D5.4 Appendix C

ATE – Tests – Vertical 1 Scenario 5 (Sabotage)

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 C
Work package contributing to the deliverable	WP5
Due date	Jan 2022 – M36
Actual submission date	2nd February, 2022

Responsible organisation	TECNALIA
Editor	Estibaliz Amparan
Dissemination level	PU
Revision	V1.0

Abstract	This document is part of the ATE activities performed during task 5.4 of the CAPE program of the SPARTA project. This document contains test procedures and report to evaluate the requirements related with Sensor based Plausibility check security mechanism
Keywords	Assessment, security test, platoon, safety, security, connected cars

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

Editor

Estibaliz Amparan (TEC)

Contributors (ordered according to beneficiary numbers)

Cristina Martínez, Angel López (TEC)

Mirko Malacario, Claudio Porretti, Nicoletta Imperatori (LEO)

Reviewers (ordered according to beneficiary numbers)

Maximilian Tschirschnitz (TUM)

Rimantas Zylius (L3CE)

Disclaimer

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 “Fault-injection and analysis of faulty scenarios” Scenario (also known as Scenario 5) of the “Connected and Cooperative Car Cybersecurity” vertical (also known as Connected Car Vertical or Vertical 1).

For the Scenario 5, the Basic Scenario described on deliverable D5.2 [1] and D5.3 [2] is used as basis. The goal of the scenario 5 is to evaluate the requirement related with Sensor based Plausibility check security mechanism. Thus, the Sabotage tool will be used to simulate how different faults, originated from a random hardware fault or cyber-attack, affects to the behaviour of the security mechanism.

Figure 1 shows the life-cycle of the requirements that have been elicited for the Connected Car Vertical. First of all, the requirements have been stated in the Protection Profile document [3]. Secondly, the requirements have been addressed in Secure Design, as it was documented in D5.2 [1] and D5.3 [2]. Finally, the requirements have been tested and assessed, and this process has been documented in the current document, thus supporting the ATE test phase of the evaluation procedure based on Common Criteria [4], [5] and [6].

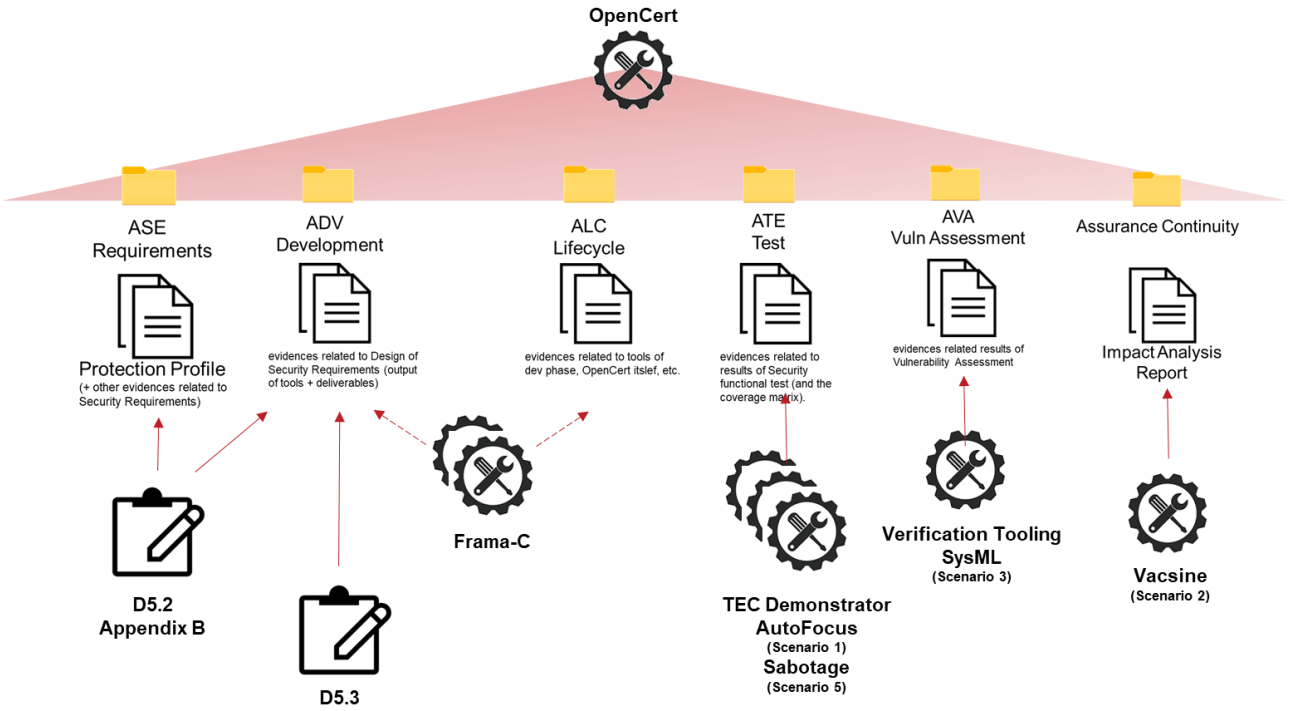

Figure 1: Evaluation strategy for the Connected Car Vertical

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 preparation** 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 preparation

2.1 System overview

The Connected Car Vertical (Vertical 1) has been fully described in D5.2 [1]. In this section, we provide an overview of the case study description, focusing first on the first scenario, named “Basic Scenario”, since Scenario 5 will be based on this first scenario.

The goal of the Connected Car Vertical is to advance the cyber-security of connected vehicles driving in platoon mode. A platoon is a sequence of vehicles as depicted by Figure 2, that it is composed by a leader vehicle and a sequence of followers.

Each vehicle in the platoon communicates using dedicated communication channels. Moreover, each vehicle in the platoon possesses sensors, such as cameras, distance sensors, enabling a highly automated mode of operation. Indeed, when formed, the platoon requires only driver supervision.

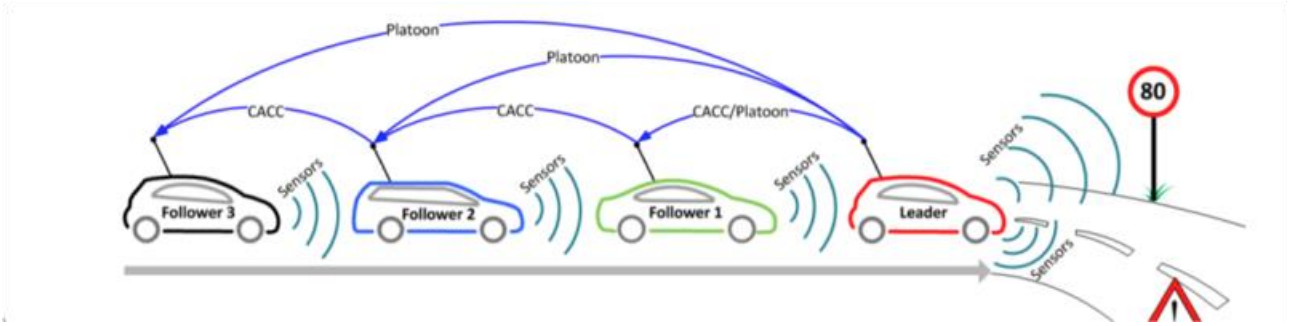

Figure 2: Platooning scenario

We consider a platoon of three members, with one leader and two followers using Cooperative Adaptive Cruise Control (CACC). All cars have exactly the same hardware and the same platooning software but with different configurations.

The platoon vehicles navigate on the circuit designed.

The vehicles can communicate each other thanks to a WiFi 802.11n access point.

For the Scenario 5, all the tests will be carried out in simulation using the Sabotage tool. Sabotage is based on simulation-based fault injection methodology. It will be used in the first phases of the v-cycle where some vehicle functions will be simulated to test the efficiency of the security mechanism to be developed.

2.1.1 Hardware preparation

The hardware preparation is not applicable on this scenario. Tests will be performed in a simulated environment.

2.1.2 Software preparation

As mentioned in the previous section, certain functionalities of the vehicles will be simulated. More specifically, the reception of the information received by the distance sensor (ultrasonic sensor) and the information sent by the previous vehicle (WiFi communication) will be simulated.

In Figure 3, the “distance sensor” block represents the function of the vehicle distance sensor. The “Scenario” block gets the information of the speed of the vehicle in front and the speed on the vehicle that is testing.

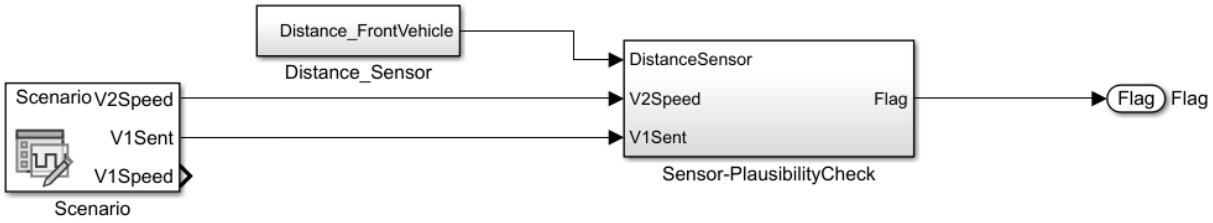

Figure 3: Simulink model

The goal of the different tests to be carried out is to ensure the correct operation of the sensor-based plausibility check algorithm. The mission of this algorithm is detecting if the information received from the preceding vehicle is reliable or, on the contrary, the vehicle may be under attack. The information received by the vehicle will be cross-checked with the information received from the proximity sensor. This algorithm has been developed within the "Sensor-Plausibility Check" block (see Figure 3).

A mathematical algorithm has been developed to check if the distance received by the sensor and the preceding vehicle speed are coherent.

$$v1 = v2 + \frac{d_t - d_0}{t}$$

Using the distance sensor, the difference of distance between the previous instant (*d0*) and the current distance (*dt*) is calculated. Thus, we will verify if the distance between both vehicles remains the same or if, on the contrary, it has changed. Multiplying this value by the time (*t*), we can calculate the speed that has increased or decreased. On the next step, as this speed is relative because it is calculated with respect to the position of the rear vehicle, we must add the speed of the rear vehicle (*v2*). Finally, we should compare if the calculated front vehicle speed and the speed sent by the preceding vehicle are coherent. They will be consistent as long as the difference between them is not greater than 10%.

To carry out the following tests we are going to use the Sabotage tool.

Sabotage is a simulation-based fault injection tool, where some specific faults are injected into the system to observe its behaviour. The Sabotage tool is rooted on the Eclipse and Matlab/Simulink frameworks and is able to configure, run and analyse the simulation results.

Sabotage is the responsible to automatically inject faults into the plausibility check algorithm and compute the results. It gives the designer the possibility of creating the fault list and selecting where to monitor fault injection experiments by including signal monitors (readout points). On the test configuration, two steps are followed. The first one is the creation of the fault list where it will be selected the fault what will be added into the system as preconfigured blocks in Simulink. The fault list is used to define the faults that will be injected on the system. These faults are characterised by a type (e.g. frozen, stuckat0, delay, invert, oscillation or random), target location, injection, and duration. The second step is the definition of readout points, where the most important signals of the Simulink system are stored in a mldatx file.

After performing the configuration of the fault injection scenarios, it is possible to run the simulation. The simulation will be performed automatically; the result will be stored and shown graphically as shown in the Figure 4.

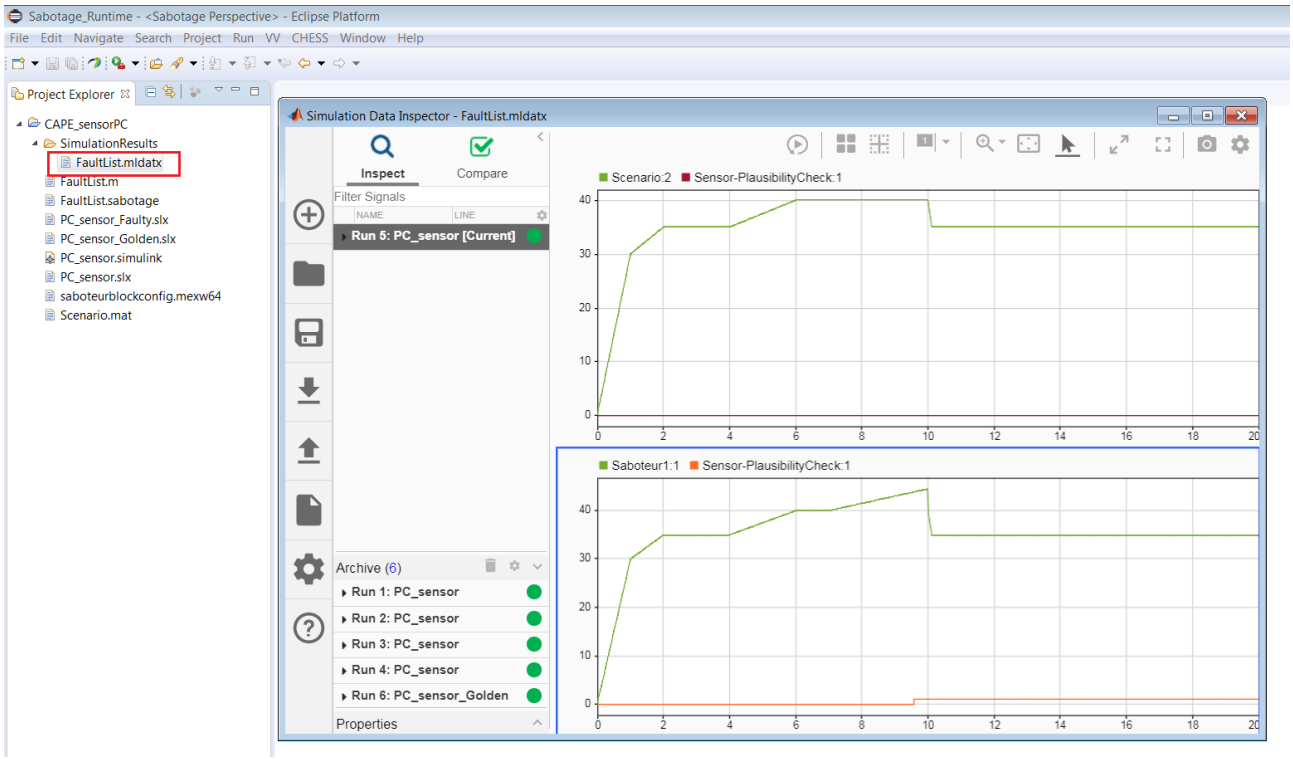

Figure 4: Result stored in “mldatx” file and graphical view

Chapter 3 Test descriptions

Table 1 shows the requirements that have been implemented for the Scenario 5 in the Connected Car Vertical. In the next sections, we describe the test descriptions that have been elaborated to support the test of these requirements.

Req. Id	Short Description
PMM_VCS-SPC.2	VCS message consistent to distances history

Table 1: Requirements covered by the Sabotage tool in the Scenario 5

3.1 PMM_VCS-SPC.2_TC1

Test case to validate that the plausibility check algorithm does not set an attack flag when the error tolerance is less than 10% between the calculated distance and the distance provided by the distance sensor.

3.1.1 Security Requirements addressed

PMM_VCS-SPC.2

3.1.2 Test preconditions

There is a platoon composed of one leader and one follower.

There are no obstacles in the circuit.

The speed of the leader vehicle is variable during the simulation and the safety distance between them, (50 cm) is preconfigured.

The error tolerance to consider that an attack (or failure in the system) is taking place is defined within the algorithm at 10%.

3.1.3 Expected test results

The security system does not activate.

3.1.4 Criteria for evaluating results

The test duration is 20 seconds.

Verify that the value of the flag is always 0 during the test.

3.1.5 Test Procedure

The following steps will be carried out:

- Load the Sensor based plausibility model in Sabotage.
- Load the preconfigured scenario. In this preconfigured scenario, in this case, the speed of vehicle 1 will be defined as an input.
- On the test configuration, two steps will be followed. First the creation of the fault list and then the definition of the readout points.

- Create the fault injection list and select the different faults, trigger times and durations of each one: In this test, we define the fault that will be reproduced in the simulations. For the test, one fault is defined. This fault will modify the values sent by the vehicle in front. The fault type is a stuck-value and its value will be a 7% high with respect original value, and it is activated at second 1 of the simulation during all 20 seconds.
- Define readout points: From a list that contains all the signals of the system it will be selected the most interesting signals to check. In this case, the signals to check will be the speed of the vehicle in front (V1 Real Speed), the speed sent by the vehicle in front (V1Sent) and the output values of the Sensor-based plausibility check flag).
- Run the Simulations.
- When the simulation has finished the results are stored and a graphical view is automatically opened to analyze the results.
- Analyze the graphical view and verify that the Flag signal in the “Sensor-Plausibility Check” block has a 0 value (red line in following Figure 5).

3.1.6 Test Results

The simulation result values are stored and visualized on a graphical way as shown in Figure 5. These values represent the speed of the vehicle in front (V1 Real Speed); the speed sent by the vehicle in front (V1Sent) and the output values of the Sensor-based plausibility check flag. It is possible to observe that when adding false speed values sent by the vehicle in front (V1Sent) that are not greater than 10%, the sensor-based plausibility check flag is never activated.

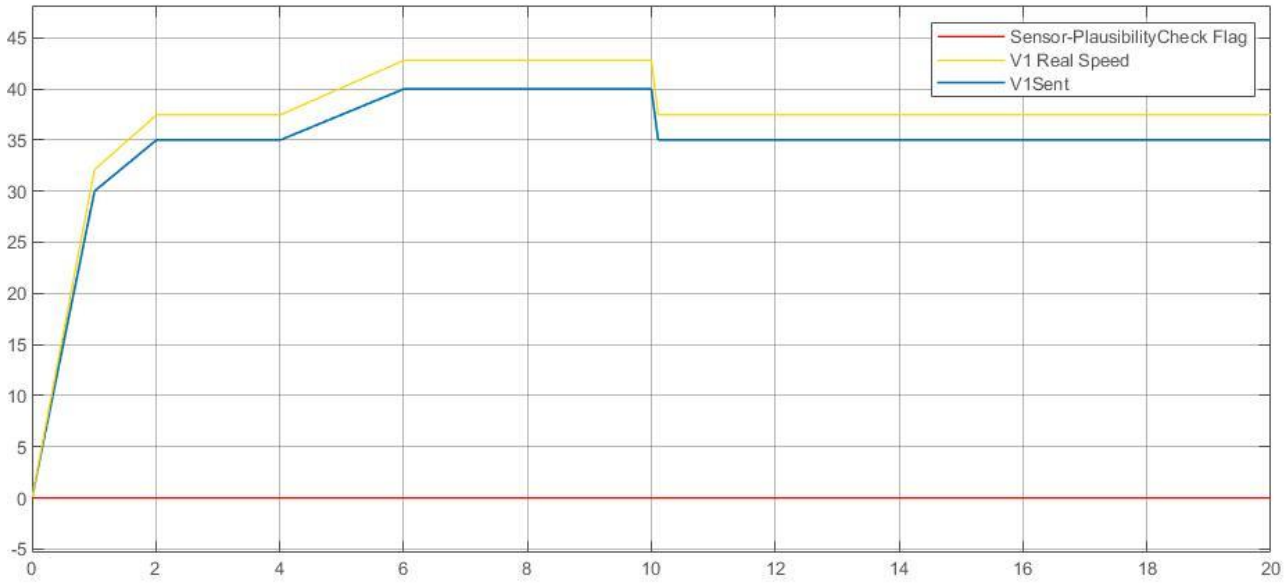

Figure 5: Simulation results when tolerance error is less than 10%.

In this way, we can define an acceptable tolerance error between the sensor measurement and the theoretical distance without considering this difference as an attack.

Status: *PASSED*

3.2 PMM_VCS-SPC.2_TC2

Test case to validate that the plausibility check algorithm sets an attack flag when the error tolerance is greater than 10% between the calculated distance and the distance provide by the distance sensor.

3.2.1 Security Requirements addressed

PMM_ VCS-SPC.2

3.2.2 Test preconditions

There is a platoon composed of one leader and one follower.

There are no obstacles in the circuit.

The speed of the leader vehicle is variable during the simulation and the safety distance between them, (50 cm) is preconfigured.

The error tolerance to consider that an attack (or failure in the system) is taking place is defined within the algorithm at 10%.

3.2.3 Expected test results

The security system will be activated.

3.2.4 Criteria for evaluating results

The test duration is 20 seconds.

Verify that the value of the Sensor-based plausibility check flag is always 1 during the test whenever the tolerance error is bigger than 10%.

3.2.5 Test Procedure

The following steps will be carried out:

- Load the Sensor based plausibility model in Sabotage.
- Load the preconfigured scenario. In this case, the speed of leader will be defined as an input.
- On the test configuration, two steps will be followed. First the creation of the fault list and then the definition of the readout points.
- Create a fault injection list and select the different faults, trigger times and durations of each one. In this list, we define the faults that will be reproduced in the simulations. For the test, one fault is defined. This fault will modify the values sent by the vehicle in front. The fault type is a stuck-value and its value will be a 40% high with respect original value, and it is activated at second 1 of the simulation during all 20 seconds.
- Define readout points: From a list that contains all the signals of the system it will be selected the most interesting signals to check. In this case, the signals to check will be the speed of the vehicle in front (V1 Real Speed), the speed sent by the vehicle in front (V1Sent) and the output values of the Sensor-based plausibility check flag).
- Run the Simulations.
- Once the simulation has finished, the results are stored, and a graphical view is automatically opened to analyze the results.
- Analyze the graphical view and verify that the value of the Sensor Plausibility Check flag signal is 1 (red line in following Figure 6).

3.2.6 Test Results

The simulation result values are stored and visualized on a graphical way as shown in Figure 6. These values represent speed of the vehicle in front, the speed sent by the vehicle in front and the output values of the sensor-based plausibility check flag. It is possible to observe that when adding faulty speed sent values greater than 10%, the sensor-based plausibility check flag is active.

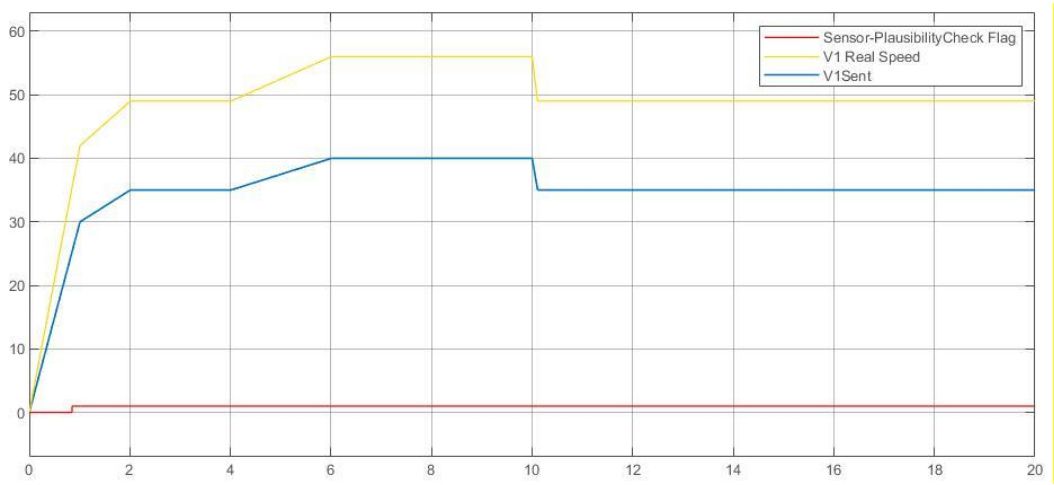

Figure 6: Simulation results when tolerance error is bigger than 10%.

Status: *PASSED*

3.3 PMM_VCS-SPC.2_TC3

Test case to validate that the plausibility check algorithm sets an attack flag when a slow attack has been produced, this means that the tolerance error is not more than 10% with respect to the last three stored values, but a cumulative error occurs.

3.3.1 Security Requirements addressed

PMM_VCS-SPC.2

3.3.2 Test preconditions

There is a platoon composed of one leader and one follower.

There are no obstacles in the circuit.

The speed of the leader vehicle is variable during the simulation and the safety distance between them, (50 cm) is preconfigured.

The error tolerance to consider that an attack (or failure in the system) is taking place is defined within the algorithm at 10%.

3.3.3 Expected test results

The security system will be activated.

3.3.4 Criteria for evaluating results

The test duration is 20 seconds.

Check in the simulation results when the sensor-based plausibility check has activated the Sensor Plausibility Check flag signal.

3.3.5 Test Procedure

The following steps will be carried out:

- Load the Sensor based plausibility model in Sabotage.
- Load the preconfigured scenario. In this case, the speed of vehicle 1 will be defined as an input.
- On the test configuration, two steps will be followed. First the creation of the fault list and then the definition of the readout points.
- Create a fault injection list and select the different faults, trigger times and durations of each one. In this list, we define the faults that will be reproduced in the simulations. For the test, one fault is defined. This fault will modify the values sent by the vehicle in front. The fault type is a ramp-up with a slope of 50% and its value will be added to original value, and it is activated at second 1 of the simulation during all 20 seconds.
- Define readout points: From a list that contains all the signals of the system it will be selected the most interesting signals to check. In this case, the signals to check will be the speed of the vehicle in front (V1 Real Speed), the speed sent by the vehicle in front (V1Sent) and the output values of the Sensor-based plausibility check flag).
- Run the Simulations.
- Once the simulation has finished, the results are stored, and a graphical view is automatically opened to analyze the results.
- Analyze the graphical view and verify that the value of the Sensor Plausibility Check Flag signal is 1.

3.3.6 Test Results

The simulation result values are stored and visualized on a graphical way as shown in Figure 7.

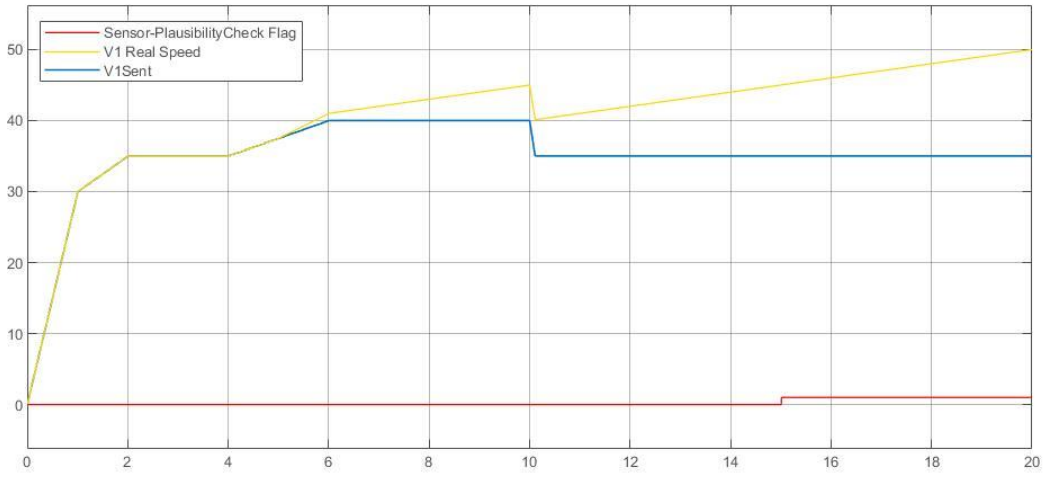

Figure 7: Ramp up error simulation results.

Status: *PASSED*

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.

Test ID	Requirement code	Results (including section reference)	Notes
PMM_ VCS-SPC.2_TC1	PMM_VCS-SPC.2	PASSED (3.1.6)	--
PMM_ VCS-SPC.2_TC2	PMM_VCS-SPC.2	PASSED (3.2.6)	--
PMM_ VCS-SPC.2_TC3	PMM_VCS-SPC.2	PASSED (3.3.6)	--

Table 2: Test Summary Coverage (Tests vs Requirements)

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
PMM_VCS-SPC.2	PMM_VCS-SPC.2_TC1 PMM_VCS-SPC.2_TC2 PMM_VCS-SPC.2_TC3	PASSED (3.1.6) PASSED (3.2.6) PASSED (3.3.6)	--

Table 3: Test Summary Coverage (Requirements vs Tests)

The following matrix (Table 4) shows the complete coverage between Security Functional Requirements and tests

	PMM_ VCS-SPC.2
PMM_ VCS-SPC.2_TC1	X
PMM_ VCS-SPC.2_TC2	X
PMM_ VCS-SPC.2_TC3	X

Table 4: Matrix of test coverage

Chapter 5 List of Abbreviations

Abbreviation	Translation
ACC	Adaptive Cruise Control
ATE	Assurance class TEst
CACC	Cooperative Adaptive Cruise Control
PMM	Platoon Management Module
SPC	Sensor-based Plausibility Check
TC	Test Case
VCM	Vehicle Control Module
VCS	Vehicle Communication System

Chapter 6 Bibliography

[1] SPARTA D5.2 Demonstrators specifications. January 2021.

[2] SPARTA D5.3 Demonstrator prototypes. January 2021.

[3] SPARTA D5.2 Appendix B Protection Profile for a Safety and Security Platooning Management Module, January 2021

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

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

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