificates signed by a trusted CA are accepted.
MSTG-NETWORK-4	The CIE ID-App pins the CIE ID-Server certificate or public key, and subsequently does not establish connections with endpoints that offer a different certificate or key, even if signed by a trusted CA.
MSTG-NETWORK-6	The CIE ID-App only depends on up-to-date connectivity and security libraries.
MSTG-PLATFORM-1	The CIE ID-App only requests the minimum set of permissions necessary.
MSTG-PLATFORM-4	The CIE ID-App does not export sensitive functionality through IPC facilities, unless these mechanisms are properly protected.

Security Req (ID)	Short Description
MSTG-PLATFORM-5	JavaScript is disabled in WebViews unless explicitly required.
MSTG-PLATFORM-6	WebViews are configured to allow only the minimum set of protocol handlers required (ideally, only https is supported). Potentially dangerous handlers, such as file, tel and app-id, are disabled.
MSTG-PLATFORM-10	A WebView's cache, storage, and loaded resources (JavaScript, etc.) should be cleared before the WebView is destroyed.
MSTG-CODE-1	The CIE ID-App is signed and provisioned with a valid certificate, of which the private key is properly protected.
MSTG-CODE-2	The CIE ID-App has been built in release mode, with settings appropriate for a release build (e.g., non-debuggable).
MSTG-CODE-3	Debugging symbols have been removed from native binaries.
MSTG-CODE-4	Debugging code and developer assistance code (e.g. test code, backdoors, hidden settings) have been removed. The CIE ID-App does not log verbose errors or debugging messages.
MSTG-RESILIENCE-1	The CIE ID-App detects, and responds to, the presence of a rooted or jailbroken device either by alerting the user or terminating the app.
MSTG-RESILIENCE-9	Obfuscation is applied to programmatic defenses, which in turn impede de-obfuscation via dynamic analysis.
TSOpen-1	The CIE ID-App's specific values are symbolically executed
TSOpen-2	The CIE ID-App's suspicious checks are identified
TSOpen-3	Suspicious checks' behavior is scanned to check for malicious behavior
TSOpen-4	Based on control dependency, logic bombs are identified
TSOpen-5	Apps are checked against existing logic bombs that trigger malicious code under specific circumstances, bypassing detection techniques.

Table 1: Security Requirements covered by Approver and TSOpen

3.1 MSTG-STORAGE-2_TC1

Test case to validate if CIE ID-App stores sensitive data and, if yes, verifies if it is securely stored. The MSTG-STORAGE-2 focuses on identifying potentially sensitive data stored by an application and verifying if it is securely stored. In general, sensitive data stored locally on the device should always be at least encrypted, and any keys used for encryption methods should be securely stored within the Android Keystore. These files should also be stored within the application sandbox. If achievable for the application, sensitive data should be stored off device or, even better, not stored at all.

3.1.1 Security Requirements addressed

MSTG-STORAGE-2

3.1.2 Test preconditions

Apk file of CIE ID App uploaded on APPROVER.

3.1.3 Expected test results

The app does not store sensitive data or securely store it.

3.1.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test pass if there is no entry in the Vulnerability Analysis section called “External Storage Accessing”, no entry in the Policy Checker section called “c M2 - UNDESIRED ACCESS TO PRIVATE DATA”, and no one entry in the Privacy Leaks section.

3.1.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis.
    - Policy Checker.
    - Privacy Leaks.

3.1.6 Test Results

The policy checker module sets as not satisfied the following requirements "OWASP M2 - UNDESIRED ACCESS TO PRIVATE DATA".

Status: *FAILED*

Notes: the security issue is not applicable to the CIE ID app. The test failed since the testing device does not have storage encryption activated.

3.2 MSTG-STORAGE-3_TC1

Test case to validate if CIE ID App writes sensitive data on application logs.

3.2.1 Security Requirements addressed

MSTG-STORAGE-3

3.2.2 Test preconditions

Apk file of CIE ID uploaded on APPROVER.

3.2.3 Expected test results

No sensitive data are written in application log.

3.2.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.

The test is passed if the Privacy Leaks module does not contain any entry regarding the application logs

3.2.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Privacy Leaks.

3.2.6 Test Results

The Privacy Leaks module does not contain any entry regarding the application logs.
Status: *PASSED*

3.3 MSTG-STORAGE-4_TC1

Test case to validate if CIE ID App shares sensitive data with third parties unless it is a necessary part of the architecture.

3.3.1 Security Requirements addressed

MSTG-STORAGE-4

3.3.2 Test preconditions

Apk file of CIE ID App.

3.3.3 Expected test results

No sensitive data are shared with third parties unless it is a necessary part of the architecture.

3.3.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test is passed if there is no entry in the Privacy Leaks module refers to the network traffic

3.3.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.

- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Privacy Leaks

3.3.6 Test Results

There is no entry in the Privacy Leaks module because the app does not share any sensitive data with third parties.
Status: *PASSED*

3.4 MSTG-STORAGE-8_TC1

Test case to validate if CIE ID App stores sensitive data in backup generated by the mobile operating system.

3.4.1 Security Requirements addressed

MSTG-STORAGE-8

3.4.2 Test preconditions

Apk file of CIE ID App.

3.4.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
No sensitive data are inserted in backup generated by the mobile operating system.

3.4.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is no entry in the Privacy Leaks regarding "Insecure Backup".

3.4.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.

- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Privacy Leaks

3.4.6 Test Results

There is no entry in the Privacy Leaks module because the app does not include any sensitive data in the backup.
Status: *PASSED*

3.5 MSTG-STORAGE-9_TC1

Test case to validate if CIE ID App removes sensitive data from view when moved app to the background.

3.5.1 Security Requirements addressed

MSTG-STORAGE-9

3.5.2 Test preconditions

Apk file of CIE ID App.

3.5.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The sensitive data should be removed from view when the app is moved to the background.

3.5.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test pass if there is no entry in the Privacy Leaks regarding "Insecure Data View".

3.5.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Privacy Leaks

3.5.6 Test Results

There is no entry in the Privacy Leaks module because the app does not include any sensitive data in the backup.
Status: *PASSED*

3.6 MSTG-CRYPTO-1_TC1

Test case to validate if CIE ID App relies on symmetric cryptography with hardcoded keys as a sole method of encryption.

3.6.1 Security Requirements addressed

MSTG-CRYPTO-1

3.6.2 Test preconditions

Apk file of CIE ID App.

3.6.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses different cryptography keys not hardcoded as the encryption method.

3.6.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is not *crypto_constant_key* entry in the Vulnerability Analysis module.

3.6.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.6.6 Test Results

There is no entry called "*crypto_constant_ke*y " in the Vulnerability Analysis section because the app does not use an hardcoded key as a sole encryption method.
Status: *PASSED*

3.7 MSTG-CRYPTO-2_TC1

Test case to validate if CIE ID App uses proven implementations of cryptographic primitives.

3.7.1 Security Requirements addressed

MSTG-CRYPTO-2

3.7.2 Test preconditions

Apk file of CIE ID App.

3.7.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses use a proven implementation of cryptographic primitives.

3.7.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is any of the following keys *crypto_constant_salt*, *crypto_ecb_cipher*, *crypto_small_iteration_count*, *crypto_constant_key,* *crypto_constant_iv* or *crypto_keystore_entry_without_password* in the Vulnerability Analysis results.

3.7.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.7.6 Test Results

The app uses sole proven cryptographic primitives implementations.
Status: *PASSED*

3.8 MSTG-CRYPTO-3_TC1

Test case to validate if CIE ID App uses cryptographic primitives that are appropriate for the particular use-case, configured with parameters that adhere to industry best practices.

3.8.1 Security Requirements addressed

MSTG-CRYPTO-3

3.8.2 Test preconditions

Apk file of CIE ID App.

3.8.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses use only appropriate cryptographic primitives.

3.8.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there are no entry such as *crypto_constant_salt*, *crypto_ecb_cipher*, *crypto_small_iteration_count*, *crypto_constant_key, crypto_constant_iv* or *crypto_keystore_entry_without_password* in the Vulnerability Analysis results.

3.8.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.8.6 Test Results

The app uses sole appropriate cryptographic implementations.
Status: *PASSED*

3.9 MSTG-CRYPTO-4_TC1

Test case to validate if CIE ID App does not use cryptographic protocols or algorithms that are widely considered deprecated for security purposes, taking into account the constraints posed by the cryptographic algorithms supported by the card (CIE).

3.9.1 Security Requirements addressed

MSTG-CRYPTO-4

3.9.2 Test preconditions

Apk file of CIE ID App.

3.9.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses use only appropriate cryptographic primitives

3.9.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there are no entry such as *crypto_constant_salt*, *crypto_ecb_cipher*, *crypto_small_iteration_count*, *crypto_constant_key, crypto_constant_iv* or *crypto_keystore_entry_without_password* in the Vulnerability Analysis results.

3.9.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.9.6 Test Results

The app uses sole appropriate cryptographic implementations.
Status: *PASSED*

3.10 MSTG-CRYPTO-6_TC1

Test case to validate if CIE ID App generates random values using a sufficiently secure random number generator.

3.10.1 Security Requirements addressed

MSTG-CRYPTO-6

3.10.2 Test preconditions

Apk file of CIE ID App.

3.10.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses generate random values using a sufficiently secure random number generator.

3.10.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is no *crypto_constant_iv* in the Vulnerability Analysis results.

3.10.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.

- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.10.6 Test Results

The app uses generate random values using a sufficiently secure random number generator.
Status: *PASSED*

3.11 MSTG-NETWORK-1_TC1

Test case to validate if CIE ID App encrypts data on the network using TLS and consistently uses the secure channel throughout the CIE ID App.

3.11.1 Security Requirements addressed

MSTG-NETWORK-1

3.11.2 Test preconditions

Apk file of CIE ID App.

3.11.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses TLS to communicate with the app backend.

3.11.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test pass if the app uses only HTTPS to communicate with other services through the network.

3.11.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.

- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Network Analysis

3.11.6 Test Results

The app uses HTTPS to communicate with the other services.
Status: *PASSED*

3.12 MSTG-NETWORK-2_TC1

Test case to validate if the TLS settings are in line with current best practices, or as close as possible if the mobile operating system does not support the recommended standards.

3.12.1 Security Requirements addressed

MSTG-NETWORK-2

3.12.2 Test preconditions

Apk file of CIE ID App.

3.12.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses TLS with best practice to communicate using the network.

3.12.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test pass if the app uses only HTTPS with all best practices to communicate with the other services through the network and no one entry such as NO PINNING, ALL CERTIFICATE, or HTTP is found on the Network Analysis module.

3.12.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Network Analysis

3.12.6 Test Results

The CIE ID App performs HTTPS requests without using the SSL Pinning Techniques.

Status: *FAILED*

Note: the URLs accessed are not referred to the backend of the app; thus, the vulnerability is not applicable.

3.13 MSTG-NETWORK-3_TC1

Test case to validate if CIE ID App verifies the X.509 certificate of the CIE ID-Server when the secure channel is established. Only certificates signed by a trusted CA are accepted.

3.13.1 Security Requirements addressed

MSTG-NETWORK-3

3.13.2 Test preconditions

Apk file of CIE ID App.

3.13.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app accepts only certificate signed by a trusted CA.

3.13.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test pass if there are no entry called ALL CERTIFICATE in the Network Analysis module and “insecure_hostname_verifier” in the Vulnerability Analysis module.

3.13.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Network Analysis
    - Vulnerability Analysis

3.13.6 Test Results

The app accepts only certificates signed by a trusted CA.
Status: *PASSED*

3.14 MSTG-NETWORK-4_TC1

Test case to validate if CIE ID App pins the CIE ID-Server certificate or public key, and subsequently does not establish connections with endpoints that offer a different certificate or key, even if signed by a trusted CA.

3.14.1 Security Requirements addressed

MSTG-NETWORK-4

3.14.2 Test preconditions

Apk file of CIE ID App.

3.14.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app accepts only the CIE ID-Server certificate.

3.14.4 Criteria for evaluating results

The duration is on average 45 minutes. However, the duration of dynamic analysis depends on the number of apps in the APPROVER analysis queue.
The test pass if there are no entry called NO PINNING in the Network Analysis module and "*Invalid_server_certificate*" in the Vulnerability Analysis module

3.14.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Network Analysis

3.14.6 Test Results

The CIE ID App performs HTTPS requests without using the SSL Pinning Techniques.

Status: *FAILED*

Note: the URLs accessed are not referred to the backend of the app; thus, the vulnerability is not applicable.

3.15 MSTG-NETWORK-6_TC1

Test case to validate if CIE ID App depends only on up-to-date connectivity and security libraries.

3.15.1 Security Requirements addressed

MSTG-NETWORK-6

3.15.2 Test preconditions

Apk file of CIE ID App.

3.15.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app uses only up-to-date connectivity and security libraries.

3.15.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is no entry called *insecure_socket_factory* in the Vulnerability Analysis results.

3.15.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.15.6 Test Results

The app uses only up-to-date connectivity and security libraries.
Status: *PASSED*

3.16 MSTG-PLATFORM-1_TC1

Test case to validate if CIE ID App requests only the minimum set of permissions necessary.

3.16.1 Security Requirements addressed

MSTG-PLATFORM-1

3.16.2 Test preconditions

Apk file of CIE ID App.

3.16.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app use only the minimum set of permissions necessary.

3.16.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there are no permissions declared but not used in the Permission Analysis results.

3.16.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Permissions Analysis

3.16.6 Test Results

The app declared several permissions, but they aren’t used.
Status: *FAILED*

Note: the permissions are required for the inclusion of external libraries; thus, the vulnerability is not applicable.

3.17 MSTG-PLATFORM-4_TC1

Test case to validate if CIE ID App does not export sensitive functionality through IPC facilities, unless these mechanisms are properly protected.

3.17.1 Security Requirements addressed

MSTG-PLATFORM-4

3.17.2 Test preconditions

Apk file of CIE ID App.

3.17.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app does not export sensitive functionality through IPC facilities, unless these mechanisms are properly protected.

3.17.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there are no entry such exported_content_provider, exported_without_prefix, implicit_intent_service, or intent filter misconfiguration in the Vulnerability Analysis results

3.17.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.17.6 Test Results

- Found "exported" components (except for Launcher) for receiving outside applications' actions (AndroidManifest.xml). These components can be initialized by other apps and used maliciously.
- The application contains implicit Intents for starting Services. Using an implicit Intent to start a service is a security hazard because you cannot be certain of what service will respond to the intent, and the user cannot see which service starts.

Status: *FAILED*

Note: the exported components are required for the inclusion of external libraries (Google Libraries); thus, the vulnerability is not applicable.

3.18 MSTG-PLATFORM-5_TC1

Test case to validate if in the CIE ID App JavaScript is disabled in WebViews unless explicitly required.

3.18.1 Security Requirements addressed

MSTG-PLATFORM-5

3.18.2 Test preconditions

Apk file of CIE ID App.

3.18.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app disables JavaScript in WebView unless is not explicitly required.

3.18.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is no entry called *webview_javascript_enabled* in the Vulnerability Analysis results.

3.18.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.18.6 Test Results

- Found "setJavaScriptEnabled(true)" in WebView. Enabling JavaScript exposes to malicious injection of code that would be executed with the same permissions (XSS attacks).

Status: *FAILED*

Note: this functionality is required to load Javascript code inside the WebView; thus, the vulnerability is not applicable.

3.19 MSTG-PLATFORM-6_TC1

Test case to validate if WebViews are configured to allow only the minimum set of protocol handlers required (ideally, only https is supported). Potentially dangerous handlers, such as file, tel and app-id, are disabled.

3.19.1 Security Requirements addressed

MSTG-PLATFORM-6

3.19.2 Test preconditions

Apk file of CIE ID App.

3.19.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app should configure WebView in order to use only HTTPS protocol.

3.19.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if there is no entry such as webview_intercept_request or webview_allow_file_access in the Vulnerability Analysis results.

3.19.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.19.6 Test Results

The app loads different resources within the WebView. However, the method shouldInterceptRequest has not been overridden by developers.

Status: *FAILED*

Note: this functionality is required to load Javascript resoruces inside the WebView; thus, the vulnerability is not applicable.

3.20 MSTG-PLATFORM-10_TC1

Test case to validate if CIE ID App clears WebView's cache, storage, and loaded resources (JavaScript, etc.) before the WebView is destroyed.

3.20.1 Security Requirements addressed

MSTG-PLATFORM-10

3.20.2 Test preconditions

Apk file of CIE ID App.

3.20.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app clears WebView’s cache, storage and loaded resources before the WebView is destroyed.

3.20.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.

The test pass if there is no webview_misconfiguration entry in the Vulnerability Analysis results.

3.20.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the following sections (on APPROVER web page or on GitLab issue tab):
    - Vulnerability Analysis

3.20.6 Test Results

The app has been tested using static analysis without any errors.

Status: *PASSED*

3.21 MSTG-CODE-1_TC1

Test case to validate if CIE ID App is signed and provisioned with a valid certificate, of which the private key is properly protected.

3.21.1 Security Requirements addressed

MSTG-CODE-1

3.21.2 Test preconditions

Apk file of CIE ID App.

3.21.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app is signed and provisioned with a valid certificate.

3.21.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if the app is installed on virtual environment.

3.21.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.

- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze if the app has been tested using dynamic analysis, evaluating if at least one of the following modules has been completed without any errors:
    - Network Analysis
    - Logcat Viewer
    - Privacy Leaks
    - API Invocation Monitor
    - File Analysis

3.21.6 Test Results

The app has been tested using dynamic analysis without any errors.

Status: *PASSED*

3.22 MSTG-CODE-2_TC1

Test case to validate if CIE ID App has been built in release mode, with settings appropriate for a release build (e.g. non-debuggable).

3.22.1 Security Requirements addressed

MSTG-CODE-2

3.22.2 Test preconditions

Apk file of CIE ID App.

3.22.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app has been built in release mode and the flag android.debuggable is set to false or it is not present on AndroidManifest.xml

3.22.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if the app is built using release mode and the debuggable option is disabled.

3.22.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.

- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the Vulnerability Analysis and check if Android *Debug Mode Checking* entry is present.

3.22.6 Test Results

DEBUG mode is ON (android:debuggable="true" in AndroidManifest.xml). Debug mode is extremely discouraged in production since malicious users can debug the app and sniff verbose error information through Logcat.

Status: *FAILED*

Note: the version of the app used for the test has been compiled in debug mode. The version available to the public has the debug mode disabled.

3.23 MSTG-CODE-3_TC1

Test case to validate if Debugging symbols have been removed from native binaries.

3.23.1 Security Requirements addressed

MSTG-CODE-3

3.23.2 Test preconditions

Apk file of CIE ID App.

3.23.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The developer removed debugging symbols from native binaries.

3.23.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if the debugging symbols have been removed from native binaries.

3.23.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.

- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze the Vulnerability Analysis and check if *debugging symbols* entry is present.

3.23.6 Test Results

The debugging symbols have been removed from native binaries.

Status: *PASSED*

3.24 MSTG-CODE-4_TC1

Test case to validate if Debugging code and developer assistance code (e.g., test code, backdoors, hidden settings) have been removed. The CIE ID-App does not log verbose errors or debugging messages.

3.24.1 Security Requirements addressed

MSTG-CODE-4

3.24.2 Test preconditions

Apk file of CIE ID App.

3.24.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app does not log verbose errors or debugging messages.

3.24.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if the debugging code and developer assistance code has been removed.

3.24.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the users should analyze
    - the Vulnerability Analysis and check if *debug code* entry is present.

- Logcat Viewer and check if debug logs or verbose errors are present in the application logs.

3.24.6 Test Results

The app does not log any debug information on the application log and all developer assistance code has been removed.
Status: *PASSED*

3.25 MSTG-RESILIENCE-1_TC1

Test case to validate if CIE ID App detects, and responds to, the presence of a rooted or jailbroken device either by alerting the user or terminating the app.

3.25.1 Security Requirements addressed

MSTG-RESILIENCE-1

3.25.2 Test preconditions

Apk file of CIE ID App.

3.25.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app performs a check of the environment and, if the device has root permissions, the app does not work.

3.25.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if the app is installed on a rooted device and works correctly.

3.25.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test the user should analyze the Dynamic Analysis phase and check if the app worked correctly.

3.25.6 Test Results

The app has been installed on a rooted device, but show only a fake page.

Status: *PASSED (False Negative)*

3.26 MSTG-RESILIENCE-9_TC1

Test case to validate if obfuscation is applied to programmatic defenses, which in turn impede de-obfuscation via dynamic analysis.

3.26.1 Security Requirements addressed

MSTG-RESILIENCE-9

3.26.2 Test preconditions

Apk file of CIE ID App.

3.26.3 Expected test results

The App is uploaded on APPROVER that decompiles the app in order to perform the static analysis and install the app in a virtual environment in order to perform the dynamic analysis.
The app code is obfuscated, and level of code obfuscation is high.

3.26.4 Criteria for evaluating results

The test duration is variable and ranges from a minimum of about 1 minute to a maximum of 45 minutes.
The test pass if the app code is obfuscated enough.

3.26.5 Test Procedure

The following steps will be carried out:

- Open a terminal inside the local repository of the CIE ID App.
- Create a new commit of the CIE ID App.
- Pushes the new version of the CIE ID App on GitLab.
- The pipeline defined through the GitLab-CI is started at each push, and a new version of the app (i.e., apk) is built starting from the source code.
- After the app building, the pipeline automatically uploads the apk file on APPROVER and waits for the results.
- At the end of the analysis, if APPROVER has found privacy or security issues, the GitLab-CI pipeline automatically opens new issues on GitLab repositories.
- To view the results, the user can log in on the APPROVER web page and see the results by clicking on the last analyzed app. Otherwise, the user can click on the issue tab on GitLab and see the current app's state.
- To evaluate the test, the user should analyze the App Information section and the value of Obfuscation.

3.26.6 Test Results

The actual percentage of app code obfuscation is about 4.5%

Status: *FAILED*

Note: the version of the app used for the test has been compiled in debug mode. The version available to the public has the obfuscation enabled.

3.27 TSOpen-1_TC1

Test case to check if values in app under test are symbolically evaluated.

3.27.1 Security Requirements addressed

TSOpen-1

3.27.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.
A time-related logic bomb is present in the app.

3.27.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen statically and symbolically models the time-related value that is used to trigger the logic bomb.

The time-related value used to trigger the logic bomb is identified and correctly modelled

3.27.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
This test case fails if the time-related value used to trigger the logic bomb is not modelled correctly.

3.27.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models the time-related value used to trigger the logic bomb
- A practitioner inspects TSOpen's output to check if the time-related value has been modelled

3.27.6 Test Results

The app contains a time-related values that is used to trigger a logic bomb:

Status: *PASSED*

3.27.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an "Out of memory" exception: -Xss1g -Xmx25g

3.28 TSOpen-1_TC2

Test case to check if values in app under test are symbolically evaluated.

3.28.1 Security Requirements addressed

TSOpen-1

3.28.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.

3.28.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen statically and symbolically models TIME-, LOCATION-, and SMS-related values in the app.

3.28.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
Check that time-, location-, and sms-related values are modelled

3.28.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models time-, location-, and sms-related values
- A practitioner inspects TSOpen's output to check if time-, location-, and sms-related values have been correctly modelled.

3.28.6 Test Results

Time-, location-, and sms-related values are modelled.

Status: *PASSED*

3.28.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an "Out of memory" exception: -Xss1g -Xmx25g

3.29 TSOpen-2_TC1

Test case to check if suspicious conditions in app under test are identified.

3.29.1 Security Requirements addressed

TSOpen-2

3.29.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.
A time-related logic bomb is present in the app.

3.29.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen statically identifies the suspicious condition used to trigger the time-related logic bomb based on the symbolic execution and the data propagation (i.e., time-related tag).

3.29.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
The condition used to trigger the logic bomb based on a time-related value is identified by TSOpen

3.29.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models the time-related value used to trigger the logic bomb
- TSOpen identifies the suspicious condition used to trigger the logic bomb
- A practitioner inspects TSOpen's output to check if the condition used to trigger the logic bomb has been identified as suspicious

.

3.29.6 Test Results

The app contains a suspicious condition based on a time-related value that is used to trigger a logic bomb:

Status: *PASSED*

3.29.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an “Out of memory” exception: -Xss1g -Xmx25g

3.30 TSOpen-2_TC2

Test case to check if suspicious conditions in app under test are identified.

3.30.1 Security Requirements addressed

TSOpen-2

3.30.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.

3.30.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen statically identifies the suspicious conditions used to trigger code based on the symbolic execution and the data propagation (i.e., time-, location-, and sms-related tags).

3.30.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
Check that some conditions are identified as suspicious

3.30.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models time-, location-, and sms-related values used to trigger code
- TSOpen identifies suspicious conditions used to trigger code
- A practitioner inspects TSOpen’s output to check if conditions used to trigger code have been identified as suspicious

3.30.6 Test Results

No bomb logic in the code, some conditions can be identified as suspicious based on TSOpen definition.

Status: *PASSED*

3.30.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an “Out of memory” exception: -Xss1g -Xmx25g

3.31 TSOpen-3_TC1

Test case to check if suspicious conditions' behaviour in app under test are scanned for malicious behaviour.

3.31.1 Security Requirements addressed

TSOpen-3

3.31.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.
A time-related logic bomb is present in the app.

3.31.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen statically scans the guarded code of the suspicious condition identified based on a time-related value for sensitive API usage and report it.

3.31.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
Check that sensitive behavior has been triggered and suspicious condition is identified by TSOpen.

3.31.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models the time-related value used to trigger the logic bomb
- TSOpen identifies the suspicious condition used to trigger the logic bomb
- TSOpen scans the guarded code of the suspicious condition for sensitive API usage
- A practitioner inspects TSOpen's output to check if the sensitive behavior triggered by the suspicious condition is identified by TSOpen

3.31.6 Test Results

The app contains sensitive behavior triggered by a suspicious condition based on a time-related value that is used to trigger a logic bomb:

Status: PASSED

3.31.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an "Out of memory" exception: -Xss1g -Xmx25g

3.32 TSOpen-3_TC2

Test case to check if suspicious conditions' behaviour in app under test are scanned for malicious behaviour.

3.32.1 Security Requirements addressed

TSOpen-3

3.32.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.

3.32.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen statically scans the guarded code of suspicious conditions identified based on time-, location-, and sms-related values for sensitive API usage and report it.

3.32.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
The app does not contain suspicious condition with sensitive API called in the guarded code

3.32.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models time-, location-, and sms-related values used to trigger code
- TSOpen identifies suspicious conditions used to trigger code
- TSOpen scans the guarded code of the suspicious conditions for sensitive API usage
- A practitioner inspects TSOpen's output to check if the sensitive behavior triggered by the suspicious conditions ares identified by TSOpen

3.32.6 Test Results

The app does not contain suspicious condition with sensitive API called in the guarded code,.

Status: *PASSED*

3.32.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an "Out of memory" exception: -Xss1g -Xmx25g

3.33 TSOpen-4_TC1

Test case to check if logic bombs are identified based on control dependency step.

3.33.1 Security Requirements addressed

TSOpen-4

3.33.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.
A time-related logic bomb is present in the app.

3.33.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen identifies given suspicious trigger identified in previous steps as logic bombs based on the control dependency step, i.e., the guarded code of a trigger contains at least a sensitive method.

3.33.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
Check that logic is identified as a potential logic bomb by TSOpen.

3.33.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated

- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models the time-related value used to trigger the logic bomb
- TSOpen identifies the suspicious condition used to trigger the logic bomb
- TSOpen scans the guarded code of the suspicious condition for sensitive API usage
- TSOpen identifies the suspicious conditions as well as its guarded code as a potential logic bomb.
- A practitioner inspects TSOpen's output to check if the potential logic bomb identified correspond to the logic bomb present in the app.

3.33.6 Test Results

The app contains a logic bomb based on a time-related value:

Status: *PASSED*

3.33.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an "Out of memory" exception: -Xss1g -Xmx25g

3.34 TSOpen-4_TC2

Test case to check if logic bombs are identified based on control dependency step.

3.34.1 Security Requirements addressed

TSOpen-4

3.34.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.

3.34.3 Expected test results

The APK file is given as input to TSOpen that analyses it.
TSOpen should not identify logic bomb.

3.34.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
Check that logic bombs are not found by TSOpen.

3.34.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models the time-, location-, and sms-related values used to trigger code
- TSOpen identifies the suspicious conditions used to trigger code
- TSOpen scans the guarded code of the suspicious conditions for sensitive API usage
- TSOpen does not identifiy the suspicious conditions as well as its guarded code as a potential logic bomb.
- A practitioner inspects TSOpen's output to check if no potential logic bombs are identified in the app.

3.34.6 Test Results

The app does not contain a logic bomb:

Status: PASSED

3.34.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an “Out of memory” exception: -Xss1g -Xmx25g

3.35 TSOpen-5_TC1

Test case to check if logic bombs that trigger malicious code under specific circumstances exist in app under test.

3.35.1 Security Requirements addressed

TSOpen-5

3.35.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.
A time-related logic bomb is present in the app.

3.35.3 Expected test results

The APK file is given as input to TSOpen that analyses it. TSOpen identifies the time-related logic bomb present in the app.

3.35.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour. This test case fails if the time-related logic bomb is not found.
Check that existing time-related logic bomb is found by TSOpen.

3.35.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models the time-related value used to trigger the logic bomb
- TSOpen identifies the suspicious condition used to trigger the logic bomb
- TSOpen scans the guarded code of the suspicious condition for sensitive API usage
- TSOpen identifies the suspicious conditions as well as its guarded code as a potential logic bomb.
- TSOpen reports the time-related logic bomb.
- A practitioner inspects TSOpen’s output to check if the logic bomb present in the app is reported by TSOpen

3.35.6 Test Results

The app contains a logic bomb based on a time-related value:

Status: *PASSED*

3.35.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an “Out of memory” exception: -Xss1g -Xmx25g

3.36 TSOpen-5_TC2

Test case to check if logic bombs that trigger malicious code under specific circumstances exist in app under test.

3.36.1 Security Requirements addressed

TSOpen-5

3.36.2 Test preconditions

Apk file of the CIE ID App is given as input to TSOpen.

3.36.3 Expected test results

The APK file is given as input to TSOpen that analyses it.
TSOpen does not identify any logic bomb in the app.

3.36.4 Criteria for evaluating results

The duration of TSOpen execution should not exceed one hour.
Check that no logic bombs have been detected by TSOpen

3.36.5 Test Procedure

The following steps will be carried out:

- CIE ID App is updated
- CIE ID App is built on the gitlab
- The pipeline automatically gives CIE ID App APK file as input to TSOpen
- TSOpen statically analyses the APK
- TSOpen models time-, location-, and sms-related values used to trigger code
- TSOpen identifies the suspicious condition used to trigger codes
- TSOpen scans the guarded code of the suspicious conditions for sensitive API usage
- TSOpen does not identify the suspicious conditions as well as its guarded code as a potential logic bomb.
- TSOpen does not report any logic bomb.
- A practitioner inspects TSOpen's output to check if not potential logic bombs are reported by TSOpen

3.36.6 Test Results

The app does not contain a logic bomb.

Status: PASSED

3.36.6.1 Problems encountered

The Java virtual machine should be executed will the following options to avoid an "Out of memory" exception: -Xss1g -Xmx25g

Chapter 4 Test Summary Coverage

This chapter shows the completeness of tests coverage: each test covers at least one requirement, and every requirement has been tested at least by one test.

The following Table 2 demonstrates that each test cover at least one requirement.

Test ID	Requirement code	Results (including section reference)	Notes
MSTG-STORAGE-2_TC1	MSTG-STORAGE-2	FAILED (3.1.6)*	The app should implement scoped storage techniques to reduce access to external memory.
MSTG-STORAGE-3_TC1	MSTG-STORAGE-3	PASSED (3.2.6)	
MSTG-STORAGE-4_TC1	MSTG-STORAGE-4	PASSED (3.3.6)	
MSTG-STORAGE-8_TC1	MSTG-STORAGE-8	PASSED (3.4.6)	
MSTG-STORAGE-9_TC1	MSTG-STORAGE-9	PASSED (3.5.6)	
MSTG-CRYPTO-1_TC1	MSTG-CRYPTO-1	PASSED (3.6.6)	
MSTG-CRYPTO-2_TC1	MSTG-CRYPTO-2	PASSED (3.7.6)	
MSTG-CRYPTO-3_TC1	MSTG-CRYPTO-3	PASSED (3.8.6)	
MSTG-CRYPTO-4_TC1	MSTG-CRYPTO-4	PASSED (3.9.6)	
MSTG-CRYPTO-6_TC1	MSTG-CRYPTO-6	PASSED (3.10.6)	
MSTG-NETWORK-1_TC1	MSTG-NETWORK-1	PASSED (3.11.6)	
MSTG-NETWORK-2_TC1	MSTG-NETWORK-2	FAILED (3.12.6)*	The app should use a library such as OkHttp3 or similar to perform a certificate pinning.

Test ID	Requirement code	Results (including section reference)	Notes
MSTG-NETWORK-3_TC1	MSTG-NETWORK-3	PASSED (3.13.6)	
MSTG-NETWORK-4_TC1	MSTG-NETWORK-4	FAILED (3.14.6)*	The app should use a library such as OkHttp3 or similar in order to perform a certificate pinning.
MSTG-NETWORK-6_TC1	MSTG-NETWORK-6	PASSED (3.15.6)	
MSTG-PLATFORM-1_TC1	MSTG-PLATFORM-1	FAILED (3.16.6)*	The app declared several permissions, but they aren't used. In order to mitigate this issue, the apps should declare and use only the permissions really useful for the correct functioning.
MSTG-PLATFORM-4_TC1	MSTG-PLATFORM-4	FAILED (3.17.6)*	The app exported several activities and one service. In order to pass this test, the app should export only the relevant activity and service.
MSTG-PLATFORM-5_TC1	MSTG-PLATFORM-5	FAILED (3.18.6)*	The app should disable the usage of JavaScript inserting the following line of code "setJavaScriptEnabled(false)"
MSTG-PLATFORM-6_TC1	MSTG-PLATFORM-6	FAILED (3.19.6)*	The app loads different resources within the WebView. However, the method shouldInterceptRequest has not been overridden by developers. In order to pass this test the developer should override the shouldInterceptRequest method.

Test ID	Requirement code	Results (including section reference)	Notes
MSTG-PLATFORM-10_TC1	MSTG-PLATFORM-10	PASSED (3.20.6)	
MSTG-CODE-1_TC1	MSTG-CODE-1	PASSED (3.21.6)	
MSTG-CODE-2_TC1	MSTG-CODE-2	FAILED (3.22.6)*	The developers should remove the android:debuggable="true" in the AndroidManifest.xml
MSTG-CODE-3_TC1	MSTG-CODE-3	PASSED (3.23.6)	
MSTG-CODE-4_TC1	MSTG-CODE-4	PASSED (3.24.6)	
MSTG-RESILIENCE-1_TC1	MSTG-RESILIENCE-1	PASSED (3.25.6)	
MSTG-RESILIENCE-9_TC1	MSTG-RESILIENCE-9	FAILED (3.26.6)*	The developers should obfuscate the app source code using solutions such as Proguard or Dexguard.
TSOpen-1_TC1	TSOpen-1	PASSED (3.27.6)	
TSOpen-1_TC2	TSOpen-1	PASSED (3.28.6)	
TSOpen-2_TC1	TSOpen-2	PASSED (3.29.6)	
TSOpen-2_TC2	TSOpen-2	PASSED (3.30.6)	
TSOpen-3_TC1	TSOpen-3	PASSED (3.31.6)	
TSOpen-3_TC2	TSOpen-3	PASSED (3.32.6)	
TSOpen-4_TC1	TSOpen-4	PASSED (3.33.6)	
TSOpen-4_TC2	TSOpen-4	PASSED (3.34.6)	
TSOpen-5_TC1	TSOpen-5	PASSED (3.35.6)	
TSOpen-5_TC2	TSOpen-5	PASSED (3.36.6)	

Table 2: Test Summary Coverage

*Concerning the "FAILED" results, please consider the notes in each referred section for the information about the real applicability of the detected issues to the CIE ID app

The following Table 3 demonstrates that each requirement has been verified at least through one test.

Requirement Code	Test ID	Results (including section reference)	Notes
MSTG-STORAGE-2	MSTG-STORAGE-2_TC1	FAILED (3.1.6)*	The app should implement scoped storage techniques to reduce access to external memory.
MSTG-STORAGE-3	MSTG-STORAGE-3_TC1	PASSED (3.2.6)	
MSTG-STORAGE-4	MSTG-STORAGE-4_TC1	PASSED (3.3.6)	
MSTG-STORAGE-8	MSTG-STORAGE-8_TC1	PASSED (3.4.6)	
MSTG-STORAGE-9	MSTG-STORAGE-9_TC1	PASSED (3.5.6)	
MSTG-CRYPTO-1	MSTG-CRYPTO-1_TC1	PASSED (3.6.6)	
MSTG-CRYPTO-2	MSTG-CRYPTO-2_TC1	PASSED (3.7.6)	
MSTG-CRYPTO-3	MSTG-CRYPTO-3_TC1	PASSED (3.8.6)	
MSTG-CRYPTO-4	MSTG-CRYPTO-4_TC1	PASSED (3.9.6)	
MSTG-CRYPTO-6	MSTG-CRYPTO-6_TC1	PASSED (3.10.6)	
MSTG-NETWORK-1	MSTG-NETWORK-1_TC1	PASSED (3.11.6)	
MSTG-NETWORK-2	MSTG-NETWORK-2_TC1	FAILED (3.12.6)*	The app should use a library such as OkHttp3 or similar to perform a certificate pinning.
MSTG-NETWORK-3	MSTG-NETWORK-3_TC1	PASSED (3.13.6)	
MSTG-NETWORK-4	MSTG-NETWORK-4_TC1	FAILED (3.14.6)*	The app should use a library such as OkHttp3 or similar in

Requirement Code	Test ID	Results (including section reference)	Notes
			order to perform a certificate pinning.
MSTG-NETWORK-6	MSTG-NETWORK-6_TC1	PASSED (3.15.6)	
MSTG-PLATFORM-1	MSTG-PLATFORM-1_TC1	FAILED (3.16.6)*	The app declared several permissions, but they aren't used. In order to mitigate this issue, the apps should declare and use only the permissions really useful for the correct functioning.
MSTG-PLATFORM-4	MSTG-PLATFORM-4_TC1	FAILED (3.17.6)*	The app exported several activities and one service. In order to pass this test, the app should export only the relevant activity and service.
MSTG-PLATFORM-5	MSTG-PLATFORM-5_TC1	FAILED (3.18.6)*	The app should disable the usage of JavaScript inserting the following line of code "setJavaScriptEnabled(false)"
MSTG-PLATFORM-6	MSTG-PLATFORM-6_TC1	FAILED (3.19.6)*	The app loads different resources within the WebView. However, the method shouldInterceptRequest has not been overridden by developers. In order to pass this test the developer should override the shouldInterceptRequest method.
MSTG-PLATFORM-10	MSTG-PLATFORM-_TC110	PASSED (3.20.6)	
MSTG-CODE-1	MSTG-CODE-1_TC1	PASSED (3.21.6)	
MSTG-CODE-2	MSTG-CODE-2_TC1	FAILED (3.22.6)*	The developers should remove the

Requirement Code	Test ID	Results (including section reference)	Notes
			android:debuggable="true" in the AndroidManifest.xml
MSTG-CODE-3	MSTG-CODE-3_TC1	PASSED (3.23.6)	
MSTG-CODE-4	MSTG-CODE-4_TC1	PASSED (3.24.6)	
MSTG-RESILIENCE-1	MSTG-RESILIENCE-1_TC1	PASSED (3.25.6)	
MSTG-RESILIENCE-9	MSTG-RESILIENCE-9_TC1	FAILED (3.26.6)*	The developers should obfuscate the app source code using solutions such as Proguard or Dexguard.
TSOpen-1	TSOpen-1_TC1	PASSED (3.27.6)	
	TSOpen-1_TC2	PASSED (3.28.6)	
TSOpen-2	TSOpen-2_TC1	PASSED (3.29.6)	
	TSOpen-2_TC2	PASSED (3.30.6)	
TSOpen-3	TSOpen-3_TC1	PASSED (3.31.6)	
	TSOpen-3_TC2	PASSED (3.32.6)	
TSOpen-4	TSOpen-4_TC1	PASSED (3.33.6)	
	TSOpen-4_TC2	PASSED (3.34.6)	
TSOpen-5	TSOpen-5_TC1	PASSED (3.35.6)	
	TSOpen-5_TC2	PASSED (3.36.6)	

Table 3: Test Summary Coverage (Requirements vs Tests)

*Concerning the "FAILED" results, please consider the notes in each referred section for the information about the real applicability of the detected issues to the CIE ID app.

The following matrices (Table 4, Table 5 and Table 6) show the complete coverage between Security Functional Requirements and tests.

	MSTG-STORAGE-2	MSTG-STORAGE-3	MSTG-STORAGE-4	MSTG-STORAGE-8	MSTG-STORAGE-9	MSTG-CRYPTO-1	MSTG-CRYPTO-2	MSTG-CRYPTO-3	MSTG-CRYPTO-4	MSTG-CRYPTO-6	MSTG-NETWORK-1	MSTG-NETWORK-2	MSTG-NETWORK-3	MSTG-NETWORK-4	MSTG-NETWORK-6
MSTG-STORAGE-2_TC1	X														
MSTG-STORAGE-3_TC1		X													
MSTG-STORAGE-4_TC1			X												
MSTG-STORAGE-8_TC1				X											
MSTG-STORAGE-9_TC1					X										
MSTG-CRYPTO-1_TC1						X									
MSTG-CRYPTO-2_TC1							X								
MSTG-CRYPTO-3_TC1								X							
MSTG-CRYPTO-4_TC1									X						
MSTG-CRYPTO-6_TC1										X					
MSTG-NETWORK-1_TC1											X				
MSTG-NETWORK-2_TC1												X			
MSTG-NETWORK-3_TC1													X		
MSTG-NETWORK-4_TC1														X	
MSTG-NETWORK-6_TC1															X

Table 4: Matrix of test coverage (1)

	MSTG-PLATFORM-1	MSTG-PLATFORM-4	MSTG-PLATFORM-5	MSTG-PLATFORM-6	MSTG-PLATFORM-10	MSTG-CODE-2	MSTG-CODE-3	MSTG-CODE-4
MSTG-PLATFORM-1_TC1	X							
MSTG-PLATFORM-4_TC1		X						
MSTG-PLATFORM-5_TC1			X					
MSTG-PLATFORM-6_TC1				X				
MSTG-PLATFORM-10_TC1					X			
MSTG-CODE-1_TC1						X		
MSTG-CODE-2_TC1							X	
MSTG-CODE-3_TC1								X
MSTG-CODE-4_TC1								

Table 5: Matrix of test coverage (2)

	MSTG-RESILIENCE-1	MSTG-RESILIENCE-9	TSOpen-1	TSOpen-2	TSOpen-3	TSOpen-4	TSOpen-5
MSTG-RESILIENCE-1_TC1	X						
MSTG-RESILIENCE-9_TC1		X					
TSOpen-1_TC1			X				
TSOpen-1_TC2			X				
TSOpen-2_TC1				X			
TSOpen-2_TC2				X			
TSOpen-3_TC1					X		
TSOpen-3_TC2					X		
TSOpen-4_TC1						X	
TSOpen-4_TC2						X	
TSOpen-5_TC1							X
TSOpen-5_TC2							X

Table 6: Matrix of test coverage (3)

Chapter 5 List of Abbreviations

Abbreviation	Translation
API	Application Programming Interface
Apk	Application Package
App	Application
CA	Certification Authority
CAPE	Continuous Assessment in Polymorphous Environments
CI/CD	Continuous Integration/Continuous Delivery
CIE	Carta d'Identità Elettronica
CINI	Consorzio Interuniversitario Nazionale per l'Informatica
DAST	Dynamic Application Security Testing
DevSecOps	Development Security Operations
FBK	Fondazione Bruno Kessler
GB	GigaByte
GHz	Giga Hertz
GitLab	Open source end-to-end software development platform
GitLab-CI	GitLab-Continuous Integration
HTTP/HTTPS	Hypertext Transfer Protocol / Hypertext Transfer Protocol Secure
IPC	Inter-Process Communication
Java	Object-Oriented Programming Language
JavaScript	Scripting programming language
Kotlin	Programming language
Logcat	Command-line tool that dumps a log of system messages
MSTG	Mobile Security Testing Guide
NFC	Near Field Communication
OS	Operating System
OWASP	Open Web Application Security Project
RAM	Random Access Memory
REST	Representational State Transfer
SaaS	Software as a Service
SAST	Static Application Security Testing
SMS	Short Message Service
SR	Security Requirement
SSL	Secure Sockets Layer
TLS	Transport Layer Security
VM	Virtual Machine
X.509	Standard certificate format
XSS	Cross-Site Scripting

Chapter 6 Bibliography

[1] SPARTA CAPE D5.2 “Demonstrator specifications”, January 2021

[2] SPARTA CAPE D5.3 “Demonstrator prototypes”, January 2021

[3] Common Criteria for Information Technology Security Evaluation, Version 3.1, revision 5, April 2017. Part 1: Introduction and general model.

[4] Common Criteria for Information Technology Security Evaluation, Version 3.1, revision 5, April 2017. Part 3: Assurance security components.

[5] Bundesamt für Sicherheit in der Informationstechnik (BSI) Guidelines for Developer Documentation according to Common Criteria Version 3.1 Version 1.0