D5.4 Appendix B

ATE – Tests – Vertical 1 Scenario 1 (AutoFOCUS3)

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

Responsible organisation	FTS
Editor	Yuri Gil Dantas
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 for the AutoFOCUS3 tool used in Scenario 1 named Basic Scenario
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

Yuri Gil Dantas (FTS)

Contributors (ordered according to beneficiary numbers)

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 describes the test cases for evaluating the correctness of the platooning (security) requirements elicited for the basic scenario (also known as Scenario 1) of the Connected and Cooperative Car Cybersecurity" vertical project (also known as Connected Car Vertical or Vertical 1).

The platooning requirements are described in Protection Profile document [5]. These requirements have been implemented in the model-based engineering tool AutoFOCUS3 (AF3 for short) [7]. We have evaluated the correctness of such requirements by means of simulations and by means of experiments (as documented in D5.2 [4] and D5.3 [6]). The test cases described in this document focuses on the evaluation performed by means of simulation.

To evaluate the correctness of such requirements, we specify inputs for each requirement and observe the output results to test compliance with the requirement. We run the test cases with the help of the Simulator View of AF3.

The remainder of this document is structured as follows:

- **Chapter 1 Introduction** is the current section presenting the objectives, scope and structure of the document.
- **Chapter 2 Test preparations,** presents AutoFOCUS3 in a nutshell as well as software needed for running the tests cases.
- **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 Connected Car Vertical (Vertical 1) has been fully described in D5.2 [4]. In this section, we provide an overview of the case study description, focusing on the first scenario, named “Basic 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 1, 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.

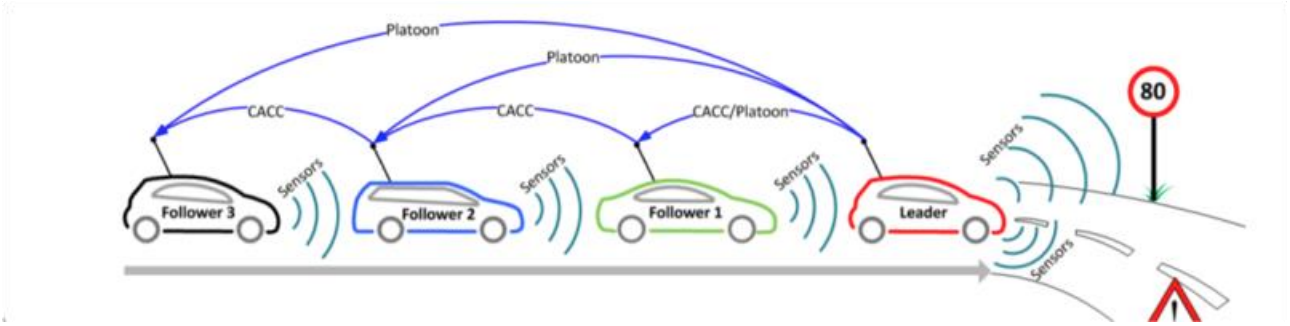

Figure 1: Platooning scenario

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.

2.1.1 AutoFOCUS3

AutoFOCUS3 (AF3 for short) is a model-based engineering tool for safety-critical systems [7]. AF3 supports the design, development and validation of safety-critical systems in many development phases, including architectural design, and implementation.

AF3 enables the specification of, e.g., the logical architecture of the system that includes components and channels. For each component specified, AF3 provides two ways to specify the behaviour of such components, namely code specification or automaton specification. Code specification allows one to specify the behaviour of components in a C-like language, and automaton specification in a graphical state automaton diagram with states and transitions between states. The architecture as well as the behaviour of the platoon vehicles have been designed in AF3.

AF3 provides a way to validate the behavior of components by means of simulation. Figure 2 illustrates the AF3 simulation perspective. It simulates the behavior of **ComponentB** that has a very simplistic implementation to increment integer values. To run the simulations, one needs to manually provide the input values for components (see value 3 for input port **inputB**). The result of the simulation is shown on the right-hand side of Figure 2, i.e., the output computed by **ComponentB** is 4 (see output port value for **outputB**).

We use the AF3 simulation perspective for validating the platooning requirements elicited for the basic scenario, as shown in the next chapter.

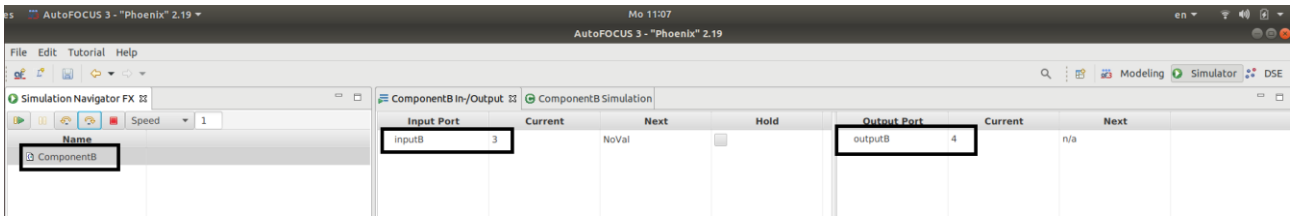

Figure 2. Illustration of the AF3 Simulator View

2.1.2 Software preparation

The test cases presented in the next chapter are performed in AutoFOCUS3 using the CACC model designed by FORTISS. To run the test cases, the following tests are needed:

1. Download the binary of AutoFOCUS3 available at [7]
2. Download the CACC model available at [8]
3. Open AutoFOCUS3
4. Import the CACC model into AutoFOCUS3
5. The behavior of any component of the model can be simulated. In order to execute a simulation, select a component in your Modelling Perspective, open the context menu (by right-clicking on it) and choose “Run Simulator”. In the next chapter, we make explicit what component shall be simulated.

Chapter 3 Test descriptions

Table 1 shows the requirements that have been implemented for the Scenario 1 in AutoFOCUS3. The next sections describe the tests cases elaborated to evaluate the correctness of the implemented requirements.

Req. Id	Short Description
PMM_IF.1	Maintain heart-beat data (vehicle identifier, speed, direction, geo-position, timestamp) to VCS
PMM_IF.2	Maintain heart-beat data from VCS
PMM_IF.3	Maintain incoming emergency brake information flow from other vehicles
PMM_IF.4	Maintain outgoing emergency brake information flow to other vehicles
PMM_IF.5	Maintain data from VCM
PMM_IF.6	Maintain data to VCM
PMM_PC.1	Data passes all VCS plausibility checks
PMM_PC.2	Data passes all VCM plausibility checks
PMM_VCS-HPC.1	Maintain heart-beat data history
PMM_VCS-HPC.2	Heart-beat message consistent to the history
PMM_VCS-HPC.3	Emergency brake consistent to the history
PMM_VCS-SPC.1	Maintain distances history
PMM_VCS-SPC.2	VCS message consistent to the distances history
PMM_VCS-SPC.3	Emergency brake consistent to distances history
PMM_VCM-HPC.1	Maintain sensor data history
PMM_VCM-HPC.2	Sensor message consistent to the history

Table 1: Requirements covered by AutoFOCUS3 in the Scenario 1

The test cases for PMM_VCM-HPC.1 and PMM_VCM-HPC.2 are equivalent to the test cases for PMM_VCS-SPC.1 and PMM_VCS-SPC.2. Hence, the tests cases for PMM_VCM-HPC.1 and PMM_VCM-HPC.2 are omitted in this report.

Remark: In the following, the terms “speed” and “velocity” has been used as interchangeable.

3.1 PMM_IF.1_TC1

Test case to validate the generation and the composition of a heartbeat (HB) data message.

3.1.1 Security Requirements addressed

PMM_IF.1

3.1.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view (i.e., in the **platoonInfoIn** as described in the Test Procedure). We refer to the leader as *LDR* and the follower as *FLW*.

3.1.3 Expected test results

FLW generates a HB message that includes the vehicle unique identifier, the vehicle speed, and the vehicle steering angle.

3.1.4 Criteria for evaluating results

The generated HB message is displayed in the AF3 Simulator view.

3.1.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat() . This means that the received message is a HB message.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.0 (speed of the vehicle)
    - SteeringAngle: 11.00 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The HB message can be seen in the output port **platoonStoredNew.**

3.1.6 Test Results

Check the output port **platoonStoredNew,** in particular the members **id** (it will contain the value 2) **Velocity** (it will contain the value 4.0)**,** and **SteeringAngle** (it will contain the value 11.00).

Status: *PASSED WITH DEVIATIONS*

Deviations from test procedure

HB messages include the vehicle's unique identifier (id), vehicle speed (velocity), and direction (steering angle). It does not contain Geo-Position and Timestamp. We have not modelled GPS, especially because our rovers do not support GPS. Our model specifies the notion of tick (as an integer number) for managing and scheduling when vehicles sent or received messages. However, it does not record the actual time for when messages are, e.g., sent or received. Hence, we are not considering timestamps in our simulations.

3.2 PMM_IF.1_TC2

Test case to validate the sending of HB messages.

3.2.1 Security Requirements addressed

PMM_IF.1

3.2.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view (i.e., in the **platoonInfoIn** as described in the Test Procedure). We refer to the leader as *LDR* and the follower as *FLW*.

3.2.3 Expected test results

The generated HB message by FLW is sent to LDR.

3.2.4 Criteria for evaluating results

The HB message sent is displayed as output in the AF3 Simulator view.

3.2.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat() . This means that the received message is a HB message.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
- Define the inputs **Velocity** and **SteeringAngle**.

  - Velocity: 4.0 (speed of the vehicle)
  - SteeringAngle: 11.00 (steering angle of the vehicle)
- Define the input **DistanceFront**.
  - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The HB message can be seen in the output port **sendMessageType.**

3.2.6 Test Results

Check the output port **sendMessageType**. This port will display **followerMessage()** as output. This means that a HB message generated by FLW will be send to LDR.

Status: *PASSED WITH DEVIATIONS*

Deviations from test procedure

HB messages include the vehicle's unique identifier (id), vehicle speed (velocity), and direction (steering angle). It does not contain Geo-Position and Timestamp. We have not modelled GPS (for geo-position), especially because our rovers do not support GPS. Our model specifies the notion of tick (as an integer number) for managing and scheduling when vehicles sent or received messages. However, it does not record the actual time for when messages are, e.g., sent or received. Hence, we are not considering timestamps in our simulations. We also assumed that the messages are digitally signed and encrypted, but we do not model these in the platoon model.

3.3 PMM_IF.2_TC1

Test case to validate the reception of HB messages and its composition:
- Vehicle speed
- Direction
- Geo-Position
- Timestamp
- Digitally signed certificates

3.3.1 Security Requirements addressed

PMM_IF.2

3.3.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view (i.e., in the **platoonInfoIn** as described in the Test Procedure).We refer to the leader as *LDR* and the follower as *FLW*.

3.3.3 Expected test results

FLW receives a HB message generated by LDR.

3.3.4 Criteria for evaluating results

The HB message received by FLW is displayed as input in the AF3 Simulator view.

3.3.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat() . This means that the received message is a HB message.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.0 (speed of the vehicle)
    - SteeringAngle: 11.00 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the left-hand side of the Simulator view. The HB message can be seen in the input port **receiveMessageType**.

3.3.6 Test Results

Check the input port **receiveMessageType**. This port will display **heartbeat()** as input. This means that a HB message has been received. The content of the HB message is displayed in the input port **platoonInfoIn**. In particular, **platoonInfoIn** contains a **broadcastPlatoon** type with the unique ID of LDR (who sent the message), the LDR’s velocity, and LDR’s steering angle.

Status: *PASSED WITH DEVIATION*

Deviations from test procedure

HB messages include the vehicle's unique identifier (id), vehicle speed (velocity), and direction (steering angle). It does not contain Geo-Position and Timestamp. We have not modelled GPS (for geo-position), especially because our rovers do not support GPS. Our model specifies the notion of tick (as an integer number) for managing and scheduling when vehicles sent or received messages. However, it does not record the actual time for when messages are, e.g., sent or received. Hence, we are not considering timestamps in our simulations. We also assumed that the messages are digitally signed and encrypted, but we do not model these in the platoon model.

3.4 PMM_IF.3_TC1

Test case to validate the reception of emergency brake messages (EB) and its structure.

3.4.1 Security Requirements addressed

PMM_IF.3

3.4.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view.We refer to the leader as *LDR* and the follower as FLW. .

3.4.3 Expected test results

The EB message sent from LDR is received by FLW.

3.4.4 Criteria for evaluating results

The received EB message is displayed as input in the AF3 Simulator view and FLW sets the variable **EB** to true.

3.4.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 1 (amount of vehicle platoons – in our model amount equals to 1 means that LDR has triggered an emergency brake)
    - id: 2 (unique identifier of FLW)

  - leaderID: 1 (unique identifier of LDR)
- Define the inputs **Velocity** and **SteeringAngle**.
  - Velocity: 4.0 (speed of the vehicle)
  - SteeringAngle: 11.00 (steering angle of the vehicle)
- Define the input **DistanceFront**.
  - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The evidence that FLW has received an EB message can be seen in the output port **EB**.

3.4.6 Test Results

Check the output port **EB**. It has been set to **true**. This means that FLW1 will stop the vehicle following the EB message received by LDR.

Status: *PASSED WITH DEVIATION*

Deviations from test procedure

EB messages include the ID of the vehicle that triggered the emergency brake. It does not contain Timestamp. Our model specifies the notion of tick (as an integer number) for managing and scheduling when vehicles sent or received messages. However, it does not record the actual time for when messages are, e.g., sent or received. Hence, we are not considering timestamps in our simulations. We also assumed that the messages are digitally signed and encrypted, but we do not model these in the platoon model.

3.5 PMM_IF.4_TC1

Test case to validate the generation and the composition of EB messages.

3.5.1 Security Requirements addressed

PMM_IF.4

3.5.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.5.3 Expected test results

An EB message is generated by LDR, including broadcastPlatoon message that contains information from all vehicles in the platoon (i.e., unique ids, velocity, steeringAngle and position)

3.5.4 Criteria for evaluating results

The generated EB message is displayed as output in the AF3 Simulator view.

3.5.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Leader” by clicking on the Platoon States component and selecting the “Leader” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the sent message is a HB message to followers in the platoon (i.e., to FLW). .
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 1 (amount of vehicle platoons – in our model amount equals to 1 means that LDR has triggered an emergency brake)
    - id: 1 (unique identifier of LDR)
    - leaderID: 1 (unique identifier of LDR)
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.0 (speed of the vehicle)
    - SteeringAngle: 11.00 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The sent HB message can be seen in the output port **platoonInfoIn**.

3.5.6 Test Results

Check the output port **platoonInfoIn**. This port contains the member **amount** equals to 1 (i.e., an EB message has been triggered by LDR). It also contains the ID of the sender (i.e., id: 1).

Status: *PASSED WITH DEVIATIONS*

Deviations from test procedure

HB messages include the vehicle's unique identifier (id), vehicle speed (velocity), and direction (steering angle). It does not contain Geo-Position and Timestamp. We have not modelled GPS, especially because our rovers do not support GPS. Our model specifies the notion of tick (as an integer number) for managing and scheduling when vehicles sent or received messages. However,

it does not record the actual time for when messages are, e.g., sent or received. Hence, we are not considering timestamps in our simulations.

3.6 PMM_IF.4_TC2

Test case to validate the sending of EB messages from the leader to the follower.

3.6.1 Security Requirements addressed

PMM_IF.4

3.6.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as *LDR* and the follower as *FLW*.

3.6.3 Expected test results

EB messages generated by LDR are sent to FLW.

3.6.4 Criteria for evaluating results

A HB message (triggering an emergency brake) is the type of message sent to FLW.

3.6.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Leader” by clicking on the Platoon States component and selecting the “Leader” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the sent message is a HB message from LDR to FLW.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 1 (amount of vehicle platoons – in our model amount equals to 1 means that LDR has triggered an emergency brake)
    - id: 1 (unique identifier of LDR)
    - leaderID: 1 (unique identifier of LDR)
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.0 (speed of the vehicle)
    - SteeringAngle: 11.00 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)

- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The HB message can be seen in the output port **sendMessageType**.

3.6.6 Test Results

Check the output port **sendMessageType**. This port will display heartbeat() as output. This means that a HB message is generated from LDR to FLW will be send to LDR. Check also the output port **platoonInfoIn**. This port contains the member **amount** equals to 1 (i.e., an EB message has been triggered by LDR).

Status: *PASSED WITH DEVIATIONS*

Deviations from test procedure

HB messages include the vehicle's unique identifier (id), vehicle speed (velocity), and direction (steering angle). It does not contain Geo-Position and Timestamp. We have not modelled GPS, especially because our rovers do not support GPS. Our model specifies the notion of tick (as an integer number) for managing and scheduling when vehicles sent or received messages. However, it does not record the actual time for when messages are, e.g., sent or received. Hence, we are not considering timestamps in our simulations.

3.7 PMM_IF.5_TC1

Test case to validate the reception of incoming messages from VCM.

3.7.1 Security Requirements addressed

PMM_IF.5.1

3.7.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.7.3 Expected test results

The gap information from the sensor is received by FLW.

3.7.4 Criteria for evaluating results

The received gap information is received by FLW and the local history is updated with its value. This is displayed as output in the AF3 Simulator view.

3.7.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project

- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11. 0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 9.7 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The input of the simulation can be seen on the left-hand side of the Simulator view. The variable **DistanceFront** show the received gap information from the sensor.

3.7.6 Test Results

Check the output port **DistanceFront**. It shows the received gap information from the sensor. The input of the simulation can be seen on the left-hand side of the Simulator view. Check also the output port **historyNew**. It is updated with a new entry that includes the **DistanceFront** 9.7.

Status: *PASSED WITH DEVIATIONS*

Deviations from test procedure

HB messages include the vehicle's unique identifier (id), vehicle speed (velocity), direction (steering angle), and gap to the next vehicle (distanceFront). It does not contain the distance to the edges of the lane, because the modelled plausibility checks are currently not using this information. In our model, however, the lane keeping assistance component receives this information from the sensors.

3.8 PMM_IF.6_TC1

Test case to validate the outgoing information from the TOE to VCM.

3.8.1 Security Requirements addressed

PMM_IF.6

3.8.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view. We refer to the leader as *LDR* and the follower as *FLW*.

3.8.3 Expected test results

FLW adapts its speed based on the HB message received by LDR.

3.8.4 Criteria for evaluating results

The adapted speed value of FLW is shown as output of the Simulator view.

3.8.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat() . This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.0}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.0 (speed of the vehicle)
    - SteeringAngle: 11. 0 (steering angle of the vehicle)
- Define the input **DistanceFront**.

    - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The adapted speed value of FLW can be seen in the output port **platoonStoredNew**.

3.8.6 Test Results

Check the output port **platoonStoredNew**. This port contains the adapted speed value (based on the received HB message by LDR) in the member **Velocity**. This port also contains the steering angle in the member **SteeringAngle**.

Status: *PASSED*

3.9 PMM_PC.1-2_TC1

Test case to validate that the TOE accepts data incoming from the VCS only if the data passes all plausibility checks defined.

3.9.1 Security Requirements addressed

PMM_PC.1 and PMM_PC.2

3.9.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view. We refer to the leader as LDR and the follower as FLW.

3.9.3 Expected test results

FLW accepts incoming HB message if it passes the plausibility checks.

3.9.4 Criteria for evaluating results

The speed of FLW is changed only if it passes the plausibility checks; the changed speed is displayed as output in the AF3 Simulator view.

3.9.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.

  - receiveMessageType: heartbeat() . This means that the received message is a HB message from LDR.

- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
  - amount: 2 (amount of vehicle platoons)
  - id: 2 (unique identifier of FLW)
  - leaderID: 1 (unique identifier of LDR)
  -
  - broadcastPlatoon:[
    {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
    {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

    (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
  - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
  - Velocity: 4.9 (speed of the vehicle)
  - SteeringAngle: 11. 0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
  - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The adapted speed value of FLW can be seen in the output port **platoonStoredNew**.

3.9.6 Test Results

Check the output port **platoonStoredNew**. This port contains the adapted speed value (based on the received HB message by LDR) in the member **Velocity**. This is true because the received speed (labelled as **velocity**) and distance front (labelled as **distanceFront**) did not deviate in 30% w.r.t to the history.

Status: *PASSED.*

3.10 PMM_PC.1-2_TC2

Test case to validate that the TOE accepts data incoming from the VCS only if the data passes all plausibility checks defined.

3.10.1 Security Requirements addressed

PMM_PC.1 and PMM_PC.2

3.10.2 Test preconditions

The leader of the platoon and 1 follower shall be defined in the AF3 Simulator view. Note that more than 1 follower could be defined for this test, however only 1 follower is sufficient. We refer to the leader as LDR and the follower as FLW.

3.10.3 Expected test results

FLW does not accept incoming HB message if it does not pass the plausibility checks.

3.10.4 Criteria for evaluating results

The speed of FLW is not changed if it does pass the plausibility checks; the current speed of FLW remains unchanged as displayed as output in the AF3 Simulator view.

3.10.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat() . This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 25.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (25.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11. 0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)

- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The speed of FLW remains unchanged as shown in the output port **platoonStoredNew**.

3.10.6 Test Results

Check the output port **platoonStoredNew**. This port contains the adapted speed value (based on the received HB message by LDR) in the member **Velocity**. This is true because the received speed (labelled as **velocity**) did deviate in 30% w.r.t to the history.

Status: *PASSED.*

3.11 PMM_VCS-HPC.1_TC1

Test case to validate the maintenance of a heart-beat data history.

3.11.1 Security Requirements addressed

PMM_VCS-HPC.1

3.11.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.11.3 Expected test results

FLW stores relevant data from HB messages sent to LDR, including the speed and steering angle.

3.11.4 Criteria for evaluating results

The history of previous HB messages (sent to LDR) are displayed as output in the AF3 Simulator view.

3.11.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.

- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    -
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The stored information based on the received HB can be seen in the output port **historyNew**.

3.11.6 Test Results

Check the output port **historyNew**. This port contains now two entries with relevant information (e.g., velocity) from the received HB message sent by LDR.

Status: *PASSED*

3.12 PMM_VCS_HPC.2_TC1

Test case to validate that the TOE accepts HB messages consistent to the history.

3.12.1 Security Requirements addressed

PMM_VCS_HPC.2.1

3.12.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.12.3 Expected test results

FLW only updates the history if the speed or distance front value deviate less than 30% w.r.t. the average of the last values from the history. Otherwise, the incoming HB message from LDR is dropped and history is not updated.

3.12.4 Criteria for evaluating results

The history is not updated when any of the plausibility checks failed (i.e., deviate more than 30%). It means the variable **historyOld** is equal to **historyNew** displayed as input and output in the AF3 Simulator view, respectively.

3.12.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 15.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (15.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.

- The output of the simulation is displayed on the right-hand side of the Simulator view. The unchanged history can be seen in the output port **historyNew**.

3.12.6 Test Results

Check both the input port **historyOld** and the output port **historyNew**. The data stored in **historyOld** is equal to **historyNew**. That is, **historyNew** will contain the same data inserted as input (i.e., [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]]). This is true because the received speed from LDR deviates in more than 30% w.r.t. the average of the local history. As a result, the local history is not updated.

Status: *PASSED.*

3.13 PMM_VCS_HPC.2_TC2

Test case to validate that the TOE accepts HB messages consistent to the history.

3.13.1 Security Requirements addressed

PMM_VCS_HPC.2.1

3.13.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.13.3 Expected test results

FLW updates the history when the speed and distance front value does not deviate in more than 30% w.r.t. the average of the last values from the history.

3.13.4 Criteria for evaluating results

The history is not updated when any of the plausibility checks failed (i.e., deviate more than 30%). It means the variable **historyOld** is equal to **historyNew** displayed as input and output in the AF3 Simulator view, respectively.

3.13.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.

  - amount: 2 (amount of vehicle platoons)
  - id: 2 (unique identifier of FLW)
  - leaderID: 1 (unique identifier of LDR)
  - broadcastPlatoon:[
    {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
    {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

    (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
  - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).

- Define the inputs **Velocity** and **SteeringAngle**.
  - Velocity: 4.9 (speed of the vehicle)
  - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
  - DistanceFront: 10.0 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The changed history can be seen in the output port **historyNew**.

3.13.6 Test Results

Check both the input port **historyOld** and the output port **historyNew**. The data stored in historyNew is different from the date stored in historyOld. That is, historyNew will contain the following data ([[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9], [distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 5.0]]).

Status: *PASSED.*

3.14 PMM_VCS-HPC.3_TC1

Test case to validate that the TOE accepts HB messages (triggering EB) consistent to the sensor data history.

3.14.1 Security Requirements addressed

PMM_VCS-HPC.3.1

3.14.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.14.3 Expected test results

FLW does not trigger an EB if the sensor-based plausibility check fails.

3.14.4 Criteria for evaluating results

The variable EB is not set to true when the incoming HB (triggering an EB) does not pass the sensor-based plausibility check, as displayed in the output of the AF3 Simulator view.

3.14.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat() . This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 1 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 8.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (8.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9], [distanceFront: 9.8, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 2 entries).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 39.8 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. **EB** displays that FLW will not stop the vehicle.

3.14.6 Test Results

Check the output port **EB**. This port is set to false since the sensor-based plausibility check has failed, i.e., **DistanceFront** (i.e., 39.8) did deviate in 30% w.r.t. the history.

Status: *PASSED*

3.15 PMM_VCS-HPC.3_TC2

Test case to validate that the TOE accepts HB messages (triggering EB) consistent to the sensor data history.

3.15.1 Security Requirements addressed

PMM_VCS-HPC.3.1

3.15.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.15.3 Expected test results

FLW does trigger an EB if the sensor-based plausibility check does not fail.

3.15.4 Criteria for evaluating results

The variable EB is set to true when the incoming HB (triggering an EB) does passes the sensor-based plausibility check, as displayed in the output of the AF3 Simulator view.

3.15.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 1 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 8.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

(This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (8.0))

  - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9], [distanceFront: 9.8, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 2 entries).

- Define the inputs **Velocity** and **SteeringAngle**.
  - Velocity: 4.9 (speed of the vehicle)
  - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
  - DistanceFront: 9.8 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. **EB** displays that FLW will not stop the vehicle.

3.15.6 Test Results

Check the output port **EB**. This port is set to true since the sensor-based plausibility check has not failed, i.e., **DistanceFront** (i.e., 9.8) did not deviate in 30% w.r.t. the history.

Status: *PASSED*

3.16 PMM_VCS-SPC.1_TC1

Test case to validate the maintenance of a history of gaps to the vehicle in front measured by the sensors data history.

3.16.1 Security Requirements addressed

PMM_VCS-SPC.1.1

3.16.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.16.3 Expected test results

FLW stores previous received gap information from the sensor.

3.16.4 Criteria for evaluating results

The local history of FLW is updated with incoming sensor information. The new local history is displayed as output in the AF3 Simulator view.

3.16.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
      {distanceFront : 10.0, steeringAngle : 11.0, id : 2, position : 2, velocity : 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 9.8 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The result can be seen in the output port **historyNew**.

3.16.6 Test Results

Check the output port **historyNew**. This port contains a new entry with **DistanceFront** value **9.8**.

Status: *PASSED*

3.17 PMM_VCS-SPC.2-3-HPC.1-2_TC1

Test case to validate that the TOE accepts HB messages consistent to the sensor data history.

3.17.1 Security Requirements addressed

PMM_VCS-SPC.2
PMM_VCS-SPC.3
PMM_VCS-HPC.1
PMM_VCS-HPC.2

3.17.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.17.3 Expected test results

FLW does not update the speed of the vehicle if the sensor-based plausibility check fails.

3.17.4 Criteria for evaluating results

The speed of FLW is not updated after receiving a HB message from LDR that does not pass the sensor-based plausibility check. The speed of FLW remains unchanged as can be seen in the input and output of the AF3 Simulator view.

3.17.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project
- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 8.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (8.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.

  - Velocity: 4.9 (speed of the vehicle)
  - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
  - DistanceFront: 9.8 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The result in displayed in the output port **platoonStoredNew**.

3.17.6 Test Results

Check both the input port **platoonStoredOld** the output port **platoonStoredNew**. The data from these two ports are the same because the speed value received by LDR did not pass the sensor-based plausibility check.

Status: *PASSED*

3.18 PMM_VCS-SPC.2-3-HPC.1-2_TC2

Test case to validate that the TOE accepts HB messages consistent to the sensor data history.

3.18.1 Security Requirements addressed

PMM_VCS-SPC.2
PMM_VCS-SPC.3
PMM_VCS-HPC.1
PMM_VCS-HPC.2

3.18.2 Test preconditions

The leader of the platoon and 1 follower shall be defined. We refer to the leader as LDR and the follower as FLW.

3.18.3 Expected test results

FLW does update the speed of the vehicle if the sensor-based plausibility check does not fail.

3.18.4 Criteria for evaluating results

The speed of FLW is updated after receiving a HB message from LDR since it does pass the sensor-based plausibility check. The speed of FLW is adapted as displayed in the AF3 Simulator view.

3.18.5 Test Procedure

The following steps will be carried out:

- Open AF3 and import the modelled CACC project

- Set the platoon state for “Follower” by clicking on the Platoon States component and selecting the “Follower” state
- Open the AF3 Simulator view for the Platoon Management component, and click on Platoon Info
- Define the input **receiveMessageType**.
    - receiveMessageType: heartbeat(). This means that the received message is a HB message from LDR.
- Define the input **platoonInfoIn** that includes the following. Note that the numbers provided below are examples only.
    - amount: 2 (amount of vehicle platoons)
    - id: 2 (unique identifier of FLW)
    - leaderID: 1 (unique identifier of LDR)
    - broadcastPlatoon:[
      {distanceFront : 0.0, steeringAngle : 11.0, id : 1, position : 1, velocity : 5.0},
      {distanceFront: 10.0, steeringAngle: 11.0, id: 2, position: 2, velocity: 4.9}]

      (This variable contains the information broadcasted by LDR to FLW, including the speed of LDR (5.0))
    - History: [[distanceFront: 10.0, steeringAngle 11.0, id: 1, position: 1, velocity: 4.9]] (history is a list of previous data from the current vehicle (in this case FLW). Note that here history only contains 1 entry).
- Define the inputs **Velocity** and **SteeringAngle**.
    - Velocity: 4.9 (speed of the vehicle)
    - SteeringAngle: 11.0 (steering angle of the vehicle)
- Define the input **DistanceFront**.
    - DistanceFront: 9.8 (distance to the preceding vehicle)
- Click on “Hold” for the defined inputs, i.e., **receiveMessageType, platoonInfoIn, Velocity, SteeringAngle** and **DistanceFront.** This ensures that such values are kept throughout the simulation.
- Click on the yellow arrow on the left-hand side to execute one simulation step.
- The output of the simulation is displayed on the right-hand side of the Simulator view. The result in displayed in the output port **platoonStoredNew**.

3.18.6 Test Results

Check both the input port **platoonStoredOld** the output port **platoonStoredNew**. The data from these two ports are different because the velocity of FLW has been adapted (see new velocity in **platoonStoredNew)**. This is true because the incoming HB message (incl. velocity of LDR) did pass the sensor-based plausibility check.

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.

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

Test ID	Requirement code	Results (including section reference)	Notes
PMM_IF.1_TC1	PMM_IF.1	PASSED WITH DEVIATIONS (3.1.6)	GPS, timestamp, and digitally signed certificates are not modelled
PMM_IF.1_TC2	PMM_IF.1	PASSED WITH DEVIATIONS (3.2.6)	GPS, timestamp, and digitally signed certificates are not modelled
PMM_IF.2_TC1	PMM_IF.2	PASSED WITH DEVIATIONS (3.3.6)	GPS, timestamp, and digitally signed certificates are not modelled
PMM_IF.3_TC1	PMM_IF.3	PASSED WITH DEVIATIONS (3.4.6)	Timestamp and digitally signed certificates are not modelled.
PMM_IF.4_TC1	PMM_IF.4	PASSED WITH DEVIATIONS (3.5.6)	GPS and timestamp are not modelled
PMM_IF.4_TC2	PMM_IF.4	PASSED WITH DEVIATIONS (3.6.6)	GPS and timestamp are not modelled
PMM_IF.5_TC1	PMM_IF.5	PASSED WITH DEVIATIONS (3.7.6)	Distance to the edges of the lane not modelled
PMM_IF.6_TC1	PMM_IF.6	PASSED (3.8.6)	--
PMM_PC.1-2_TC1	PMM_PC.1 PMM_PC.2	PASSED (3.9.6)	--
PMM_PC.1-2_TC2	PMM_PC.1 PMM_PC.2	PASSED (3.9.6)	--
PMM_VCS-HPC.1_TC1	PMM_VCS-HPC.1	PASSED (3.11.6)	--
PMM_VCS-HPC.2_TC1	PMM_VCS-HPC.2	PASSED (3.12.6)	--
PMM_VCS-HPC.2_TC2	PMM_VCS-HPC.2	PASSED (3.12.6)	--
PMM_VCS-HPC.3_TC1	PMM_VCS-HPC3.	PASSED (3.14.6)	--
PMM_VCS-HPC.3_TC2	PMM_VCS-HPC3.	PASSED (3.15.6)	--
PMM_VCS-SPC.1_TC1	PMM_VCS-SPC.1	PASSED (3.16.6)	--

Test ID	Requirement code	Results (including section reference)	Notes
PMM_VCS-SPC.2-3-HPC.1-2_TC1	PMM_VCS-SPC.2 PMM_VCS-SPC.3 PMM_VCS-HPC.1 PMM_VCS-HPC.2	PASSED (3.17.6)	--
PMM_VCS-SPC.2-3-HPC.1-2_TC2	PMM_VCS-SPC.2 PMM_VCS-SPC.3 PMM_VCS-HPC.1 PMM_VCS-HPC.2	PASSED (3.18.6)	--

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_IF.1	PMM_IF.1_TC1,	PASSED WITH DEVIATIONS (3.1.6)	GPS, timestamp, and digitally signed certificates are not modelled
	PMM_IF.1_TC2	PASSED WITH DEVIATIONS (3.2.6)	GPS, timestamp, and digitally signed certificates are not modelled
PMM_IF.2	PMM_IF.2_TC1	PASSED WITH DEVIATIONS (3.3.6)	GPS, timestamp, and digitally signed certificates are not modelled
PMM_IF.3	PMM_IF.3_TC1	PASSED WITH DEVIATIONS (3.4.6)	Timestamp and digitally signed certificates are not modelled.
PMM_IF.4	PMM_IF.4_TC1	PASSED WITH DEVIATIONS (3.5.6)	GPS and timestamp are not modelled
	PMM_IF.4_TC2	PASSED WITH DEVIATIONS (3.6.6)	GPS and timestamp are not modelled
PMM_IF.5	PMM_IF.5_TC1	PASSED WITH DEVIATIONS (3.7.6)	Distance to the edges of the lane not modelled
PMM_IF.6	PMM_IF.6_TC1	PASSED (3.8.6)	--
PMM_PC.1	PMM_PC.1-2_TC1	PASSED (3.9.6)	--

Requirement code	Test ID	Results (including section reference)	Notes
	PMM_PC.1-2_TC2	PASSED (3.9.6)	--
PMM_PC.2	PMM_PC.1-2_TC1	PASSED (3.9.6)	--
	PMM_PC.1-2_TC2	PASSED (3.9.6)	--
PMM_VCS-HPC.1	PMM_VCS-HPC.1_TC1	PASSED (3.11.6)	--
	PMM_VCS-SPC.2-3-HPC.1-2_TC1	PASSED (3.17.6)	--
	PMM_VCS-SPC.2-3-HPC.1-2_TC2	PASSED (3.18.6)	--
PMM_VCS-HPC.2	PMM_VCS-HPC.2_TC1	PASSED (3.12.6)	--
	PMM_VCS-HPC.2_TC2	PASSED (3.12.6)	--
	PMM_VCS-SPC.2-3-HPC.1-2_TC1	PASSED (3.17.6)	--
	PMM_VCS-SPC.2-3-HPC.1-2_TC2	PASSED (3.18.6)	--
PMM_VCS-HPC.3	PMM_VCS-HPC.3_TC1	PASSED (3.14.6)	--
	PMM_VCS-HPC.3_TC2	PASSED (3.15.6)	--
PMM_VCS-SPC.1	PMM_VCS-SPC.1_TC1	PASSED (3.16.6)	--
PMM_VCS-SPC.2	PMM_VCS-SPC.2-3-HPC.1-2_TC1	PASSED (3.17.6)	--
	PMM_VCS-SPC.2-3-HPC.1-2_TC2	PASSED (3.18.6)	--
PMM_VCS-SPC.3	PMM_VCS-SPC.2-3-HPC.1-2_TC1	PASSED (3.17.6)	--
	PMM_VCS-SPC.2-3-HPC.1-2_TC2	PASSED (3.18.6)	--

Table 3: Test Summary Coverage (Requirements vs Tests)

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

	PMM_IF.1	PMM_IF.2	PMM_IF.3	PMM_IF.4	PMM_IF.5	PMM_IF.6	PMM_PC.1	PMM_PC.2	PMM_VCS-HPC.1	PMM_VCS-HPC.2	PMM_VCS-HPC.3	PMM_VCS-SPC.1	PMM_VCS-SPC.2	PMM_VCS-SPC.3
PMM_IF.1_TC1	X													
PMM_IF.1_TC2	X													
PMM_IF.2_TC1		X												
PMM_IF.3_TC1			X											
PMM_IF.4_TC1				X										
PMM_IF.4_TC2				X										
PMM_IF.5_TC1					X									
PMM_IF.6_TC1						X								
PMM_PC.1-2_TC1							X	X						
PMM_PC.1-2_TC2							X	X						
PMM_VCS-HPC.1_TC1									X					
PMM_VCS-HPC.2_TC1										X				
PMM_VCS-HPC.2_TC2										X				
PMM_VCS-HPC.3_TC1											X			
PMM_VCS-HPC.3_TC2											X			
PMM_VCS-SPC.1_TC1												X		
PMM_VCS-SPC.2-3-HPC.1-2_TC1									X	X			X	X
PMM_VCS-SPC.2-3-HPC.1-2_TC2									X	X			X	X

Table 4: Matrix of test coverage

Chapter 5 List of Abbreviations

Abbreviation	Translation
ACC	Adaptive Cruise Control
CACC	Cooperative Adaptive Cruise Control
HB	Heartbeat
PMM	Platoon Management Module
TC	Test Case
TOE	Target Of Evaluation
TSF	TOE Security Functionality
VCM	Vehicle Communication System
VCS	Vehicle Control Module
LDR	Leader (leader of the platoon)
FLW	Follower (a follower in the platoon)
AF3	AutoFOCUS3

Chapter 6 Bibliography

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

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

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

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

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

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

[7] FORTISS GmbH. AutoFOCUS 2.19. Available at *https://www.fortiss.org/en/publications/software/autofocus-3*

[8] CACC model designed in AutoFOCUS3. Available in the Sparta repository under 03-WPs/WP5-Program-2-CAPE/T52_Convergence_Security_Safety/implementations/platooning/CACC-platoon_final_Version_18.06.af3_23