D6.3
Final Demonstration and Evaluation

Topic	H2020-SU-ICT-2018-2020
Project Title	Artificial Intelligence-based Cybersecurity for Connected and Automated Vehicles
Project Number	833611
Project Acronym	CARAMEL
Contractual Delivery Date	M30
Actual Delivery Date	M33
Contributing WP	WP6
Project Start Date	01/10/2019
Project Duration	33 Months
Dissemination Level	Public
Editors	Daniel Baños Castillo (NEXTIUM by Idneo) Natàlia Porras Mateu (NEXTIUM by Idneo)
Contributors	I2CAT, ATOS, PANASONIC, UBIWHERE, T-SYSTEMS, UCY, UPAT, AVL, 0 INFINITY, CyberLens, GREENFLUX, SIDROCO, Capgemini Engineering, NEXTIUM by Idneo

Document History		
Version	Date	Remarks
0.1	21/04/2021	Initial Document Structure, Table of Contents.
0.2	16/06/2022	Partners contributions received.
1.0	17/06/2022	Draft sent to SAB.
1.1	30/06/2022	Draft with feedback from internal reviewers.
2.0	01/07/2022	Final version for EC submission.

Contributors		
Name	**Organisation**	**Type of contribution**
Daniel Baños Castillo	NEXTIUM by Idneo	Editor, Contributor
Natàlia Porras Mateu	NEXTIUM by Idneo	Editor, Contributor, Internal Reviewer
Dimitrios Margounakis	SIDROCO (SID)	Contributor, Internal Reviewer
Eleftherios Fountoukidis	SIDROCO (SID)	Contributor, Internal Reviewer
Petros Kapsalas	PANASONIC (PANA)	Contributor, Internal Reviewer
Rita Santiago	UBIWHERE	Contributor
Nicola Durante	ATOS	Contributor
Jordi Casademont	i2CAT Foundation (i2CAT)	Contributor
Bruno Cordero	i2CAT Foundation (i2CAT)	Contributor
Jordi Marias	i2CAT Foundation (i2CAT)	Contributor
Josep Escrig	i2CAT Foundation (i2CAT)	Contributor
Adrián Pino	i2CAT Foundation (i2CAT)	Contributor
Sergi Mercadé Laborda	i2CAT Foundation (i2CAT)	Contributor
Sergi Sánchez Deutsch	i2CAT Foundation (i2CAT)	Contributor
Anish Khadka	0 INFINITY (0INF)	Contributor
Achilleas Sesis	0 INFINITY (0INF)	Contributor
Gordon Johnson	0 INFINITY (0INF)	Contributor
Tomaz Roskar	AVL	Contributor
Bob Elders	GREENFLUX (GFX)	Contributor
Christos Kyrkou	University of Cyprus (UCY)	Contributor
Christos Laoudias	University of Cyprus (UCY)	Contributor
Nikos Argyropoulos	CyberLens B.V. (CLS)	Contributor

Irene Karapistoli	CyberLens B.V. (CLS)	Contributor
Anastasios Lytos	SIDROCO (SID)	Contributor
Elisavet Grigoriou	SIDROCO (SID)	Contributor
Theocharis Saoulidis	SIDROCO (SID)	Contributor
Apostolos Tsiakalos	SIDROCO (SID)	Contributor
Achilleas Moukoulis	SIDROCO (SID)	Contributor
Andreas Kloukiniotis	University of Patras (UPAT)	Contributor
Nikos Piperigkos	University of Patras (UPAT)	Contributor
Georgios Chatziioannou	Capgemini Engineering	Contributor
Philippos Barabas	Capgemini Engineering	Contributor
Eduardo Cervantes	Capgemini Engineering	Contributor
Duong van Nguyen	PANASONIC (PANA)	Contributor
Peter Hofmann	DEUTSCHE TELEKOM SECURITY GMBH (DT-Sec)	Contributor

DISCLAIMER OF WARRANTIES

This document has been prepared by CARAMEL project partners as an account of work carried out within the framework of the contract no 833611.

Neither Project Coordinator, nor any signatory party of CARAMEL Project Consortium Agreement, nor any person acting on behalf of any of them:

- makes any warranty or representation whatsoever, express or implied,
  - with respect to the use of any information, apparatus, method, process, or similar item disclosed in this document, including merchantability and fitness for a particular purpose, or
  - that such use does not infringe on or interfere with privately owned rights, including any party's intellectual property, or
- that this document is suitable to any particular user's circumstance; or
- assumes responsibility for any damages or other liability whatsoever (including any consequential damages, even if Project Coordinator or any representative of a signatory party of the CARAMEL Project Consortium Agreement, has been advised of the possibility of such damages) resulting from your selection or use of this document or any information, apparatus, method, process, or similar item disclosed in this document.

CARAMEL has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 833611. The content of this deliverable does not reflect the official opinion of the European Union. Responsibility for the information and views expressed in the deliverable lies entirely with the author(s).

DISCLOSURE STATEMENT

"The following document has been reviewed by the CARAMEL External Security Advisory Board as well as the Ethics and Data Management Committee of the project. Hereby, it is confirmed that it does not contain any sensitive security, ethical, or data privacy issues."

Table of Contents

List of Figures

List of Tables

List of Acronyms

2D	Two-dimensional
3D	Three-dimensional
AD / AHD	Anti-hacking Device
ADAS	Advanced driver assistance systems
ADMM	Alternating Direction Method of Multiplies
AI	Artificial Intelligence
API	Application Programming Interface
ASPP	Atrous Spatial Pyramid Pooling
AT	Authorization Ticket
ATOS	ATOS is a Consortium's partner
BTP	Basic Transport Protocol
CAM	Cooperative Awareness Message
CAN	Controller Area Network
CARLA	Car Learning to Act (an open simulator for urban driving)
CAV	Connected and Autonomous Vehicle
CDF	Cumulative Distribution Function
CLL	Centralized Laplacian Localization
CNN	Convolutional Neural Networks
COM	Communications
CPI	Charging point infrastructure
CPO	Charging point operator
CPU	Central Processing Unit
CRD	Certification Revocation Decision
CRL	Certification Revocation List
C-V2X	Cellular V2X
DENM	Decentralized Event Notification Message
DGPS	Differential GPS
DNN	Deep Neural Network
DSO	Distribution System Operator
EA	Enrolment Authority
EC	Enrolment Certificate
ECU	Engine Control Unit
eMSP	electroMobility Service Provider
eNB	evolved NodeB
ESOP	Electromobility Services and Operations Provider
ETSI	European Telecommunications Standards Institute
EV	Electric Vehicle
FDI attack	False Data Injection attack
FW	Frank-Wolfe
GAN	Generative Adversarial Networks
GDPR	General Data Protection Regulation
GFX	GreenFlux (Consortium's partner)

GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Unit
HSM	Hardware Security Module
HUD	Head-up display
HW	Hardware
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial Measurement Unit
IP	Internet Protocol
ITS	Intelligent Transportation System
KITTI	KITTI Vision Benchmark Suite by Karlsruhe Institute of Technology and Toyota Technological Institute
LC	Local Controller
LDM	Local Dynamic Map
LIDAR	Light Detection and Ranging - remote sensing technology
LSTM	Long Short-Term Memory
LTE	Long Term Evolution
MAC	Media Access Control
MAP	ML/AI Algorithm Provider
MEC	Multi-access Edge Computing
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
NR-PC5	New Radio - Power Commander 5
OBU	On-Board Unit
ODB	On-Board Diagnostics
OGM	Occupancy Grid Map
OS	Operative System
OTA	Over-the-Air
PASEU	Panasonic Automotive Systems Europe
PC	Personal Computer
PGD	Projected Gradient Descent
PKI	Public Key Infrastructure
POE	Power Over Ethernet
PSPNet	Pyramid Scene Parsing Network
RAM	Random-Access Memory
R-CLL	Robust Centralized Laplacian Localization
R-CNN	Region-based Convolutional Neural Network
RMSE	Root Mean Square Error
RNN	Recurrent Neural Networks
RoI	Region of Interest
ROP	Return-Oriented Programming
ROS	Robot Operating System
RSU	RoadSide Unit
RTK	Real-Time Kinematic

SDS	Small Displacement Spoofing (SDS)
SIEP	Security Information Exchange Protocol
SMP	Security Message Protocol
SOC	Security Operation Center
SSD	Single Shot Detector
SW	Software
TCN	Temporal Convolutional Network
UC	Use Case
V2I	Vehicle-to-Infrastructure
V2I2V	Vehicle-to-Infrastructure-to-Vehicle
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
vEPC	virtual Evolved Packet Core
VLAN	Virtual Local Area Network
WIFI	Wireless Fidelity
WP	Work Package

Executive Summary

CARAMEL 'Artificial Intelligence based cybersecurity for connected and automated vehicles'

H2020 CARAMEL project comes to an end after a journey of 33 months where all the partners involved worked together with the main same objective, the development of a more secure driving experience for the connected and automated vehicles.

CARAMEL's goal was - and is - to proactively address modern vehicle cybersecurity challenges by applying advanced Artificial Intelligence (AI) and ML techniques, and to continuously seek methods to mitigate associated safety risks. So, CARAMEL intended to follow an automotive cybersecurity layered approach that deals with attestation of vehicular hardware, software, and network infrastructures.

From the beginning, CARAMEL was explained using the three innovation pillars (family of use cases) which represents:

Pillar 1: The autonomous vehicle.

Pillar 2: The connected vehicle.

Pillar 3: The plug-in electrical vehicle.

With this, the objective of this deliverable is to evaluate the CARAMEL solutions effectiveness developed, using different use cases scenarios as "actions" to represent possible attacks chosen by each partner according to their developments and expertise areas.

These use cases have been selected from the deliverable *"D2.4_ System Specification and Architecture "*, - where the use cases were introduced as possible keys for CARAMEL within the WP2 of the project- and have been performed in a real / test environment or in a simulation test.

The final demonstration took place in the PANASONIC Automotive premises in Langen (Germany) on June 20th-21st, and the results and observations extracted from there, can be found within this deliverable, inside each partner's contributions following the structure of the three innovation pillars.

In addition, it can be read at the end of the deliverable, a Roadmap for future evolution of the CARAMEL achievements, opening the door to see "where CARAMEL can/could be directed to" and a Conclusions chapter, with the aim to summarize all the important notes and lessons learned detected during the performance of the project.

1 Introduction and Project Overview

As already introduced in previous deliverables, the Cooperative Connected and Automated Mobility (CCAM) is increasing nowadays and with the idea to be the future of the automotive industry.

Due to this and to its high technology and connectivity dependance, protections against cyberattacks are in the scope of the industry to develop solutions as a top-level protection to avoid any kind of access to the autonomous and connected vehicles (attacks which could be both physical and remote).

CARAMEL's goal is to proactively address modern vehicle cybersecurity challenges through applying advanced Artificial Intelligence (AI) and Machine Learning (ML) techniques, and to continuously seek methods to mitigate associated safety risks.

1.1 *Document Scope*

Deliverable D6.3 will report on the description and assessment of scenario driven attacks, through the implemented use cases. It will also contain an overall assessment of the project results, as well as a roadmap for the future evolution of the CARAMEL achievements.

The objective of this document is to evaluate the CARAMEL solution, as per the specifications of Work Package 2 (WP2) specified use case. The scenarios described in every Use Case (UC) in WP2 will be implemented accomplishing end-to-end communication. Technology providers, integrators, and end-users will get engaged in the task ensuring the successful deployment and reaction of the system.

More specifically:

- **UC1** - **Autonomous Mobility**, the physical adversarial attacks, and the attack on the camera sensor scenarios will be deployed testing the capabilities and the limits of the autonomous mobility.
- **UC2** – **Connected Mobility**, the location spoofing attack, the attack on the V2X message transmission. and the tamper attack on the vehicle's OBU will be executed revealing the resilience of the system in terms of communication.
- **UC3** – **Electromobility**, the smart charging abuse, and the EV scheduling abuse scenarios will be implemented by checking CARAMEL's system reactions while analyzing potential threats in the power system.

1.2 *Scenarios and Actors Overviews*

Table 1 and Table 2 provide an overview of all the use cases and actors identified in CARAMEL's deliverable D2.4 (under the title *"System Specification and Architecture"*) delivered in WP2 (Work Package 2):

ID	Use case name	Description
UC1.1	Detection of physical attacks on traffic signs	The scenario deals with two kinds of attack: attacker vandalizes traffic signs (i.e., some random graffiti that hides a different part of the sign or a coordinated attack such as generating ML based image to cover the signs. The autonomous vehicle moves in the test area. Certain traffic signs have been physically modified in order to influence the driving behaviour and planning of the autonomous vehicle. CARAMEL's platform is operating in parallel to the driving system of the autonomous vehicle without influencing the decision-making module. When the vision-related sensor and the ML components of CARAMEL detect a physical attack, a corresponding notification will be displayed to the vehicle operator or passenger.
UC1.1.1	Simulated detection of attacks on traffic signs	Show and test the scenario "Detection of Attacks on Traffic Signs" in a simulated environment.
UC1.1.1.1	Creation of defaced traffic signs	Creation of defaced traffic signs.

UC1.1.1.2	Creation of labeled training data from simulation	Creation of labelled training data from simulation.
UC1.1.1.3	Creation of model from training data	Creation of model from training data.
UC1.1.1.4	Test of model in simulation environment	Test of model in simulation environment.
UC1.1.1.5	Signaling of attack events	Signaling of attack events (defaced traffic sign encountered).
UC1.2	Adversarial attacks on camera sensor	The autonomous vehicle is expected to drive from a starting location to a given destination following a specified path. At a given time instance the image of the vehicle will be tampered through a specific perturbation intended to cause the perception module to misbehave (e.g., either detect objects that are not truly present or hide objects that are within the field of view). The autonomous vehicle is expected to drive from a starting location to a given destination following a specified path. At a given time instance the image of the vehicle will be tampered through a specific perturbation intended to cause the perception module to misbehave (e.g., either detect objects that are not truly present or hide objects that are within the field of view).
UC1.2.1	Simulated adversarial attacks on camera sensor	Show and test the scenario “Adversarial Attacks on Camera Sensor” in a simulated environment.
UC1.2.1.1	Design of AI and ML-based attacks	Design of AI and ML-based attacks.
UC1.2.1.2	Design of AI and ML-based attack detectors	Design of AI and ML-based attack detectors.
UC1.2.1.3	Design of multi-modal fusion-based attack detectors	Design of multi-modal fusion-based attack detectors.
UC1.2.1.4	Test of model in simulation environment	Test of model in simulation environment (eg. CARLA simulator with attached anti-hacking device).
UC1.2.1.5	Design and setup of simulation environment	Design and setup of simulation environment (CARLA simulator and anti-hacking device).
UC1.2.2	Real-world demonstration of adversarial attacks on camera sensor	Real-world demonstration of adversarial attacks on camera sensor.
UC1.2.2.1	Transfer of multi-model fusion model to real car	Transfer of multi-modal fusion model to real car.
UC1.2.2.2	Test of model with real car in test environment	Test of model with real car in test environment.
UC2.1	Location spoofing attack	Using SDR hardware, the attacker is able to spoof GPS satellite signals. The vehicle relies on a second location stream to identify a possible GPS location spoofing attack, based on vehicle’s movement description, IMU and GPS-free localization measurements.
UC2.1.1	Simulated location spoofing attack	Simulated location spoofing attack.
UC2.1.1.1	Setup of simulation environment	Setup of simulation environment.
UC2.1.1.2	Creation of labeled training data	Creation of labelled training data.
UC2.1.1.3	Design and implement real-time fusion of multiple sensors to detect attack	Design and implement real-time fusion of multiple sensors to detect attack.
UC2.1.1.4	Test model in simulation environment	Test of model in simulation environment.
UC2.1.2	Location spoofing attack with real car	Location spoofing attack with real car.
UC2.1.3	GPS spoofing alert to MEC	Send GPS spoofing alert to MEC.
UC2.2	Attack on the V2X message transmission	There are two different kinds of attacks: a) A malicious attacker transmits fake CAM and DENM messages

		and b) a malicious attacker tries to track a specific vehicle.
UC2.2.1	Generation of fake messages	Generation of fake messages.
UC2.2.2	Sniff and Replay Messages of Compliant Vehicles	Detection of replayed messages.
UC2.2.3	Messages sent by a compliant vehicle with a fraudulent identity	Detection of messages with faked/fraudulent identity.
UC2.2.4	Tracking of vehicles by sniffing sent messages	The anti-hacking device generates new Authorization Tickets (AT) such that an attack on the privacy of the driver of the cooperative vehicle is not successful.
UC2.2.5	Demonstration of attack on V2X message transmission	The attacks on the V2X message transmission are performed in a real car, the anti-hacking device detects these attacks.
UC2.2.6	Fake message detection alert to MEC	Alerts about the detection of fake messages are relayed to the MEC or detected by the MEC.
UC2.2.7	Integration of Anti-Hacking Device	Integration of anti-hacking device into test car.
UC2.2.8	Regulatory Approval for Mobile Network Use	For the setup of the LTE small cells regulatory approval is needed by the German Agency for networks (RegTP).
UC2.2.9	Setup of MEC Devices	Setup of MEC devices for test and development.
UC2.3	Tamper attack on vehicle's OBU	The attacker is able to get physical access to an OBU by accessing the car. The attacker could also have acquired another OBU (e.g., aftermarket sample) in order to study potential vulnerabilities beforehand.
UC2.3.1	Physical attack on OBU	Attacker performs physical attacks on the OBU.
UC2.3.2	HW tamper detection	The OBU detects physical attacks on its hardware.
UC2.3.3	Tamper alert to MEC	The MEC is alerted by the OBU about the physical attack.
UC2.4	Certificate Revocation	Revoke vehicle certificate.
UC2.4.1	Revocation of Vehicle Certificates	Revoke vehicle certificate.
UC2.4.2	Publication of Certificate Revocation Lists	Publication of certificate revocation lists.
UC3.1	Smart charging abuse	The attacker(s) occupy (physically or remotely) the available charging stations starting and proceed timely in connection/disconnection actions creating an enormous load to the electric grid.
UC3.1.1	Abuse detection model creation	Abuse detection model creation for the smart charging abuse scenario.
UC3.1.2	Attack detection demonstration	In this use the smart charging abuse attack detection is demonstrated.
UC3.1.3	Anonymization of raw data	The original CDR data is anonymized for privacy reasons and GDPR compliance.
UC3.1.4	Preprocessing of data	The data is pre-processed to removed outliers and perform other necessary steps before ML training.
UC3.1.5	Selection of ML algorithms and parameters	The algorithms and parameters for successful training of the model are selected.
UC3.1.6	Application of abuse detection model	The abuse detection model is applied in the anti-hacking device (or service).
UC3.1.7	Perform attack	The attacker performs the smart charging abuse attack.
UC3.1.8	Setup of charging stations	Setup the charging stations for use in development and testing.
UC3.1.9	Signaling of attack events	Attack events are signalled to the SOC/Backend.
UC3.2	EV scheduling abuse	With proper coordination scheme, EV loads can be controlled to minimize charging costs. Attacker(s) occupy (physically or remotely) the available charging stations. Uncoordinated charging of even a 10% penetration of EV loads will notably affect power system operation, giving rise to voltage magnitude fluctuations and unacceptable load peaks.
UC3.2.1	Attack detection demonstration	In this use the EV scheduling abuse attack detection is demonstrated.

UC3.2.2	Application of Frank-Wolf algorithms	Application of the Frank-Wolf algorithm in the anti-hacking device.
UC3.2.3	Perform Attack	The attacker performs the smart charging abuse attack.
UC3.2.4	Setup of charging station	Setup the charging stations for use in development and testing.
UC3.2.5	Signaling of attack events	Attack events are signalled to the SOC/Backend.

Table 1: Use case overview.

Name	Stereotype	Description
Vehicle operator	Person	A person responsible to operate the vehicle in the case of a not fully automated one, monitoring the environment and the vehicle behavior. They are responsible for receiving notifications from the CARAMEL platform and taking the necessary actions to react to attacks.
Passenger	Person	A passenger of the car not responsible for any interaction with the car's steering.
Attacker	Person	Bad actor intent on disturbing the normal operation of vehicles or the traffic or charging infrastructure in general (such as road side units, charging backend infrastructure, chargers). The attacker might also physically attack objects such as traffic signs or lane markers in order to cause autonomous or semi-autonomous driving systems to malfunction.
Data scientist	Person	Engineer trained creating and testing machine learning models from training data.
Simulation designer	Person	Designer capable of creating or changing virtual worlds and artifacts (such as traffic signs) in a simulation environment.
Simulation operator	Person	Engineer able to set up, maintain, and operate a simulation environment.
Automotive engineer	Person	The automotive engineer is able to integrate physical components such as sensors or the anti-hacking device into a car to create a demonstration and test setup. In addition, the automotive engineer can set up and configure other required infrastructure in a test, e.g. MEC, LTE, WIFI, Charging Point, etc.
Cooperative car	System	V2X communication-enabled car. This type of car is able to communicate with other cars and infrastructure to transmit relevant data such as position, speed, etc.
Fixed infrastructure	System	Infrastructure to support connected driving, consisting of eNB (C-V2X base stations), RSU (IEEE uo2.11p fixed station), PKI servers, or MEC (multi-access edge computing node).
Outside infrastructure	System	Infrastructure such as public parking, a workshop, or private parking not specifically set up for the project.
MEC	System	Multi-access edge computing device - the Multi-access Edge Computing (MEC) server will be deployed to accommodate the required functions to run at the edge of the network, following, as much as possible, the ETSI MEC framework standardization.
Anti-hacking device	System	The anti-hacking device is a physical controller that is integrated into the car and acts as an attack detection device. In the Autonomous Mobility scenario its task is to run pre-trained ML models that work on the sensor data to detect anomalies that might point to malicious attacks. Additionally, the anti- hacking solution might be used for different functions in the context of the CARAMEL project, i.e. if needed it can ensure security for an embedded application platform. In this case, the software layer of the solution might be employed only.

PKI infrastructure	System	The PKI infrastructure is the enabler to provide security to V2X message transmissions and will be the basis to build and test the “Attack on the V2X message transmission” scenario.
Charging point infrastructure (CPI)	System	Charge Points are devices where EVs get charged. Each CP contains at least one meter per socket (MID meter) owned and controlled by the CPO. This CPO meter is connected to the energy socket through which the EV gets charged and is used to measure the energy consumed by the EV. Each CP also includes a local controller (LC) with a connection (e.g.: GPRS or wire connection) to the back-office of the CPO. Among other things (e.g.: remote updates), such a connection is used to authenticate the customer (EV owner) at the CPO.
Charging point operator (CPO)	System	The CPO is responsible for the management, maintenance and operation of the charging stations (both technical and administrative). The role of CPO can be segmented into: 1. Responsibility for administrative operation (e.g. access, roaming, billing to eMSP etc.) and 2. Responsibility for technical maintenance, which is often done by the manufacturer. CPOs play a very important role in the EV market as they are responsible for bridging the gap between the entities
eMSP	System	An eMSP (electroMobility Service Provider) is a market role that offers charging services to EV drivers. An eMSP provides value by enabling access to a variety of charge points around a geographic area, usually in the form of a charge card. This means the EMSP is responsible to set up contracts with customers (owners of EV cars) and for managing customer information and billing.
Electric vehicle (EV)	System	Electric vehicle that is charged using the charging points.
Distribution system operator (DSO)	System	The distribution system operator (DSO) manages the electrical grid. The DSO does not produce any electric power but does however ensure that it is transported from the power station to the place where it is needed. The most important task of the DSO is to maintain a stable, reliable and well-functioning electricity network.
ESOP	System	Electromobility Services and Operations Provider – GreenFlux in the project. GreenFlux provides a white-label cloud-based SaaS Service and Operations Platform which allows both CPOs and eMSP to run their EV charging business.
Simulation Environment	System	CARLA or AVL ModelConnect/VTD simulation environment.
Onboard unit (OBU)	System	The onboard unit is a microcontroller built into the vehicle that performs functions to support the operation (steering etc.) of the car as well as the communication with the outside world via V2X protocols. A typical may contain many OBUs with different functions that are connected via local communication links (eg. the CAN bus).
Hardware Security Module (HSM)	System	Hardware element brought into a microcontroller to support the detection and avoidance of manipulation of the integrity of the device by providing or enabling functions such as tamper detection, secure boot, etc.
SOC/Backend	System	Security Operation Center (SOC) or Backend run by the CARAMEL project to visualize the overall threat situation during the demos or trials.
MAP	System	ML/AI Algorithms provider (e.g., UPAT in the context of the project)
Project Administrator	Person	Person responsible for administrative tasks in the CARAMEL project.

Table 2: Actors overview.

2 Pillar 1 scenario driven attacks

In Caramel, we investigate the potential and limitations of the Cyber-attack detection and mitigation engine via investigating its efficiency in restoring the attacks either on sensor layer (models developed on WP3) or at perception layers (models developed in WP4).

2.1 *Physical Adversarial Attack on Traffic Signs (Physical Layers):*

As described in D2.1 deliverable, the physical adversarial attack use case deals with the attack on the external environment of Connected and Autonomous Vehicles (CAVs). The aim of the use case is to demonstrate the effect of the physical adversarial attack in the environment and mitigation strategies for such attacks. The use case involves the attacks which can be easily performed in the real-world without direct access to the CAVs system and only minimal information about the CAVs is required for such attacks. In this case, it is assumed that CAVs use a perception engine to navigate the environment. Likewise, we choose the traffic signs as our attack point to demonstrate the use case. Since the traffic signs are a crucial aspect of traffic management, misreading the traffic signs can result in devastating outcomes. Hence, the use case provides realistic real-world scenarios. The overview of the use case is shown in Figure 1.

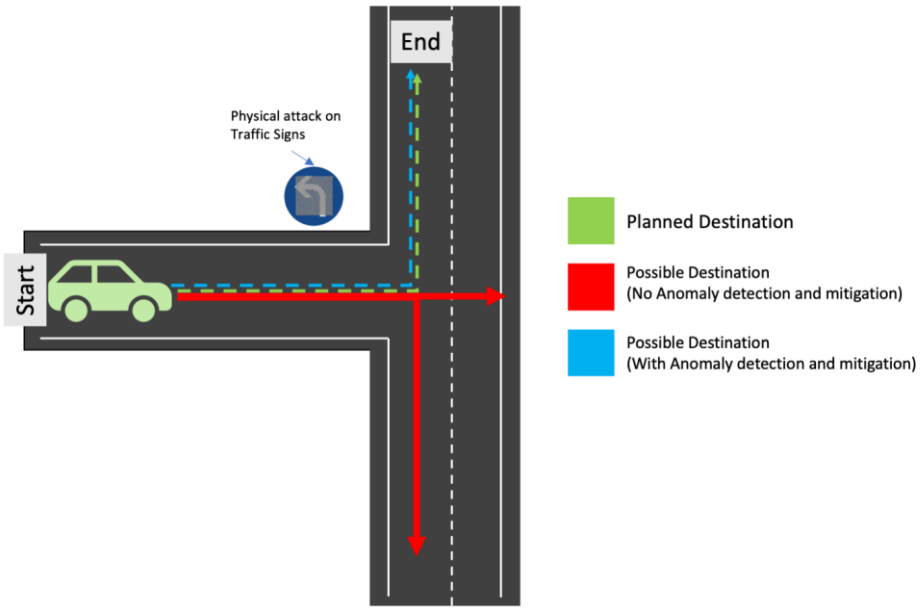


Figure 1: The overview of the Use case - Physical Adversarial Attack on traffic signs.

The use deals with two types of scenarios as described in D2.1 (Table 3):

Title	Description	Solution Developed for CARAMEL
Detection and reaction to physical attacks on traffic signs	The use case explores the state-of-the-art anomaly detection approaches to develop traffic signs anomaly detection.	Based on the various state-of-the-art research, a Deep Neural Network (DNNs) model has been developed to detect traffic sign anomalies. The models are portable in nature and have been implemented in anti-hacking devices such as Jetson AGX and nano.

Robustness to physical attacks on traffic signs	In addition, it examines the mitigation strategies for physical attacks to improve the robustness of the CAVs pipeline.	To improve the robustness to physical adversarial attack, additional DNNs model has been developed using techniques such as GAN (Generative Adversarial Networks) and autoencoder. This enables the model to generate meta-traffic signs which are anomalies free.

Table 3: Brief overview of proposed sub-use cases in Physical adversarial attack use case.

For the above use cases, following research and development were carried out in the CARAMEL project.

1. Research and develop anomaly detection methods using state-of-the-art techniques such as Deep Neural Networks (DNNs).
2. Develop an additional Deep Neural Networks (DNNs) for the mitigation purpose.
3. Generate traffic sign attacks dataset for training and testing purposes.
4. Develop a traffic sign pipeline integrating all the components such as DNNs models, anti-hacking device as well as backend.
5. Extensive testing on the robustness of the developed models.

2.1.1. Pipeline Demonstration

2.1.1.1. Introduction

For the physical adversarial attack use case, the demo is laid out according to Figure 2. In Figure 2, two types of inputs are set up and they are manual and automatic.

In manual input, a human can control the cars in the environment whereas automatic input autonomously drives the car. The server-side pipeline runs in the PC and consist of a Virtual city (i.e., Carla Simulator) as well as Deep Neural Network (DNN) for traffic sign detection and localization. Nvidia Jetson AGX has been selected as an anti-hacking device. It contains the anomaly detection and robustification DNNs models developed for the CARAMEL project. The models are available in both docker as well as in native format. The backend is a cloud system developed for centralized data collection and visualization by our partners.

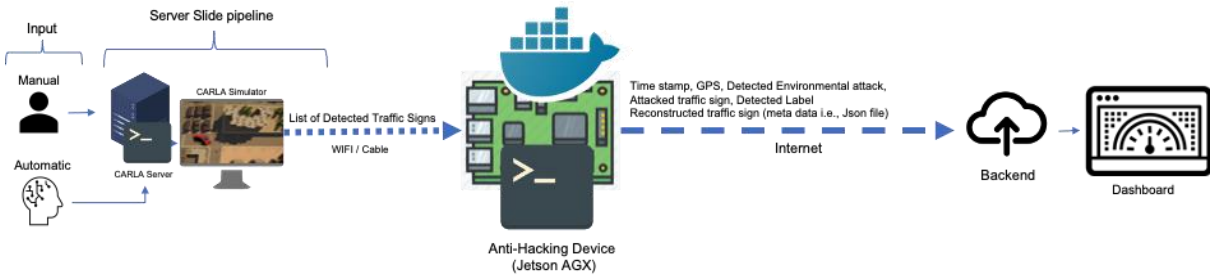


Figure 2: The overview of the traffic sign anomaly and mitigation pipeline for the demonstration purpose.

2.1.1.2. Virtual Environment

For the demonstration purpose, we choose the virtual environment to show the full extent of physical adversarial attack use cases. The virtual environment provides a wider range of flexibility in terms of dramatic changes in environmental elements such as lights, weather, location in a short period. We utilized an open-source Car simulator called CARLA simulator [1]. We developed a virtual city which includes multi types of traffic signs with various attack patterns such as graffiti, noise as Gaussian, salt and pepper, etc. Please refer to the D4.2 report for detailed information about the attack types. In addition, environmental elements such as weather, time (day/night) can easily be modified with the help of a keyboard.

Figure 3: (Right) The top-down view of the Virtual city designed for Physical adversarial attack use case. (Left) Screenshot of various weather and time.

Figure 3 shows the top-down view of a virtual town which was designed and developed for the use case. The Town consists of varieties of scenarios such as countryside (sections with forests) while it also incorporates the urban aspect of the city such as houses, traffic lights etc. In addition, Figure 3 also highlights small samples of weather varieties and time on the right column.

2.1.1.3. Deep Neural Network (DNNs) Models

Two types of Deep Neural Networks were designed and developed for the use case, which are the anomaly detection model and robustification model. Additional two models were also utilized for traffic sign localization as well as traffic sign recognition. Since the traffic sign localisation was out of the project scope, we used SSD MobileNet v2 [2] and developed in house traffic sign recognition models. The models were designed using real and synthetic data. However, for the demonstration purpose, we only focus on the model trained on synthetic data (detailed information in D4.2).

To train and test the models, we developed a pipeline to generate synthetic data from Carla simulator [1]. The synthetic data generation pipeline captures highly accurate annotation for traffic sign detection and classification purposes (refer to D4.2 for more information). Figure 4 shows the overall view of the various Deep Neural Networks and their connections. Majority of developed models run in anti-hacking devices except the third-party traffic sign detection model.

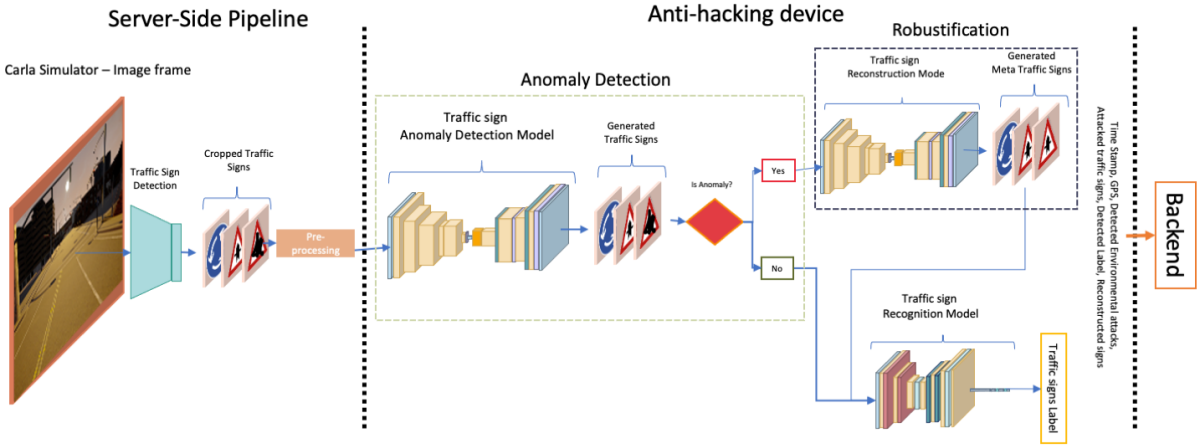


Figure 4: An overview of the anomaly detection and mitigation pipeline.

2.1.1.4. Use case test setup

The physical adversarial attack use case will be demonstrated based on the use case description on D2.4 (UC1.1). The scenario focuses on demonstrating the two sub-use cases described in Table 3 and overview of the test setup was visualized in Figure 1.

As already shown in Figure 1, the autonomous vehicle has a pre-planned destination and travels from start to end point. While performing the task, the vehicle needs to follow the traffic rules including detecting the traffic signs as well as following the signs. In normal scenarios, the vehicle should have no problem reaching the destination and should follow the green path. However, in above scenarios, the traffic sign has been attacked and the vehicle might not understand the instructions and possibly misread the signs. Numerous abnormal behaviors can occur due to such attacks. However, the red path are some examples of alternative routes that a vehicle might take in this situation. Hence, mitigation strategies should be in place to accommodate the scenarios. To that end, in D4.2, we developed a Deep Neural Network (DNNs) model to detect anomalies in traffic signs and an additional model to reconstruct the meta-traffic signs free of anomalies. The aim is to enable the autonomous vehicles to alert the backend system about possible traffic signs attack as well as reach the end destination by recognizing the anomalies. In this case, the blue path represents the path an autonomous vehicle might take when identities the attack.

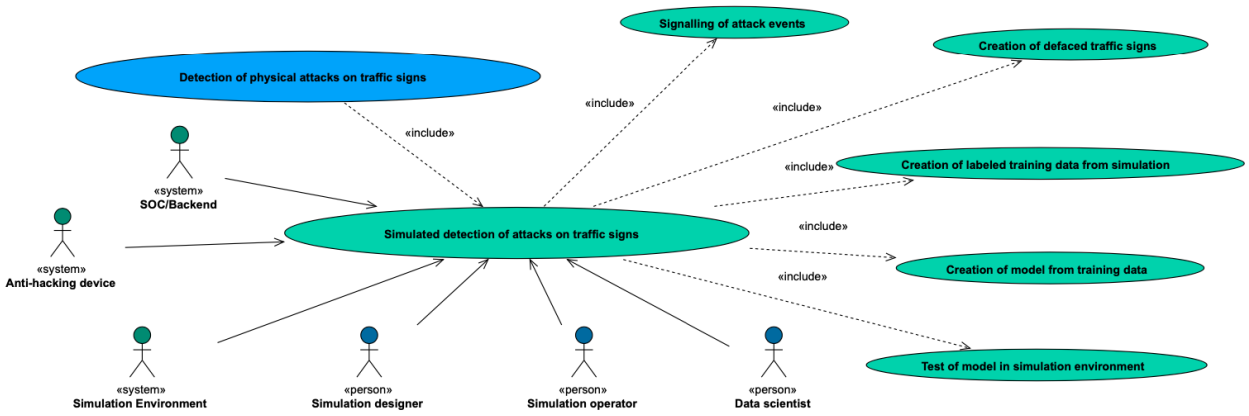


Figure 5: Use case diagram of simulated detection of attacks on traffic signs.

The Figure 5 shows the detailed use case of simulated detection of attacks on traffic signs and the following Table 4 describes the use case:

Title	Description
Purpose	Demonstrate the physical adversarial attack on traffic sign in a simulated environment
Prerequisites	A simulation environment, anti-hacking device.
Involved actors	• Simulation Operator • Simulation Designer • Data Scientist • SOC/Backend • Anti-hacking device • Simulation Environment
Flow of activities	1. A simulation designer creates attacked traffic signs in the simulated 3D environment (e.g., in CARLA simulator). 2. A data scientist generates a large quantity of the labelled data from a simulated environment. 3. A data scientist designs and develops the Deep Neural Networks (DNNs) models to detect attacked traffic signs using generated data.

	4. A simulator operator starts the test of the DNNs models in the simulation environment. The models will be running in the anti-hacking device. 5. The anti-hacking device sends information about attack events if the simulated vehicle encounters attacked traffic signs.
Success end condition	Attack event is signaled.

Table 4: Use case description.

2.2 *Cyber-Attack Detection and Mitigation at the Camera Sensor (Signal Layer)*

Our experiment considers images extracted from various videos captured at diverse weathering, lighting conditions and velocity ranges in areas including supermarkets, office with angle, parallel and perpendicular parking scenarios. The vehicle platform used for data collection is illustrated in Figure 6.

Figure 6: Picture of our ego-vehicle with sensors mounted.

Each raw image (width 1280 and height 960) is attacked by noise, which are then used as inputs for our denoised models. The parameters for denoising models are tuned to generate a noise free image. And secondly, this denoised image is being fed to our computer vision algorithm to analyze the variation in output.

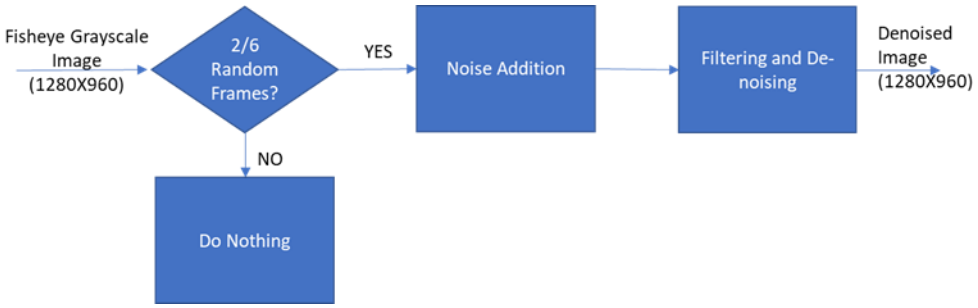


Figure 7: First Experimental Setup.

2.2.1 Vehicle Setup

Due to the required safety of manoeuvres of autonomous vehicles, these vehicles require a high precision perception of their surrounding environment. All the possible information ranging from the

details of motions of used ego vehicle to the motion of surrounding pedestrians can lead the vehicle to make an optimal decision at the right time to avoid any collision. This level of safety confidence can be just archived using an optimal fusion between the collected information of various mounted sensors on the vehicle.

In this section a short description of mounted sensors on Panasonic Automotive Systems Europe (PASEU) automated vehicle is introduced. The proposed system is a vision-based autonomous system including several components as explain below:

Cameras: four wide angle ("fish-eye") (30 frame/sec with a 190º field of view) cameras which are mounted on the side mirrors, the front, and rear bumper of the test vehicle to detect the free spaces and obstacle around the vehicle simultaneously. The detailed specifications of the used cameras are as shown in Table 5:

Parameter	Information-Size
Sensor Model	Sony ISX016
Image Format	Parallel Output YUV422
Serializer Model	FPD Link-Ⅲ
Effective Area	H：1296×V：976
Output Resolution	H：1280×V：960 (Cropping)
Frame rate	30 (Frame/S)
Data logging	54 (MB/S)

Table 5: Technical specifications of PASEU cameras.

Sonars: sonar sensors (10 Hz low range up to 6 m) in front, rear, left, and right of the vehicle to collect additional data about the close object-obstacle to the vehicle.

Velodyne LiDAR Sensor: for the further validation of the performance of the vision-based parking system a 360-degree laser scanner is used. In recent modern "level five" of autonomous vehicles (e.g., Google car), LiDAR sensors are mainly used to collect the information of the surrounding area to feed the perception sections. Lidar technology is still in its infancy, and cost-effective sensors are not yet readily available on the market. In the Panasonic driving platform, scanning lidars are therefore only used for validation. The technical specification of the used sensor is as shown in Table 6:

Parameter	Information-Size
Sensor Model	HDL-32E
Number of Channels	32Up to 100(m)
Range Accuracy	Up to ±2(cm)
Field of View (Vertical)	+10.67 to -30.67 (41.33) (Deg)
Angular Resolution (Vertical):	1.33 (Deg)
Field of View (Horizontal)	360 (Deg)
Angular Resolution (Horizontal/Azimuth)	0.1 – 0.4 (Deg)
Rotation Rate	5 – 20 (Hz)

Table 6: Technical specifications of the used laser scanner in PASEU.

Differential GPS and its internal Inertial Measurement Unit (IMU)**:** the position of the vehicle can be measure precisely at each position using an Applanix Differential GPS (DGPS). The precise position pf the vehicle with the captured information of the surrounding of the vehicle helps the autonomous system

to ensure the performance of other mounted sensors due to the accurate collected ground truth. The accuracy of the used DGPS sensor is as below (Table 7):

Parameter	DGPS	With Post Processing
Position (m)	0.3-0.5	0.02-0.05
Roll/Pitch (Deg)	0.015	0.015
Heading (Deg)	0.02	0.02

Table 7: Technical specifications of the mounted DGPS sensor on the test vehicle of PASEU.

Vehicle network: the related information and collected data from the vehicle are transferred to each component of the system via a local CAN and FlexRay system of the vehicle. This internal information regarding each vehicle motion is updated as follows (Table 8):

Parameter	Update frequency (ms)
Steering wheel position	20
Gear position	5
Rack position	20
Wheels rotation counter	20
Wheels speed	20
Blinker information	10

Table 8: FlexRay messages and their update frequencies in PASEU test vehicle.

Data logger: All the captured data of used sensors are stored on the onboard car-pc of the vehicle. Regarding syncing the data together, currently the live capture engine (Win7 64-bit with Intel(R) Xeon(R) CPU 3.50 (GHz) processor) which is responsible for data logging considers the time stamp of receiving the data (e.g., from each Camera) on the PC and store them accordingly. The logger does not consider the capture time stamp of the data itself.

When the data logger receives a new data, it assigns a timestamp to it from a global time stamp service that is starting from 0.00000 second and is always incrementing.

Ego vehicle: in the automotive industry ego vehicle term is mainly used to refer to the test vehicle which is manipulating the requested tasks. In our system, the whole APS system is mounted on a test vehicle: Mercedes Benz C-Class 2014 (S204).

2.2.2 Scenarios of the Attacks

The demo scenarios executed as part of pillar 1 demonstrations involve a malicious user holding a remote control through which it attacks to the sensors data, disturbing both the quality of the sensor signal and the timestamps of the data recorded. As illustrated in Figure 8, the scene is interpreted in an undistorted way in the right-hand side, while the scene understanding is significantly degraded when the malicious intervention occurs between the scene and the perception engine (shown in the bottom right).

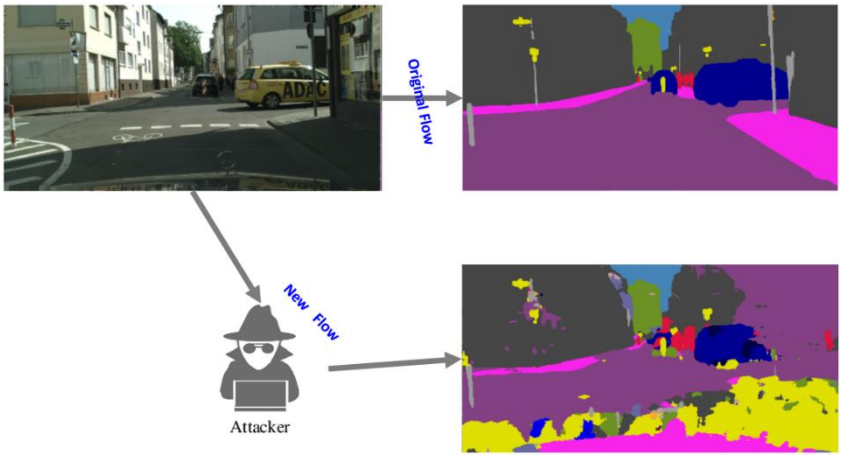


Figure 8: Schematic representation of the attack performed on the real vehicle. A malicious user attacks to the sensor data. Scene perception disturbance is illustrated on the bottom right.

The types and the scenarios of the attacks were defined in D2.2 deliverable contributed by Caramel's consortium at the beginning of the project and is also presented in Table 9 below.

Use Cases - Pillar 1	
1	Attack on the Camera Sensor Layer: This scenario would involve a cyber-attack based on activating some malicious software which got installed during the software update process. Throughout this use-case the camera sensor could be attacked in a few different ways, which could vary between adding noise lying on specific bands of the frequency spectrum/ introducing morphological deformations/ on the whole or parts of the image.
2	Attack on the Camera Sensor Layer by de-synchronizing the data: Throughout this scenario, the cyber-attack will be geared towards disturbing the association between the captured frames and the timestamp assigned to them. This will cause the failure of the perception engine, as all the architectural modules performing stochastic filtering on the scene observations will be affected by error. This use case should study the potential and the limitations of the cyber-attack detection and mitigation engine in assessing and recovering the failures.
3	Attack on the Camera Sensor by a remote agent: In addition to the aforementioned scenario, the cyber-attack detection and mitigation engine will be used to detect and mitigate the camera signal distortion in the case that a malicious remote agent interferes with the test vehicle by knowing the IP of the processing unit and sharing some erroneous data. More specifically, this use case will assume that the remote agent sends via V2X communication: time zone/ daylight related data in order some sensor parameters (e.g., gain/exposure time) to be tuned accordingly.

Table 9: Attack scenarios for the real vehicle demo as presented in deliverable D2.2.

The scenarios presented in Table 9 were used to investigate the potential and limitations of the cyber-attack detection and mitigation engine across a wide range of perception functions related to:

1. Moving Object Detection
2. Self-Localization
3. Occupancy Grid Mapping/ Object Boundaries Definition
4. Fully autonomous parking.

The number and the extend of perception engine components tested, exceeded the initial planning as this was specified in WP2, WP3 and WP4. More specifically, while in the afore mentioned work packages it was planned to have the cyber-attack detection and mitigation engine tested only for use cases (1) and (4), we finally tested it across perception components related to object detection, navigation and automated parking.

The primary navigation of autonomous vehicles depends on the effectiveness of the sensor processing techniques applied to the data collected from various visual sensors. Therefore, it is essential to develop the capability to detect objects like vehicles and pedestrians under challenging conditions such as like unpleasant weather, poor illumination conditions, high speed, and motion on uneven ground. Thus, all the scenarios discussed through sections 2.2.2.1 to 2.2.2.4 have been tested in the test area in Panasonic premises.

2.2.2.1 *Moving Object Detection*

Moving object detection is the task of identifying the physical movement of an object in a given region or area. Over last few years, moving object detection has received much of attraction due to its wide range of applications like video surveillance, human motion analysis, robot navigation, event detection, anomaly detection, video conferencing, traffic analysis and security. In addition, moving object detection is very consequential and efficacious research topic in field of computer vision and video processing since it forms a critical step for many complex processes like video object classification and video tracking activity. Consequently, identification of actual shape of moving object from a given sequence of video frames becomes pertinent. However, task of detecting actual shape of object in motion becomes tricky due to various challenges like dynamic scene changes, illumination variations, presence of shadow, camouflage, and bootstrapping problem. Throughout the demo for CARAMEL, we have performed the experiment as illustrated in Figure 9.

An object is moving in the area behind the ego vehicle various trajectory shapes, speeds and postures and the object detection perception module is detecting the object while the camera sensor is attacked in the way described in Table 9.

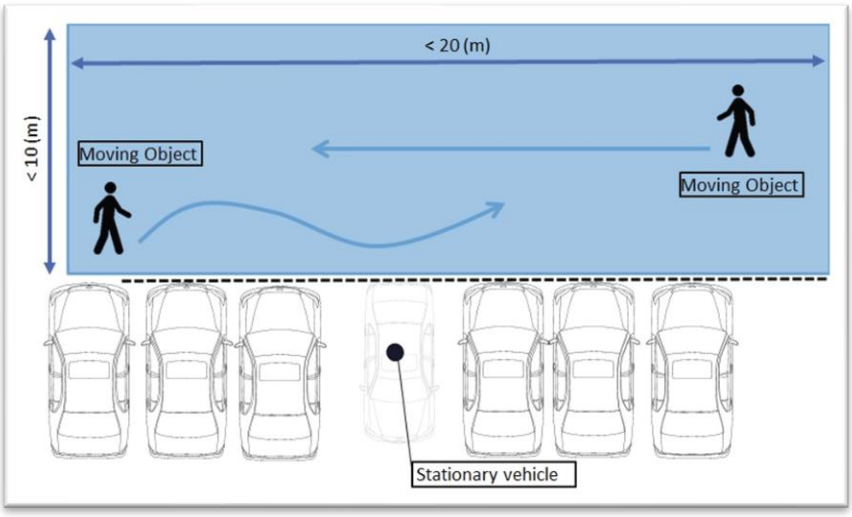


Figure 9: Schematic representation of the parameters of the experiment testing CARAMEL's cyber-attack detection and mitigation engine in the moving object detection use case.

From Figure 10 to Figure 12 illustrate instances of the experiments as well as the output of the moving object detection perception modules.

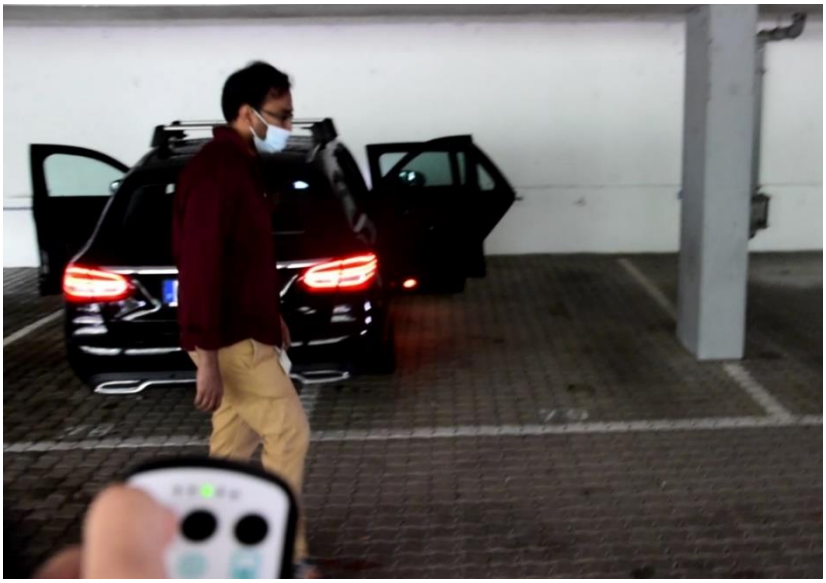
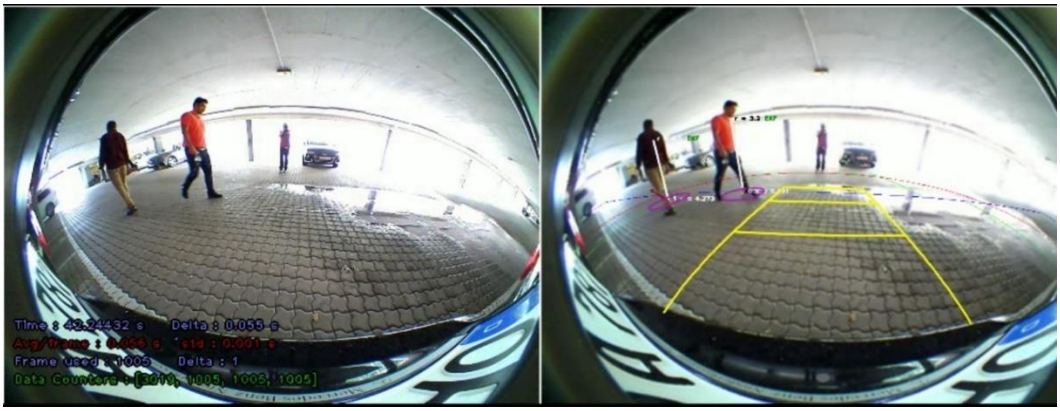


Figure 10: Malicious user attacks to the Moving Object Detection perception component. While the autonomous vehicle, engaged for CARAMEL, is static pedestrians are moving.

Figure 11 and Figure 12 illustrate the output of the moving object detection module at the presence.

(a) (b)

Figure 11. Output of the moving object detector perception engine, at the absence of cyber-attack. (a) camera output at the viewing layer, (b) Output of the perception layer: moving pedestrians crossing the vehicle trajectory. The bounding box highlights the object's boundaries. The direction, speed and time to collision is disclosed.

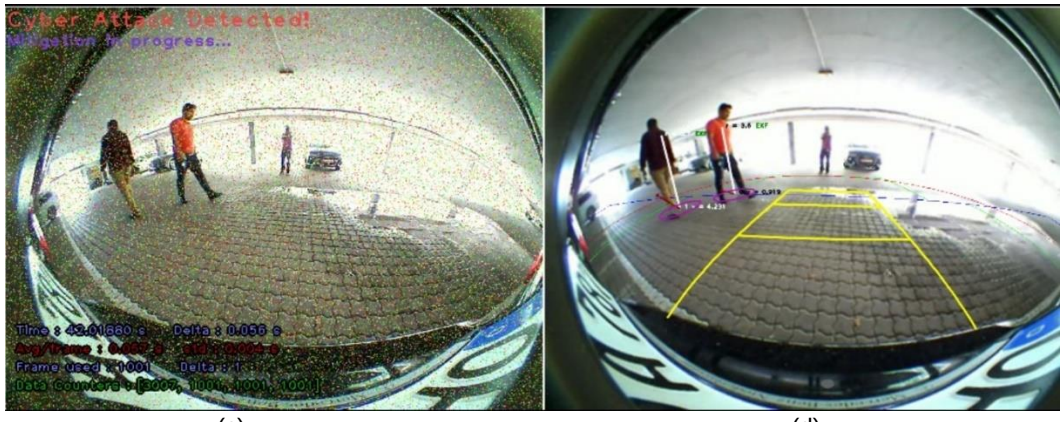


(a) (b)

Figure 12. Output of the moving object detector while the vehicle is not under cyberattack. (a) viewing layer, (b) Output of the perception layer: moving pedestrians out of vehicle trajectory. The direction, speed and time to collision is disclosed.

As part of visual inspecting the efficiency of CARAMEL's cyber-attack detection and mitigation engine, Figure 13 (a)-(d), illustrate the output of moving object detection when the vehicle is submitted to cyber-attack. Figure 13 (a), (c) present the output of camera view, when the vehicle is cyber attacked, while Figure 13 (b), (c) display the output of the moving object detection module after the CARAMEL's cyber-attack detection and mitigation engine. As seen, the localization of moving object as well as the speed and direction of moving objects are still detected.

(a) (b)



(c) (d)

Figure 13. (a), (c) Camera sensor cyber attacked, the cyber attacked is detected and the mitigation engine is under progress. (b), (d) output of the moving object detection module after the cyber-attack mitigation engine has been performed.

Subsequently, section 2.2.2.2 summarizes the demo output of investigating the efficiency of CARAMEL's cyber-attack detection and mitigation engine in self localization operation.

2.2.2.2 *Self-Localization*

As above, the scenario involves an attacker holding a remote control through which he activates some malicious software, already installed in the car (Figure 14).

Figure 14. The autonomous vehicle engaged for CARAMEL is cyber attacked by an external agent, through a remote control.

To assess the potential and the limitations of the CARAMEL's detection and mitigation engine employed to robustify the autonomous navigation function, CARAMEL's test vehicle, as part of the self-localization demo, the vehicle moves on a closed loop, on uneven ground at velocities ranging from 0-15 km/h. While the car moves, the camera sensor is attacked, the location as well as the trajectory of the vehicle is estimated through the camera-based odometry. The accurate mitigation of the attack is assessed through measuring the divergence between the start and end point of the trajectory. In case of successful mitigation, the end and start point should coincide.

In the figures presented below, Figure 15 illustrates an instance of the camera based cyber-attack as it is displayed in the viewing layer (top left), the 3D modelling of the scene after the mitigation of cyber-attack is displayed on bottom left image. The trajectory estimation along with the occupancy grid map is presented on the right image. Moreover Figure 16 illustrates the trajectory estimation (output of self-localization module). The green trajectory is derived by the visual odometry (camera sensor), while the vehicle is under attack. The red trajectory compromises the output of the localization module based on the flex ray sensor. As it can be observed, according to the camera-based localization solution, the vehicle moved on a closed trajectory, which corresponds to the shape of the trajectory performed by the vehicle. However, the red trajectory (estimated by the flex ray sensor) illustrates a considerable divergence. Thus, it can be observed that despite the camera being attacked, the camera-based localization solution performs better than the flex-ray. The inaccuracy of the flex ray odometry is basically due to insensitivity of the mechanical sensors at low speed (<8kph) and steep yaw turning.

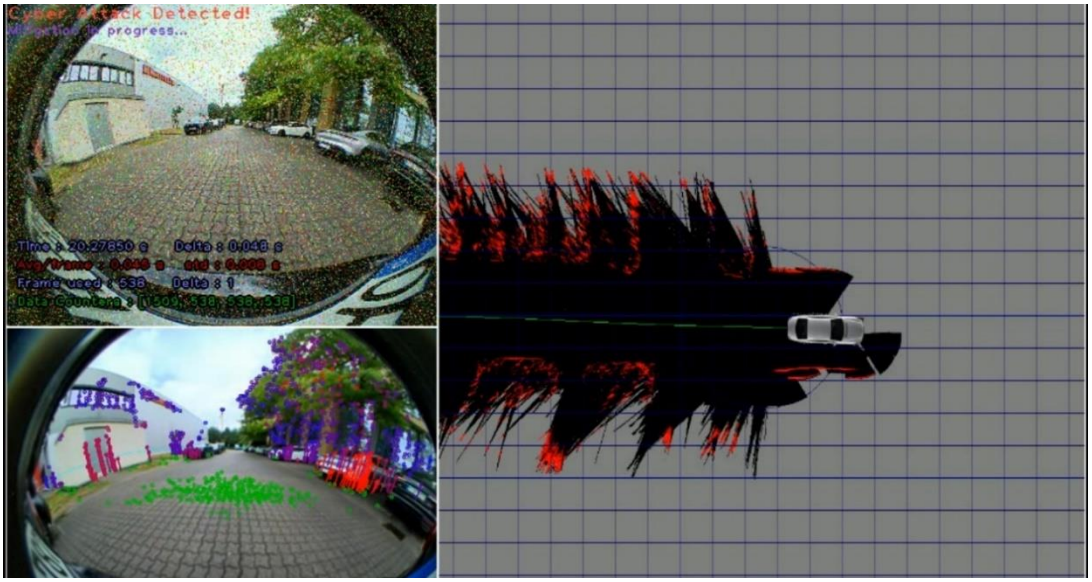


Figure 15. Camera based cyber-attack illustrated in the viewing layer (top left), bottom left: 3D modelling of the scene after the mitigation of cyber-attack. Right image: trajectory estimation and occupancy grid map.

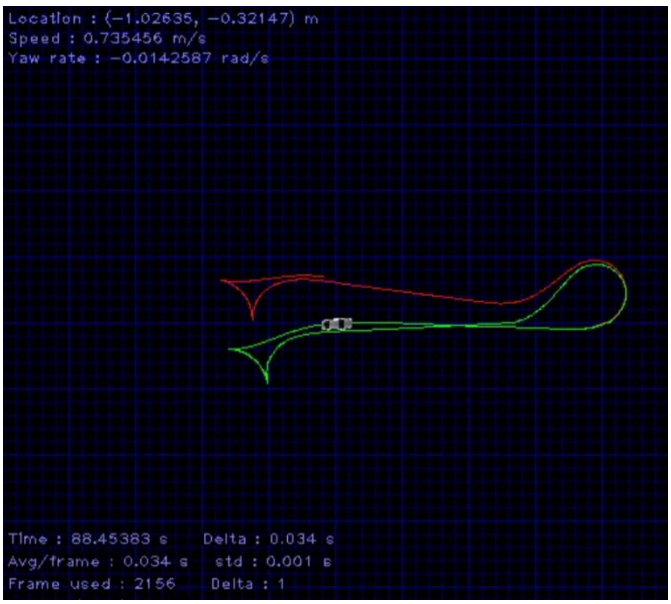


Figure 16. Trajectory estimation as output of self-localization function. The green trajectory is the visual odometry (camera sensor) output, while the camera is cyber-attacked. The red trajectory is the output of flex ray odometry.

2.2.2.3 *Occupancy Grid Mapping/ Object Boundaries Definition*

As discussed in Caramel Deliverables D3.2 and D3.5, It is of interest to understand the impact of noise on the performance of autonomous vehicle in its autonomous navigation. Concretely, we would like to understand how much degradation of the Occupancy Grid Map (OGM) occurs (Figure 17) when the camera is cyber attacked as well as how much the front of the obstacles is distorted (Figure 18). For obstacle avoidance in navigation, it is practically efficient and much more meaningful to compare the outcome of obstacle polygons extracted between OGMs. This is because the polygons are fundamentally used for calculating time-to-collision and planning the local path to avoid possible collision. On the other hand, for parking applications, it is much more meaningful to measure the free-space and obstacle's boundaries within surrounding environment to examine whether a possible parking-slot is detected, and an optimal trajectory can be planned accordingly. To make this clear, let's

review an example in Figure 17. In both figures, the point cloud created by a camera-based 3D reconstruction solution is fed into the OGM. Obstacle polygons (Figure 18) are boundaries of high objects (>15 cm height), which are extracted and marked in red by a pre-determined solution.

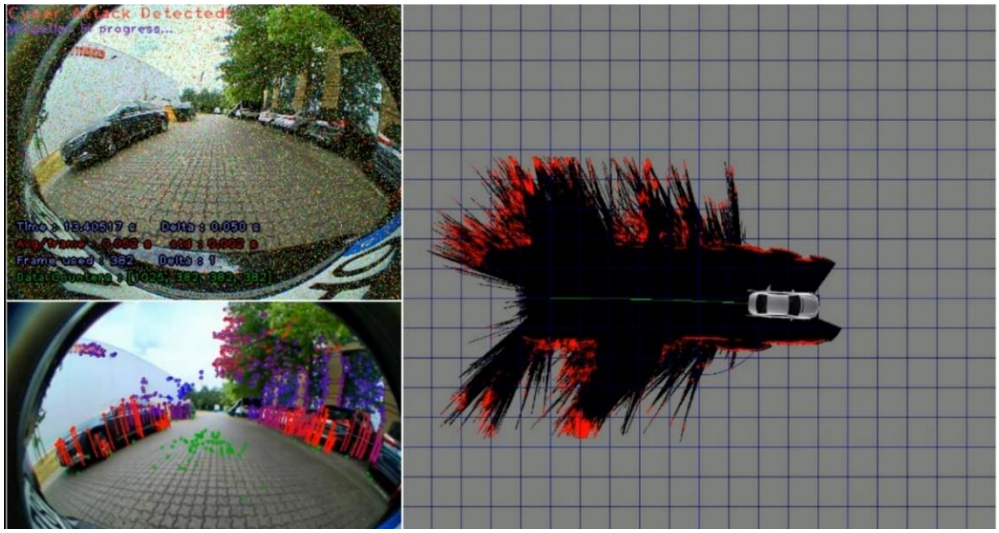


Figure 17. Output of the CARAMEL's cyber-attack immunization engine at the OGM level. Top Left: output of the cyber-attack at the viewing layer. Bottom Left: 3D Environmental sensing layer output. Right Image: Occupancy Grid Map built during the car moving as displayed in the left image.

Figure 18. Output of boundary estimation while the camera is cyber attacked. Top left: Output of 3D environmental sensing. Bottom left: Cyber-attack at the camera sensor. Right image: Polygons enclosing the point clouds clustered as discrete objects.

As it can be assessed by the Figure 17 and Figure 18, the efficacy of CARAMEL's cyber-attack immunization engine can provide robust scene understanding under adverse illumination conditions both outdoors Figure 17 and indoors Figure 18 as well as at diverse weathering conditions, like raining Figure 17.

As the demonstration developed into exploring the robustness of CARAMEL's cyber-attack immunization engine, at first the critical modules contributing to Automated Emergency Braking (e.g., Moving Object Detection, Self-Localization) and obstacle avoidance (self-localization, Occupancy Grid Mapping) were assessed and at a next step, the robustness of CARAMEL's solution in addressing cyber-attacks to the Fully Automated Parking is also investigated. The investigation of this function was out of the scope of the Grant Agreement. However, as the derived results regarding lower-level functions outperformed consortium's expectations, it was decided to also investigate the potential and limitations of CARAMEL's solution at this higher level of functionality.

2.2.2.4 *Fully autonomous parking*

Numerous advanced driver assistance systems (ADAS) are applied in daily transportation vehicles. However, only a few fully autonomous driving applications are implemented due to safety and law considerations. Therefore, because of the relatively low risks and well-known environments, autonomous parking may be the first fully autonomous application. In addition, parking becomes one of the major challenges in metropolitan cities recently because of the increment of vehicle numbers. Also, the size of parking slot and garage space can be decreased without human factor consideration. Therefore, autonomous parking is regarded as one of the autonomous vehicle major benefits. In advance, regarding to the functions involved in autonomous parking, they are categorized into six broad categories (Figure 19): sensors, localization, perception, planning, control. Except for vehicle control, that there are still more strict functional safety rules needed to be fulfilled, all the other modules were involved in the demo presented in the current session.

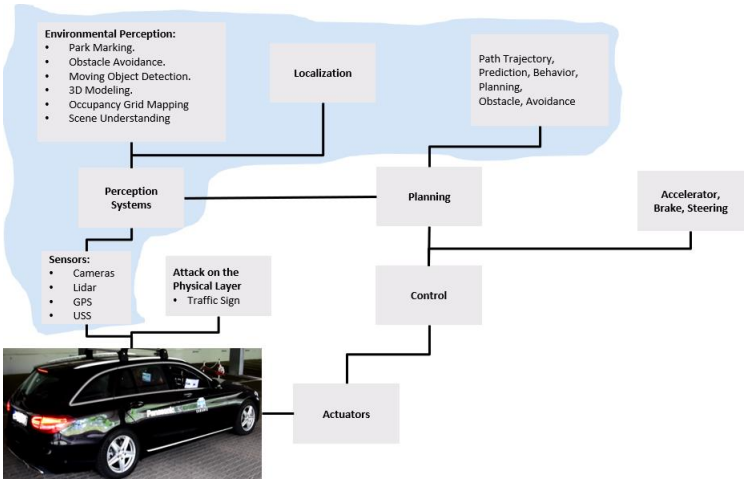


Figure 19. The functional modules implemented on the autonomous vehicle engaged for CARAMEL. The modules enclosed in the light blue bubble are attacked during the demo presented in 2.2.2.4.

As part of assessing the robustness of CARAMEL's solution, the output of Autonomous Parking function at the absence and presence of cyber-attack is investigated. Figure 20 illustrates the output of parking slot detection when no attack is performed, while Figure 21 presents re-iteration of the same experiment at the occurrence of cyber-attack.

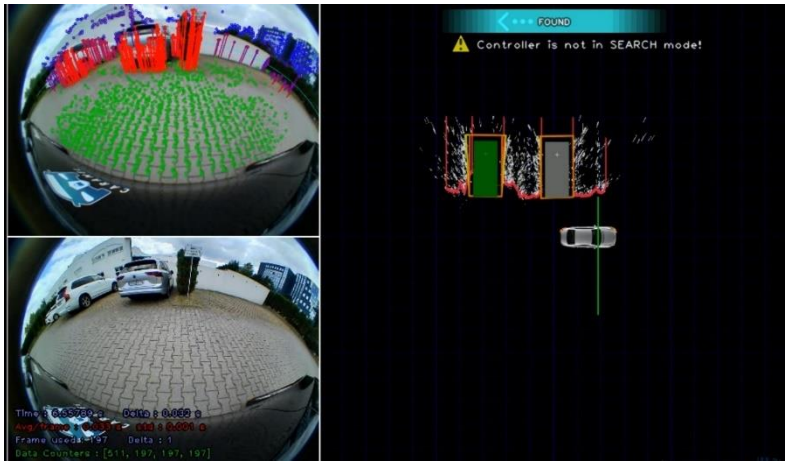


Figure 20. Bottom Left: Viewing Layer, Top Left: 3D environmental modelling, Right Image: Object boundaries (red lines) and Parking slots detected (grey and green). Green slot: Parking slot topology for which the cost function of the planner is minimized.

Figure 21. Bottom Left: Viewing Layer presenting the output of camera sensor being attacked. Top Left: 3D Environmental Sensing. Right Image: Object boundaries presented by the red lines and the available parking slots highlighted by grey and green are presented. The green is the one assessed as optimal by the planner.

Through a comparative study of Figure 21 and Figure 22 it can be verified that the shape and the topology of the object boundaries as detected before and after the cyber-attack are identical. Moreover, the position of the available and parking slots and the optimal are not disturbed because of the attack.

Figure 22 and Figure 23 also provide a comparative study of the final parking function performance in the absence of attack and while the vehicle is attacked correspondingly.

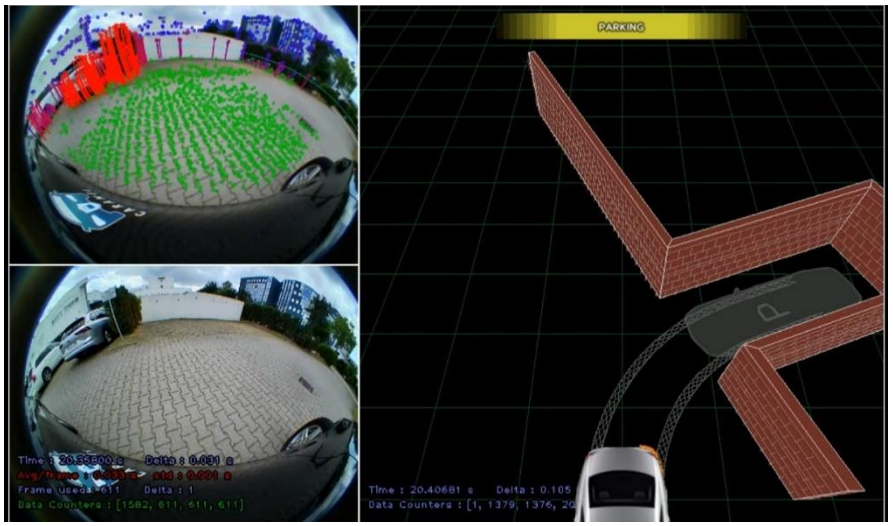


Figure 22. Bottom Left: Viewing Layer presenting the output of camera sensor. Top Left: 3D Environmental Sensing. Right Image: graphic representation of the position and planned trajectory of ego vehicle, while parking.

As it can be assessed through Figure 22 and Figure 23, CARAMEL's solution robustifies the Automated Parking function towards the cyber-attacks. Thus, allowing all the modules involved: perception, localization, planning to be immune.

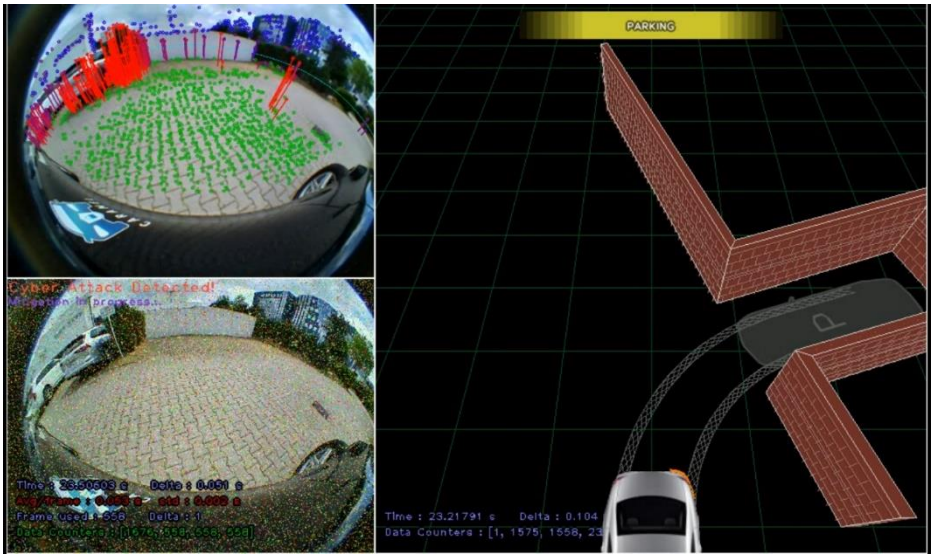


Figure 23. Bottom Left: Viewing Layer presenting the output of camera sensor under cyber-attack. Top Left: 3D Environmental Sensing. Right Image: graphic representation of the position and planned trajectory of ego vehicle, while parking.

2.3 *Adversarial attacks on Camera Sensor (Perception Layer): attack use case description and assessment*

The vehicle will be evaluated in real images and point clouds using the KITTI dataset [3]. In each frame, the perception engine will fuse the 2D and 3D results coming from the camera and LIDAR sensors accordingly to increase the situational awareness of the driver. At some point, an external attacker will deteriorate the image data by applying adversarial noise and some objects of the scene (pedestrian/vehicle/bicycle) will be hidden from the perception engine. The proposed multimodal fusion algorithm will manage to alleviate this type of noise by integrating a denoising method atop a robust CNN (Convolutional Neural Networks) detector and then correlating the 2D with 3D outputs.

In overall, to highlight the impact of our technique, the different perception results will be shown to the user in case of an attack, by activating and deactivating our algorithm. The proposed use case is shown in Table 10.

Title	Description	Solution Developed for CARAMEL
Robustness to adversarial attacks on the camera sensor	The autonomous vehicle is expected to drive from a starting location to a given destination following a specified path. At a given time instance, the image of the vehicle will be tampered through a specific perturbation (adversarial noise) intended to cause the perception module to misbehave (e.g., hide objects that are within the field of view).	A two-stage solution has been proposed. A coarse mitigation step has been applied at the first stage inside the anti-hacking device, aiming to alleviate the adversarial attack from malicious images. Then, a fine mitigation step has been implemented inside the on-board unit that fuses information from multiple modalities. Finally, a robust decision for the scene perception is shown to the driver.

Table 10: Brief overview of proposed use case in adversarial attack on camera sensor.

2.3.1 Pipeline Demonstration

2.3.1.1 *Introduction*

The proposed pipeline consists of two stages performing coarse mitigation and a fine mitigation step. Data from the camera and lidar sensors are given as input to the whole procedure. A 2D image denoising method has been implemented inside an embedded device for the first step, aiming to

alleviate the adversarial attack from the malicious images. Next, a multimodal fusion method has been implemented in the on-board unit as a fine mitigation step, by combining the results of a 3D object detection and a 2D image segmentation network. The final output contains a flag, indicating whether there exists an external attack on the camera sensor or not, and the detected objects in the scene. The high-level architecture is shown in Figure 24.

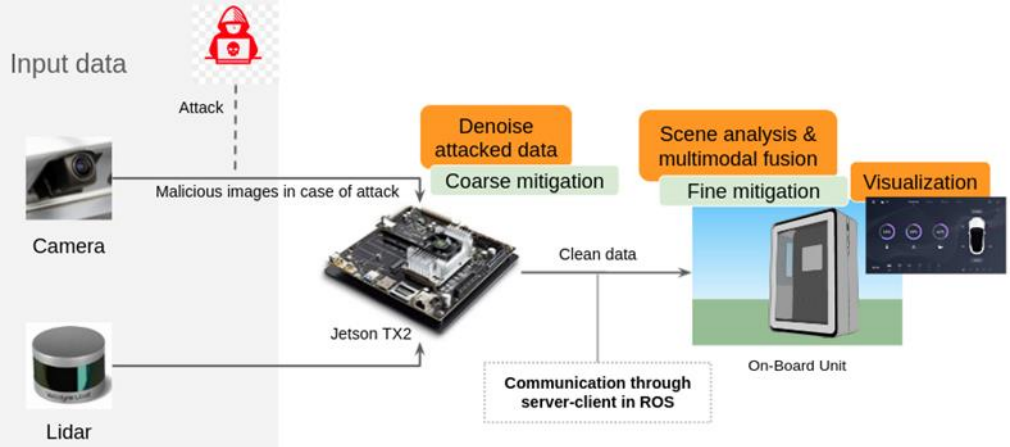


Figure 24: Overview of the pipeline showing the interaction between the anti-hacking device and the on-board unit.

2.3.1.2 *Dataset*

KITTI dataset was utilized for our experiments. It has been captured from a Volkswagen (VW) station wagon for use in mobile robotics and autonomous driving research. In total, there are recorded 6 hours of traffic scenarios at 10-100 Hz using a variety of sensor modalities such as high-resolution colour and grayscale stereo cameras, a Velodyne 3D laser scanner and a high-precision GPS/IMU inertial navigation system. It also contains object labels in the form of 3D tracklets and online benchmarks for stereo, optical flow, object detection and other tasks. In our case, data coming from the sensors of the RGB camera, semantic segmentation camera and lidar were utilized to train our algorithms. Some random images are illustrated in Figure 25.

Figure 25: Random images from Kitti dataset.

2.3.1.3 *Deep Neural Networks (DNNs) Models*

Three types of deep learning models were implemented for our proposed solution, as we can observe from Figure 26. Firstly, a 2D Image Denoising model was trained that follows the structure of AdaFM [4] network to enhance high-level vision applications. The training of a 2D Robust Image Segmentation follows, in which Deeplab [5] architecture was utilized. This model revisited the Atrous Spatial Pyramid Pooling by experimenting with cascading and parallel application of dilated convolutions. Finally, for the 3D Object Detection part, PointRCNN [6] was proposed that achieves state-of-art results in a two-stage 3D object detection framework. Finally, each of the previous models are shown in the next sections:

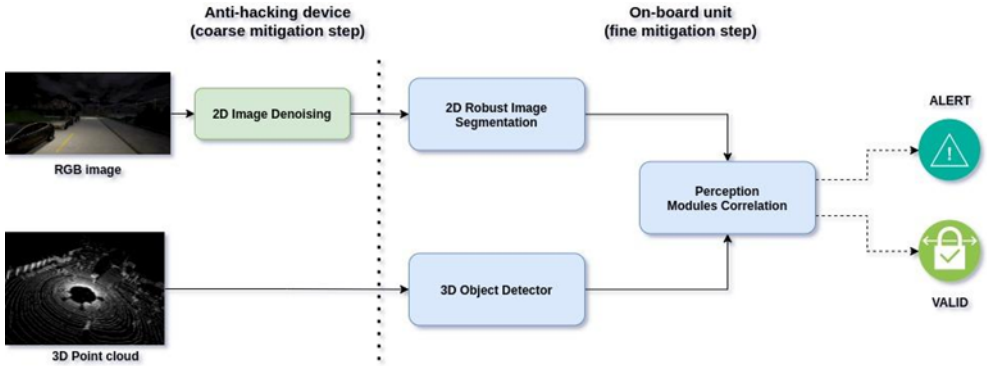


Figure 26: Overview of the pipeline between anti-hacking device and on-board unit.

2.3.1.3.1 2D Image Denoising

AdaFM network was utilized for the 2D Image Denoising part. AdaFM enables consecutive modulation of the restoration strength with little computation cost. At first a standard restoration CNN is trained for the start level, and then AdaFM layers are inserted to optimize it to the end level. After the training stage, CNN parameters are being fixed. The filters of AdaFM layers are interpolated according to the testing restoration level. By using a controlling coefficient, the CNN can consecutively and interactively manipulate the restoration effects. Finally, the whole architecture is illustrated in Figure 27.

.

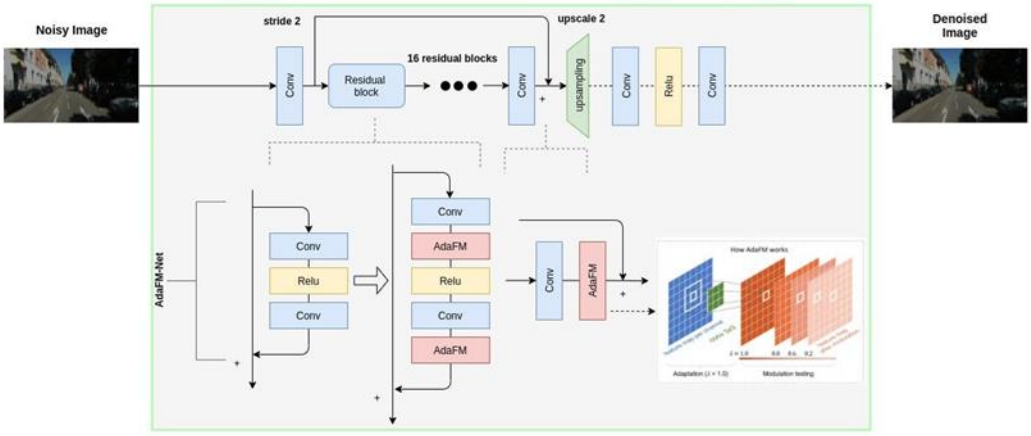


Figure 27: Architecture of 2D image denoising network.

2.3.1.3.2 2D Robust Image Segmentation

Deeplab revisited the Atrous Spatial Pyramid Pooling (ASPP) [7] by experimenting with cascading and parallel application of dilated convolutions. This allows them to improve upon their previous work [5] while achieving comparable results to PSPNet [8]. As denoted in [9] it performs multi-scale processing and should be preferred in safety-critical applications due to its inherent robustness against adversarial attacks. Finally, the whole architecture is illustrated in Figure 28.

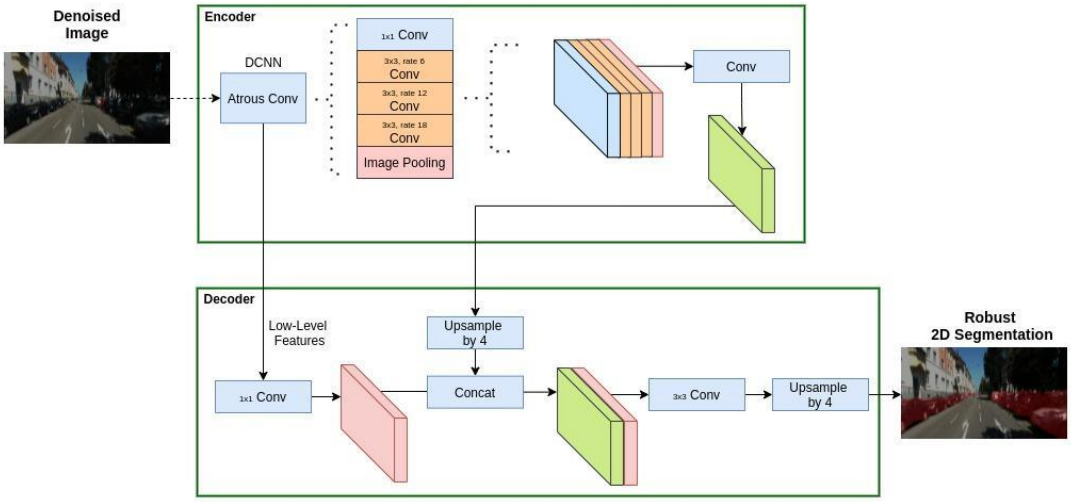


Figure 28: Architecture of 2D robust image segmentation network.

2.3.1.3.3 3D Object Detector

PointRCNN achieves state-of-art results in a two-stage 3D object detection framework. The first stage segments foreground points and generates a small number of bounding box proposals from the segmented points simultaneously, while the second stage conducts canonical 3D box refinement. The whole architecture of PointRCNN is illustrated in Figure 29.

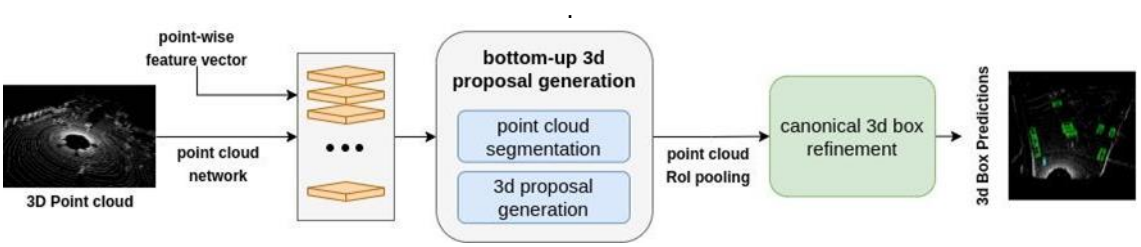


Figure 29: Architecture of 3D object detector.

2.3.1.4 *Use case test setup*

The general idea of the proposed use-case is shown in Figure 30. The autonomous vehicle (shown in orange color) is expected to drive from a starting location to a given destination following a specified path. At a given time instance the image of the vehicle will be tampered through a specific perturbation (adversarial noise) intended to cause the perception module to misbehave and hide objects that are within the field of view. On the left sub-image, the result of the perception engine is shown without the integration of our solution. In that case, an external attacker has added adversarial noise to the camera image. As a result, the segmentation network is unable to perceive the environment successfully. Hence, some objects, such as a pedestrian and a vehicle are hidden from the perception engine making it possible for a collision. On the other hand, as we can observe from the right sub-image, the integration of our solution manages to restore the attacked image. Overall, we managed to achieve contextual and situational awareness, by fusing different data sources of information to facilitate the decision-making process and decrease the rate of possible collisions.

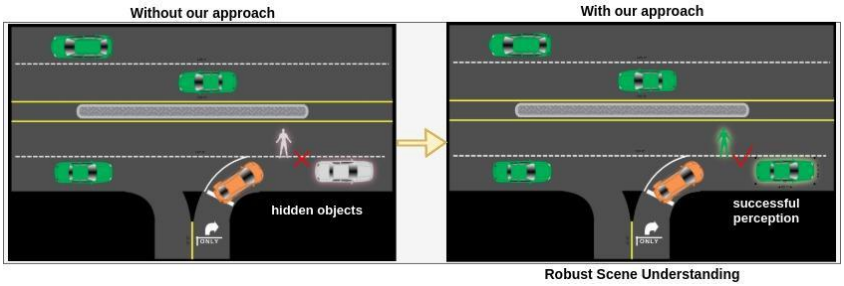


Figure 30: Proposed use-case.

2.3.1.5 *Demo layout*

The demo layout consists of three outputs as we can observe from Figure 31. On the bottom left we can observe the 3D object detection of raw point clouds coming from the Jetson device. On the bottom right, the 2D semantic segmentation is shown of the denoised image coming from jetson. Finally, on the top, the multimodal fusion output is illustrated between the 2D and 3D results.

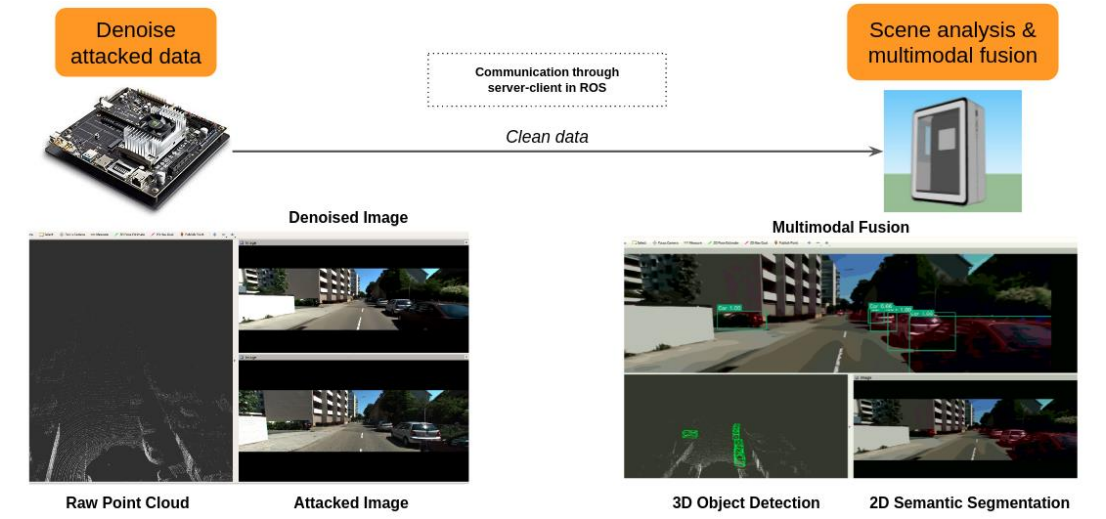


Figure 31: Demo layout.

2.3.2 Image Quality Deterioration Attacks on Camera Sensor

Autonomous vehicles increasingly rely on cameras to provide the input for perception and scene understanding (e.g., object detection, segmentation, and steering angle prediction). Hence, the ability of scene understanding models to perceive their environment effectively, under adverse conditions is crucial. When failures occur - either unintentionally or through targeted attacks, they affect the integrity of camera sensor data and in turn could render these cameras ineffective in providing reliable input to the autonomous vehicle. The aim of this use case is to demonstrate the effect of image deterioration attacks on the camera sensor and through specific scenarios demonstrate the developed detection and mitigation strategies based on the Drive Guard framework, which was elaborated in deliverables D3.2 and D4.2. Drive Guard is implemented as an anti-hacking device which acts as a middle component between the camera unit and the on-board perception module of the vehicle which for the purposes of this use-case will be simulated on a dedicated PC running the CARLA simulator.

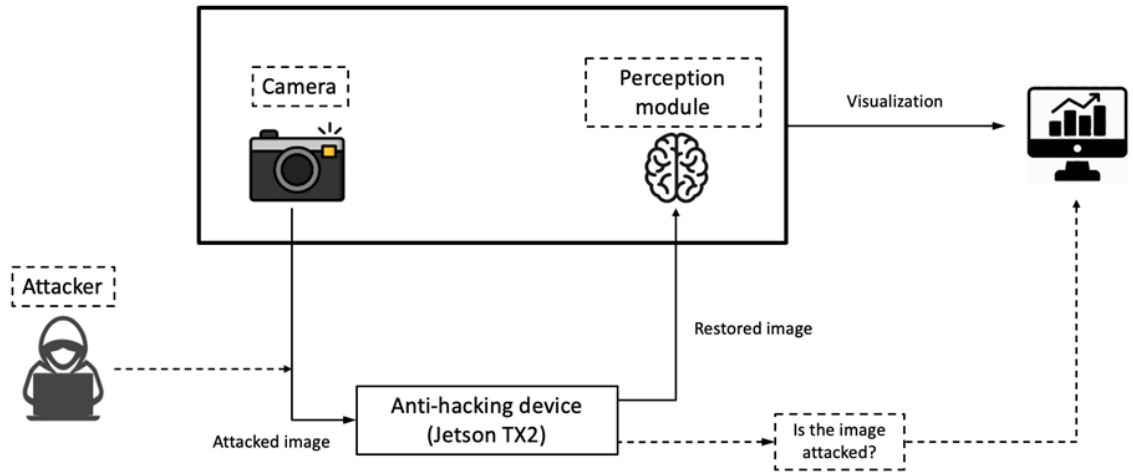


Figure 32: Layout of the use-case scenario.

2.3.2.1 *Dataset/Virtual Environment*

The open-source autonomous driving simulator, CARLA, was chosen to serve as a modular and flexible API to address the need of generating virtual environments for our demonstration purposes. This environment provides a wide range of flexibility in generating different regional sceneries as well as the opportunity to alter different environmental elements such as the involved actors and weather conditions.

Figure 33: The CARLA environment [1].

2.3.2.2 *Drive Guard Summary*

The Image quality deterioration could result in rendering the cameras ineffective in providing reliable input to the perception model of the autonomous vehicle. To mitigate the effect of this adversity we developed “Drive Guard”, a deep learning solution which was described extensively in deliverables D3.2 and D4.2. We provide here a summary for completeness.

This framework is based on a convolutional autoencoder that learns to restore the image quality. In addition to traditional autoencoder architectures, the autoencoder effectively incorporates the temporal dimension, in an approach that also encodes information from preceding frames. Through the deployment of sequences of frames as input to the model we strengthen its understanding of object structure. This is achieved by incorporating a second input stream which encodes the information of the previous frame in parallel with the current frame-steam, for the first two layers. The two streams are concatenated and inputted to the third encoding layer. The structure of the spatiotemporal autoencoder is shown in the Figure 34.

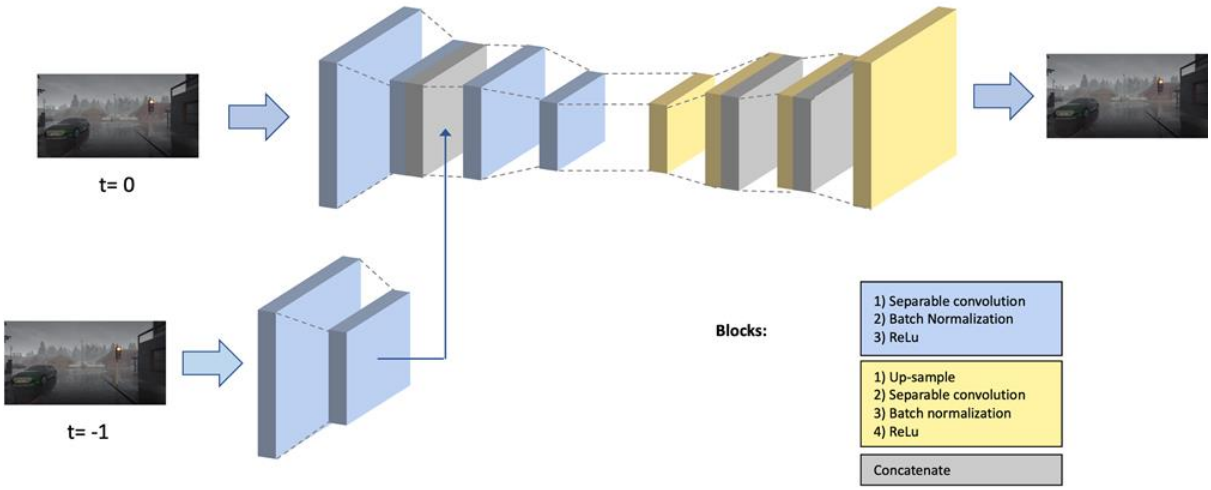


Figure 34: The Drive Guard deep learning model.

2.3.2.3 *Use case test setup*

The autonomous vehicle is expected to drive from a starting location to a given destination following a specified path. At a given time instance the image of the vehicle will be tampered through a specific perturbation intended to cause the perception module to misbehave (e.g., either detect objects that are not truly present or hide objects that are within the field of view). Drive Guard receives the raw image from the camera sensor, flags whether an attack on the image was detected or not and feeds the reconstructed image into the perception module. Then a comparison is made by evaluating and comparing the perception module outputs with and without Drive Guard.

The Involved Actors are:

- **Simulation environment:** The open-source autonomous driving simulator, CARLA, was chosen to serve as a modular and flexible API to address the need of generating virtual environments for our demonstration purposes. This environment provides a wide range of flexibility in generating different regional sceneries as well as the opportunity to alter different environmental elements such as the involved actors and weather conditions.
- **Attacker:** The attacker is responsible for executing image deterioration attacks on the images supplied by the camera sensor. These attacks are simulated by adding noise and artefacts to

the original camera sensor images. The attack is considered to be injected as a small piece of software that manipulates the input image. The severity and access control gained by this software injection can of course vary but we consider here a simple case where the attacker can manipulate the camera capturing pipeline to alter the image data.

- **Anti-hacking device:** The algorithmic detection solutions will be present into an embedded anti-hacking device that will be capable of passive detection of attacks on the vehicle's visual perception modules. The anti-hacking device will be a physical controller that will be integrated into the simulated autonomous vehicle. Its task is to run dedicated ML models that work on the sensor data to detect anomalies that might point to malicious attacks. This is responsible to receive the output of the camera sensors, identify and mitigate the possible attacks on the images and feed the reconstructed images back to the simulated perception module of the car, which for the purposes of this demo will be implemented on the CARLA PC.
- **Simulation designer/operator:** A simulator designer creates different scenarios within the CARLA simulator environment by altering the regional scenery components and the involved actors. These environments will be used to generate different scenarios, which in turn will be employed to evaluate the performance of the proposed solution in detecting and mitigating the effects of image deterioration attacks on the camera sensor. In addition, this actor tests the anti-hacking device and evaluates its performance by applying it on different simulated scenarios.
- **Data scientist:** Generates a large quantity of labelled data from the simulated environment. Designs and implements the Deep Neural Networks, able to detect and mitigate the effects of any possible attacks on the camera sensor. The models are trained on the labelled data provided by the data scientist.

2.3.2.4 *Demo layout*

The simulation will cover two possible cases. First receiving data from the CARLA simulator and using the Jetson device to detect and mitigate an image, as well as receiving data sequences corresponding to data captured from real cases. In both cases the outputs are visualized by displaying the attack detection signal and severity of the attack.

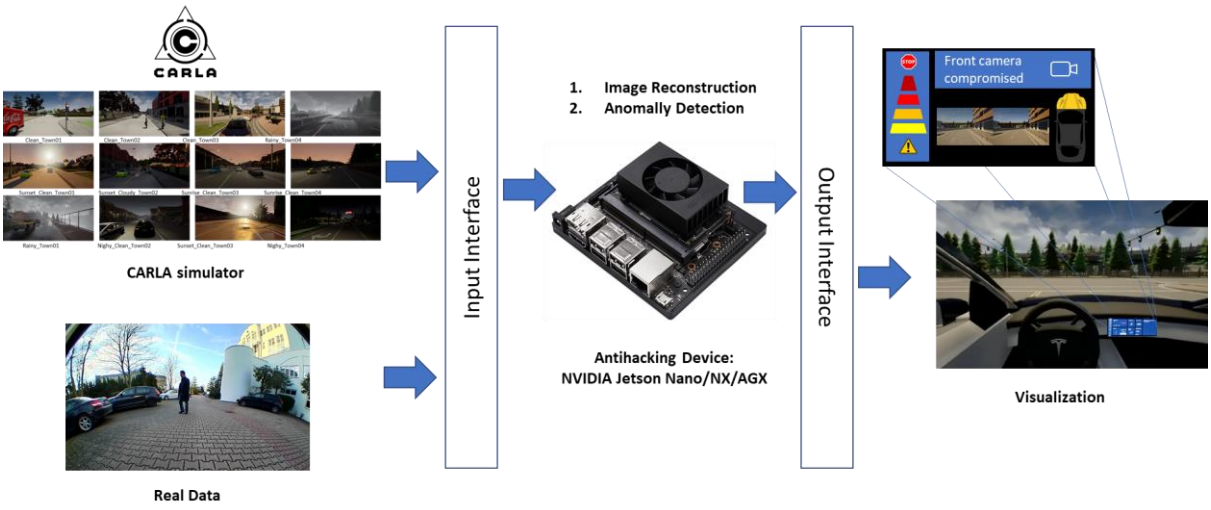


Figure 35: Example simulation and visualization layout of the overall system.

3 Pillar 2 scenario driven attacks

The mobility pillar tackles with the detection and mitigation of cyberattacks performed at the communication layers. Moreover, specific type of attacks at the OBUs and the RSUs are considered and the mitigation strategies which were developed during CARAMEL, will be analyzed during the chapter 3 and its subsections and the results of the experimentation faced “demo” will be presented inside it.

3.1 *Location spoofing attack: attack use case description and assessment*

Using SDR hardware, the attacker can spoof GPS satellite signals. The vehicle relies on a second location stream to identify a possible GPS location spoofing attack, based on the vehicle's movement description, IMU and GPS-free localization measurements.

3.1.1 Collaborative GNSS spoofing detection and mitigation mechanism

3.1.1.1 *Use case setup (UC2.1)*

Location spoofing attack aims to compromise the self-positioning ability of vehicles.

A location spoofing attack attempts to fool a GNSS receiver by broadcasting false satellite signals, focused on resembling a set of normal satellite signals. These spoofed signals may be modified in such a way to cause the receiver to estimate its location even kms away from its actual position. The impact of this attack is more devastating if we consider a group of connected vehicles, which exchange their location measurements in order to coordinate their actions. Broadcasting falsified GNSS positions, then severe traffic accidents are more likely to take place, injuring drivers, pedestrians, cars, etc. Therefore, this use case targets on evaluating the performance of CARAMEL’s collaborative defense mechanism against GNSS spoofing in terms of: i) mitigation ability (i.e. compensate the effect of spoofing), ii) detection ability (i.e. identify the ids of compromised vehicles). The conceptual architecture of this approach is shown in Figure 36.

The testing simulated framework deployed can be illustrated in Figure 37. It is composed of a data-producing end (simulator) and data consumers (ROS nodes). The simulator is used for producing data from sensors attached to vehicles. The ROS Bridge translates the sensor data to ROS-compliant messages and the ROS Nodes utilize the messages.

The described setup was deployed on a PC with the following specifications:

- Ubuntu 18.04
- RAM 16GB
- GPU NVIDIA RTX 2080
- CPU i7 Intel
- ROS Melodic
- Carla 0.9.10-1

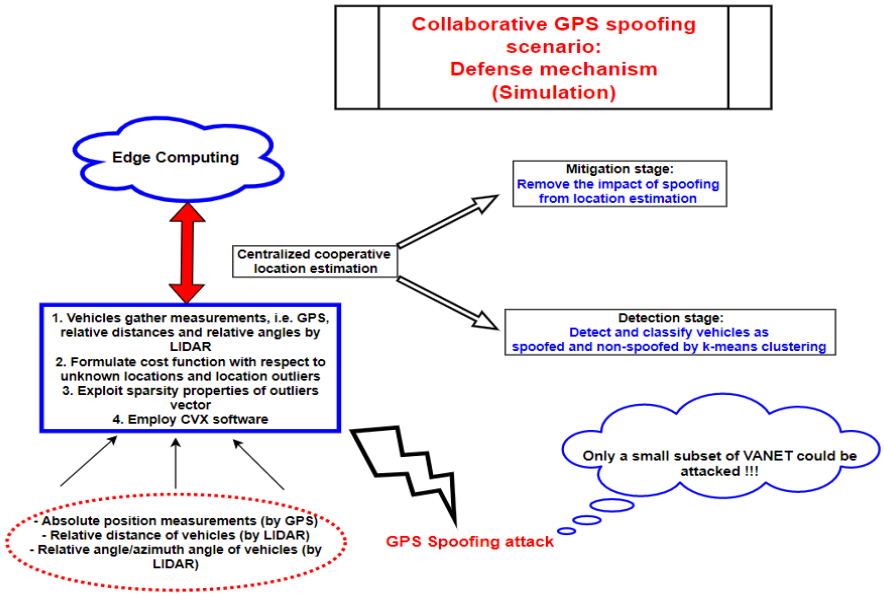


Figure 36: High-level architecture of CARAMEL's collaborating defense mechanism.

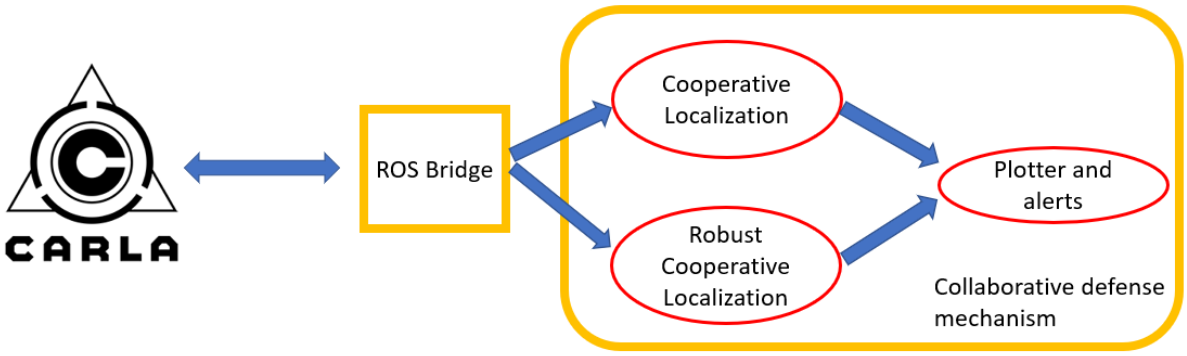


Figure 37: Testing simulated framework.

CARLA is an open-source autonomous driving simulator. It is based on Unreal Engine for conducting the simulation and utilized the Open-Drive standards for defining targets and urban settings. Simulation parameters can be controlled programmatically via a C++ or Python API. Moreover, it consists of scalable client-server architecture, in which the server is responsible for every having to do with the simulation, like scene rendering, updating physics, actors' state etc.

A brief list of the simulator's distinct features includes:

- The traffic manager is a built-in system used for enforcing realistic behaviors upon the vehicles.
- Various sensors for publishing information (LIDAR, RGB, depth, RADAR, IMU, GNSS, semantic).
- A recorder for re-enacting the simulation step by step.
- The ROS bridge for integrating CARLA to ROS.
- Easy creation and customization of assets.
- The scenario runner for describing routes and traffic scenarios.

The Robot Operating System (ROS) is a set of software libraries, tools and conventions that consists of a flexible framework for writing robot software. ROS requires a Linux OS and depending on the Linux version there is a relevant ROS version. For ROS melodic the installation procedure is the following:

- Setting up your computer to accept software from packages.ros.org.
- The ROS-Bridge facilitates the bidirectional communication between the simulation environment and ROS runtime.
- The messages from CARLA have translated to relevant ROS topics and at the same time messages originated from ROS get translated to CARLA commands.

The Involved Actors of the use case are summarized as follows:

- **Ego vehicle and the cluster which belongs to**
- **Simulation environment:** CARLA-ROS testing simulated framework.
- **Attacker:** The attacker is responsible for executing spoofing attacks against the GNSS receivers of cluster's vehicle. These attacks are simulated by adding zero mean Gaussian noise with high variance to the ground truth positions of vehicles.
- **CARAMEL's collaborative defense mechanism:** The associated algorithmic mitigation and detection solutions. Its task is to run dedicated AI solutions that work on the data transmitted to the central node. It is responsible to receive the measurements of the cluster's vehicle, identify and mitigating the possible attacks on the GNSS receivers and feeding the re-estimated positions back to the involved vehicles. For this demo, this will be implemented on the CARLA ROS PC.

3.1.1.2 *Demo layout*

Step 1: Initiate CARLA simulator:

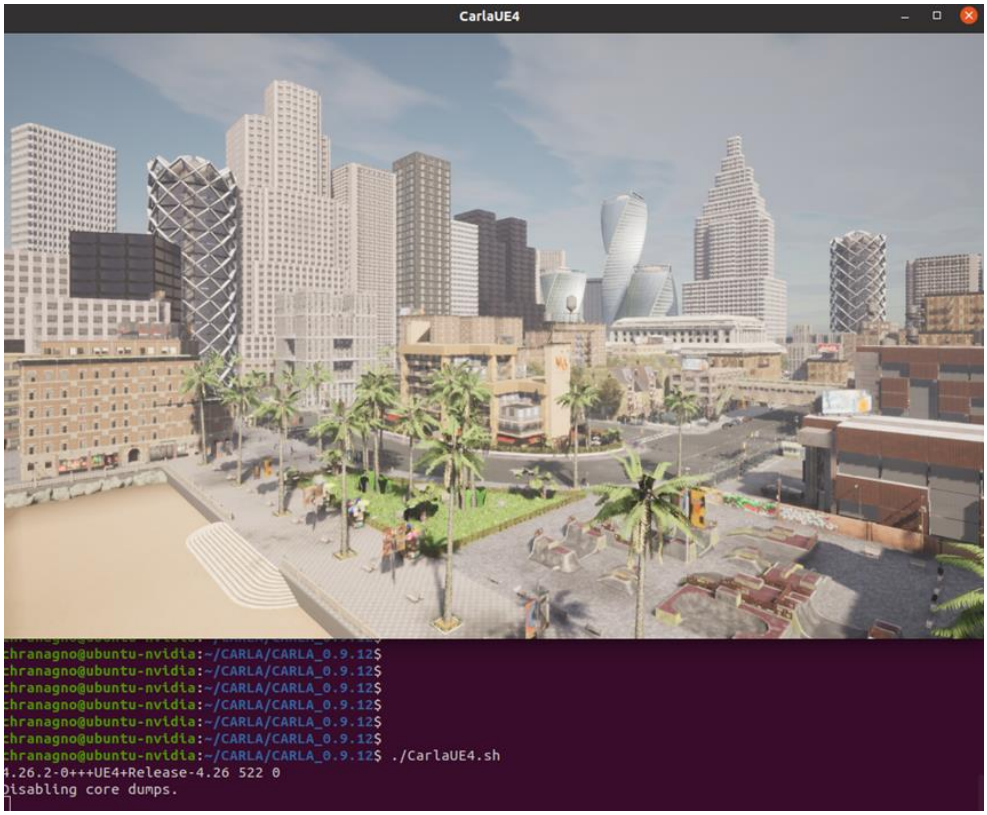


Figure 38: CARLA environment.

Step 2: Ego vehicle starts driving:

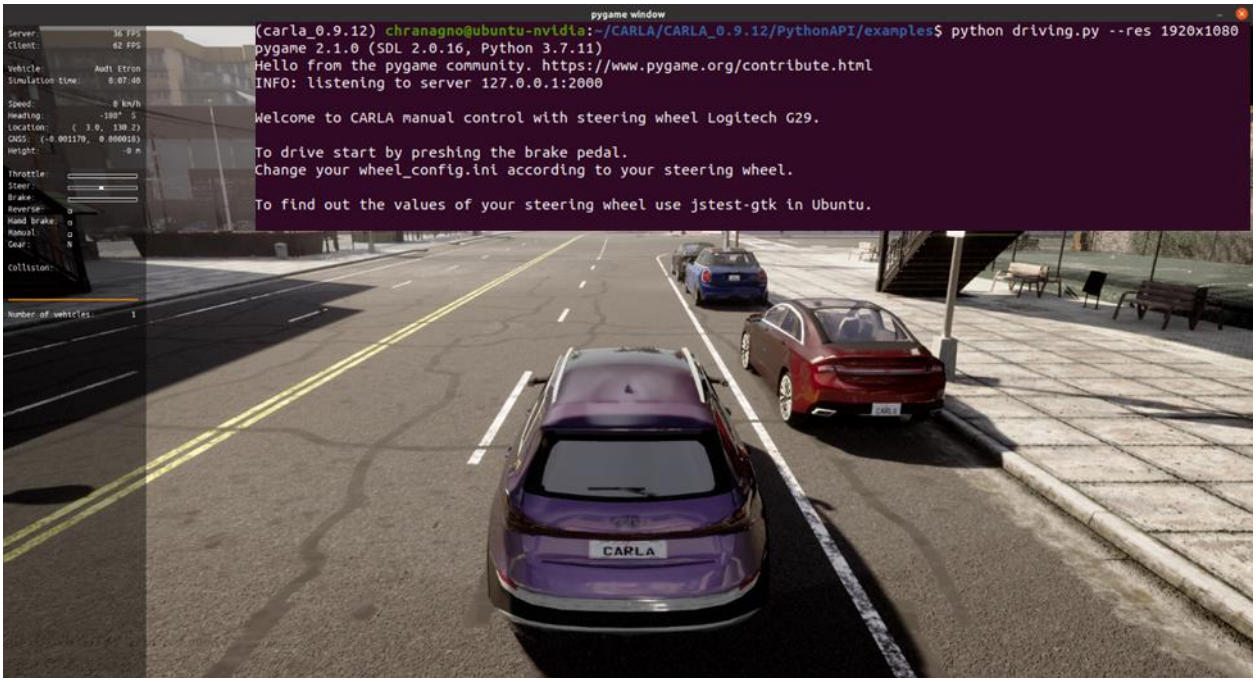


Figure 39: Ego vehicle.

Step 3: Spawn 50 vehicles inside the simulated city:

```
(carla_0.9.12) chranagno@ubuntu-nvidia:~/CARLA/CARLA_0.9.12/PythonAPI/examples$ python generate_traffic.py -n 50
spawned 50 vehicles and 10 walkers, press Ctrl+C to exit.
```

Figure 40: Spawned vehicles in the CARLA.

Figure 41: Ego vehicle and its neighbors.

Step 4: Setup the ROS bridge between the simulation environment and ROS runtime:

```
chranagno@ubuntu-nvidia:~$ roslaunch carla_ros_bridge carla_ros_bridge_with_example_ego_vehicle_cooperative.launch
town:=Carla/Maps/Town10HD_Opt
... logging to /home/chranagno/.ros/log/901efb04-e721-11ec-974d-f1954e2001a4/roslaunch-ubuntu-nvidia-1263638.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://10.10.192.32:33149/

SUMMARY
========

PARAMETERS
 * /carla/ego_vehicle/role_name: ['hero', 'ego_veh...
 * /carla/fixed_delta_seconds: 0.05
 * /carla/host: localhost
 * /carla/passive: True
 * /carla/port: 2000
 * /carla/synchronous_mode: False
 * /carla/synchronous_mode_wait_for_vehicle_control_command: True
 * /carla/timeout: 10
 * /carla/town: Carla/Maps/Town10...
 * /carla_spawn_objects_ubuntu_nvidia_1263638_5036552109459920771/objects_definition_file: /home/chranagno/c...
 * /carla_spawn_objects_ubuntu_nvidia_1263638_5036552109459920771/role_name: ego_vehicle
 * /carla_spawn_objects_ubuntu_nvidia_1263638_5036552109459920771/spawn_point_ego_vehicle:
 * /carla_spawn_objects_ubuntu_nvidia_1263638_5036552109459920771/spawn_sensors_only: True
 * /rosbag_fname:
 * /rosdistro: noetic
 * /rosversion: 1.15.14
 * /use_sim_time: True
```

Figure 42: ROS bridge.

Step 5: Execute the Python script for spoofing detection and mitigation and bird's eye view display of results:

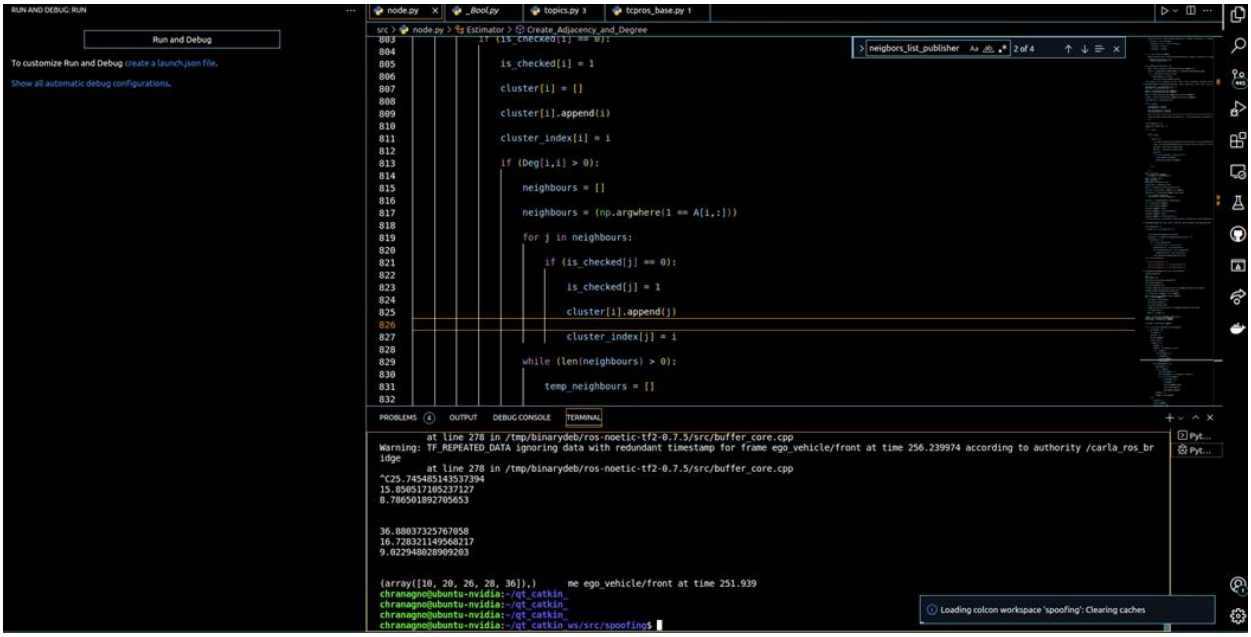


Figure 43: Python code for spoofing detection and mitigation.

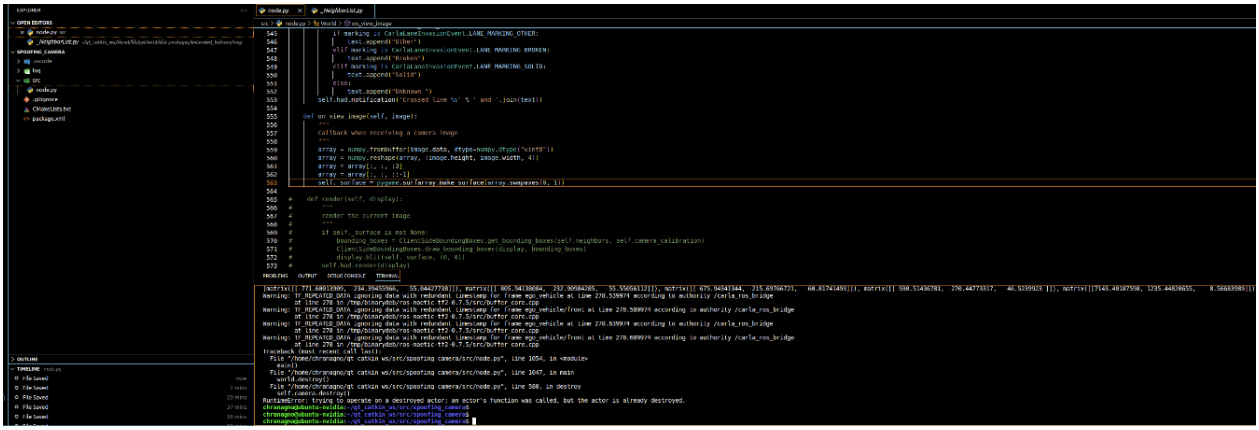


Figure 44: Python code for bird's eye view.

3.1.1.3 *Use case evaluation*

During the simulation, vehicles of the cluster send to the ego vehicle their GPS self-positions, spoofed or not, as well as their relative observations (distances and angles) towards the nearby vehicles. Ego vehicle, along with its measurements, formulates the Laplacian matrix which indicates the connectivity links among the vehicles, and the differential coordinates. The cooperative centralized method **Centralized Laplacian Localization (CLL)** [10] can be adopted by the ego vehicle to estimate cluster's positions highly accurate. However, **CLL**'s performance is seriously degraded when self-positions are spoofed. For that reason, we have developed its robust alternative, **Robust CLL (R-CLL)**, to detect and mitigate the impact of spoofing. See a more detailed algorithmic analysis in the deliverables **D4.3: Situational Awareness Solution based on the Machine Learning Applications**, **D4.4: Report on the Fallback Actions for Minimal Risk Conditions** and [11], [12]. Therefore, initially the demo video begins without spoofing, as shown below:

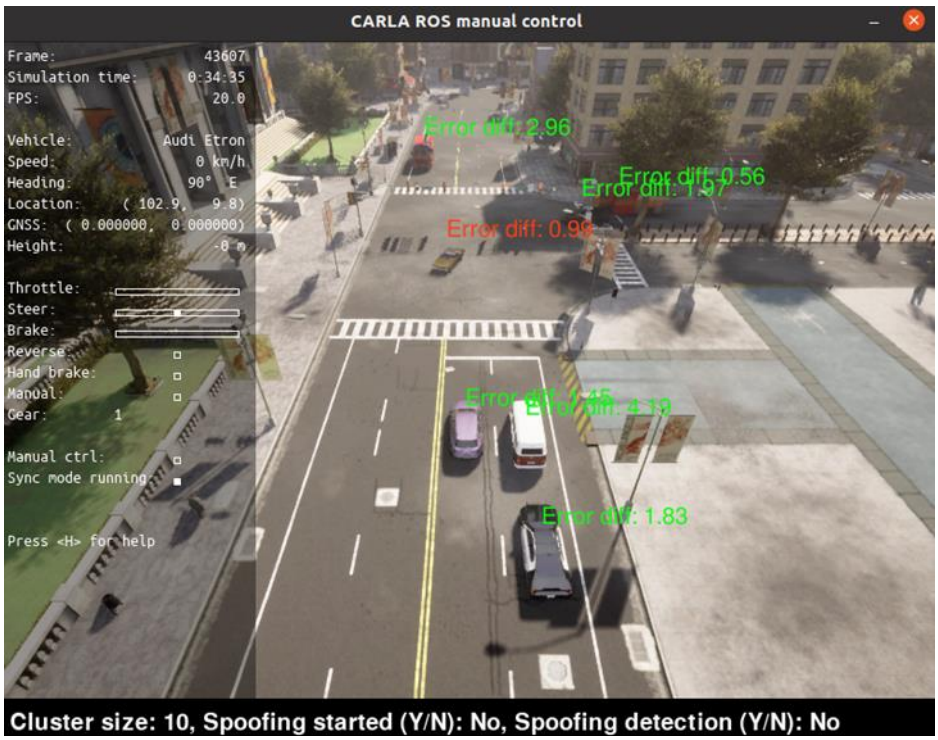


Figure 45: Bird's eye view – No spoofing.

We see from Figure 45, that the cluster consists of 10 vehicles, while the texts above vehicles indicate the benefits (green) or not (red) in meters of **R-CLL** with respect to GPS. Statistical results are provided through the corresponding Cumulative Distribution Functions (CDFs) of self-location error as well as the

root mean square error over cluster's size of vehicles' positions. CDF indicates the probability of the error to be lower than a threshold. Accuracy improves as long as CDF moves to the "left" side of the diagram. The corresponding curves are shown below:

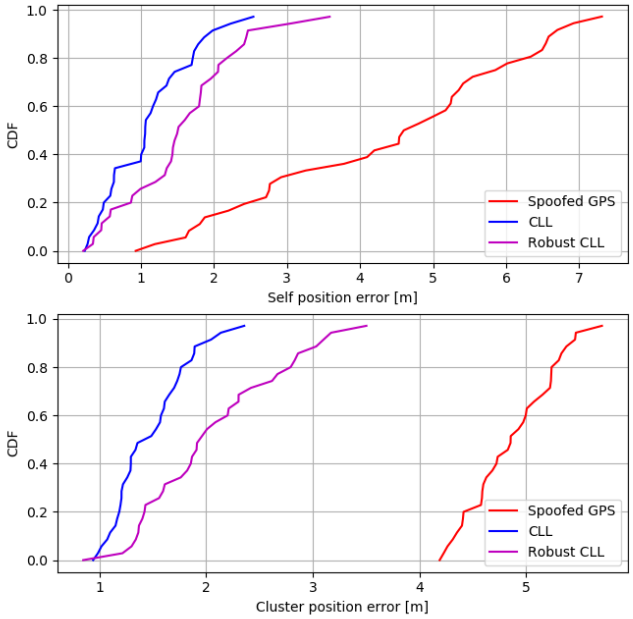


Figure 46: CDF of ego vehicle's and cluster's position error – No spoofing.

In the absence of spoofing, CLL is slightly better than R-CLL. Even in that case, R-CLL reduced self-position error of GPS by 72% and cluster's error by 59%. Additionally, both maximum errors reached almost 3.5m, against 7m and 6m with GPS.

After 200-time instances, spoofing begins and targets 10% of cluster's vehicles, along with ego vehicle. The corresponding figures are shown below:

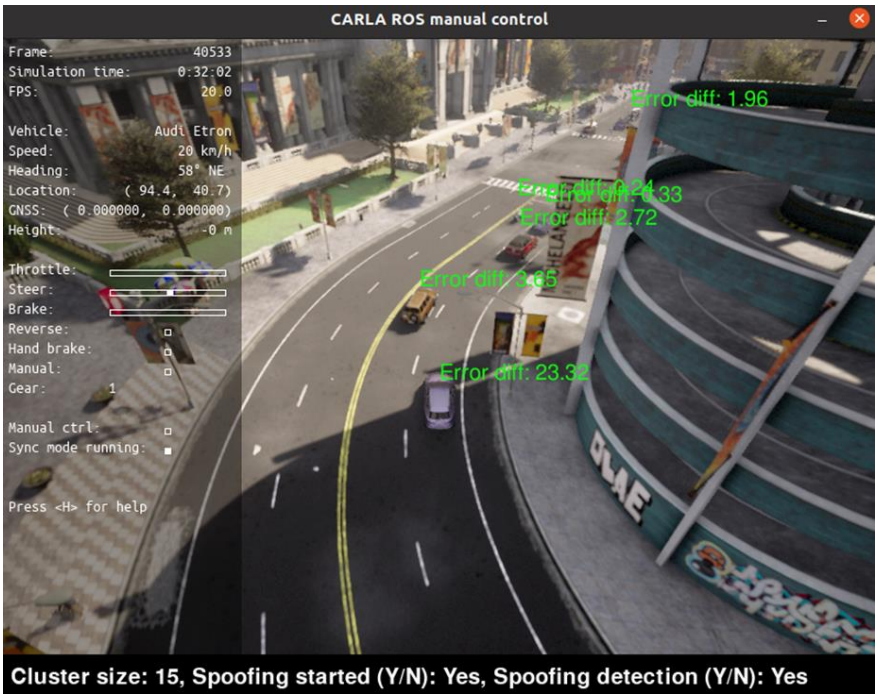


Figure 47: Bird's eye view - Spoofing.

From Figure 47, we see that the cluster consists now of 15 vehicles. An alert of spoofing detection at the ego vehicle is also displayed, and more importantly R-CLL reduces spoofed self-GPS error by 23m, addressing highly efficient the impact of location attack. Figure 48 demonstrates also the superior performance of the developed R-CLL. The latter outperforms CLL in the case of location outliers, both for self and cluster positions. For example, R-CLL achieved maximum cluster position error equal to 4m, against 8m and 16m with CLL and spoofed GPS. Actually, R-CLL reduced spoofed self and cluster GPS error by 93% and 78%.

Therefore, we conclude that CARAMEL's collaborative defense mechanism is highly efficient against location spoofing attacks, significantly mitigating their impact as well as raising alerts when spoofing was detected.

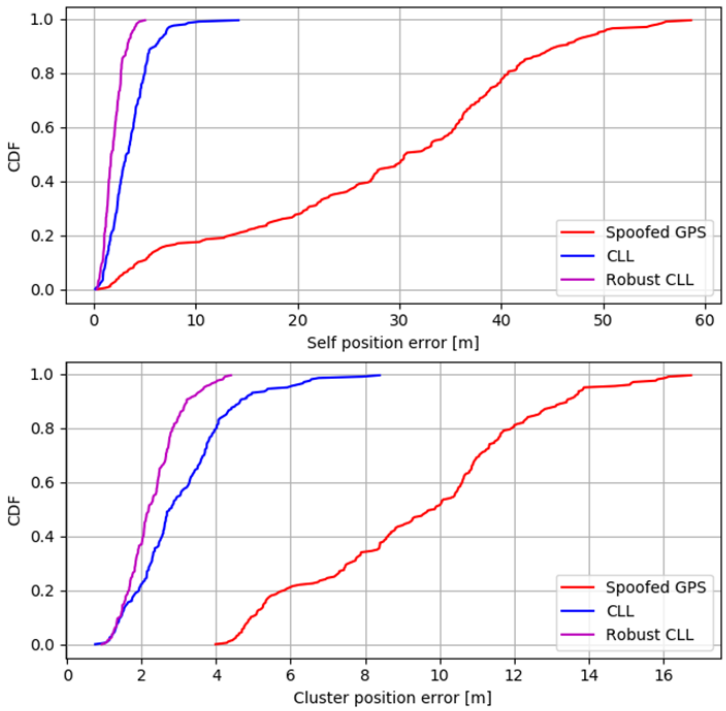


Figure 48: CDF of ego vehicle's and cluster's position error – Spoofing.

3.1.2 Location spoofing attack

Location spoofing attack use case deals with detection of jamming and location spoofing.

A location spoofing attack attempts to deceive a GNSS/RTK receiver by broadcasting incorrect satellite signals, structured to resemble a set of normal satellite signals. These spoofed signals may be modified in such a way to cause the receiver to estimate its location to be somewhere other than where it is.

Civilian GPS signals are unencrypted and therefore present a vulnerability which can be exploited by an attacker. If such an attack is undetected, the vehicle can be steered from its desired trajectory and that can lead to various safety hazards. This use case aims to demonstrate the usability and effectiveness of machine learning algorithms in detecting the anomalies in GPS data. The proposed use case description is shown in Table 11.

Title	Description	Solution Developed for CARAMEL
Location Spoofing Attack	The attacker can jam the satellite signals and the connected car does not have satellite-based location (e.g., GNSS/RTK). The attacker can spoof the satellite-based location of the connected car. CARAMEL system aims to detect the jamming and location spoofing attack.	The TCN model was trained to predict the GPS location of the vehicle at the next time frame. Vehicle measurements and GPS data were used as input features, together with several engineered features. At each time frame prediction of a model is compared to actual value coming from the GPS receiver, if the difference between them exceeds predefined threshold, GPS data is considered as spoofed, and warning is displayed to the driver.

Table 11: Brief overview of the proposed use case for location spoofing attack.

The general high-level overview of the solution, developed for this scenario is described in D3.4, is presented in Table 12:

Type of Algorithms	Description
Pre-processing of GPS data and vehicle parameters from CAN	Obtaining the data from the GPS sensor along with the vehicle parameters to predict the location of the vehicle in the next time step. The data obtained from two sources will go through a pre-processing step.
Spoofing attack detection	After the pre-processing step, the data is sent to the TCN model. At the training session, the network would have been trained with a similar dataset so that it can identify if there is any deviation in the values recorded.
Warning/Alert message	After the Neural Network detects the anomaly, a warning would be displayed to the driver.

Table 12: High-level overview of solution developed for location spoofing attack use case.

For this use case, the following research and development was carried out in the CARAMEL project:

1. Study of literature on existing anomaly detection methods in multivariate time series.
2. Setup of Virtual Environment to generate a dataset and test the model.
3. Development of Temporal Convolutional Network for predicting the vehicles position.
4. Testing robustness of the developed model.

3.1.2.1 *Pipeline Demonstration*

3.1.2.1.1 Introduction

The pipeline for demonstration of location spoofing attack is done according to Figure 49.

VTD is a simulation software which provides the environment and sensor readings. The environment consists of roads, road marks, signs, pedestrians, and vehicles. VTD sends vehicle position and speed parameters to the Model.CONNECT, where complete vehicle dynamics are simulated. During demonstration HackRF is used to perform location spoofing attacks by sending fake GPS data to the U-Blox GPS receiver, which passes it to Model.CONNECT via Python FMU. Model.CONNECT allows us to run Python script with a pretrained TCN detection model in a co-simulation environment. On each time frame detection is performed. In case of detected anomalies, a warning is displayed on the dashboard.

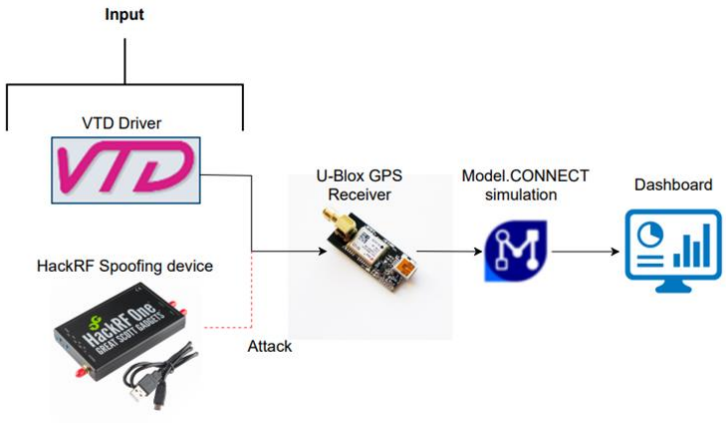


Figure 49: Overview of the pipeline for location spoofing attack use case demo.

3.1.2.1.2 Virtual Environment

Virtual environment used for demonstration purposes was created in the VTD scenario editor. ODB map files from AVL's database were used to create scenarios and generate approximately 20 hours of driving data. Due to different driving profiles on different types of roads (e.g., highway, city and rural roads), scenarios were carefully chosen to represent all the types equally.

Maps of Graz and surrounding areas were used to represent urban and rural areas, together with Austria's highway maps to represent highway driving scenarios. In total, 19 hours of generated data were used to train the model and one hour was used to perform the testing. Figure 50 shows the top-down view of one of the highway scenarios which was created for the use case. Figure 51 depicts the environment from the main actor's (Ego vehicle) point of view.

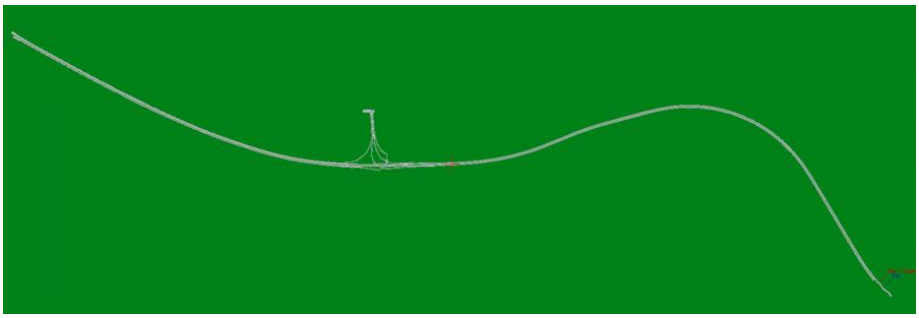


Figure 50: Top-down view of scenario in VTD (section of Phyrn Autobahn).

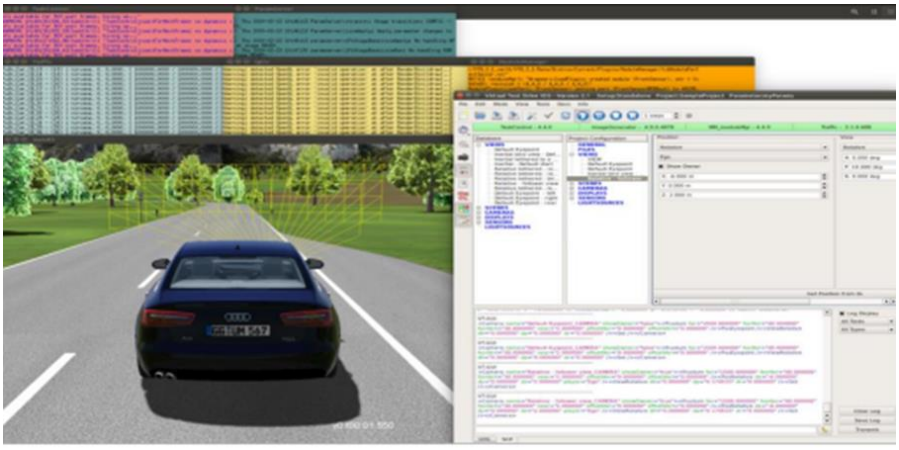


Figure 51: VTD simulation environment.

3.1.2.1.3 Temporal Convolutional Network (TCN) Model

Two states of the art neural networks for working with sequential data were considered, Long Short-Term Memory (LSTM) and Temporal Convolutional Network (TCN). Both resulted in similar performance in terms of evaluation metric – Root mean square error (RMSE). However, due to lower computation times we decided to proceed with TCN. TCN is a type of convolutional neural network, which was designed specifically for working with sequential data. It was developed as an attempt to use Convolutional Neural Networks (CNN) in the field of sequence modelling where Recurrent Neural Networks (RNN) have long been the state of the art. Results of recent studies suggest that TCNs convincingly outperform baseline recurrent architectures across a broad range of sequence modelling tasks [13]. TCN uses the concept of dilated causal convolution, which enables the network to consider multiple previous time steps when making a prediction. Figure 52 shows a dilated causal convolution with different dilation factors and kernel size k = 2.

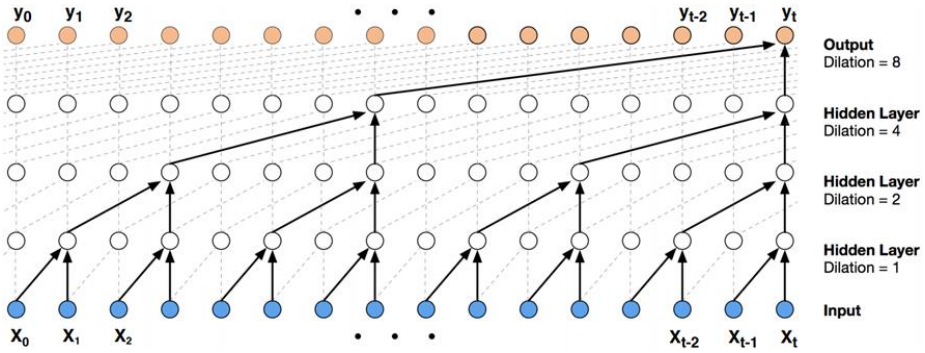


Figure 52: Dilated causal convolution.

Detection of anomaly was modelled as regression. The output of TCN is therefore a prediction of GPS data in the next time frame. RMSE was used as a performance metric. For more information on TCN and its implementation refer to deliverable D3.4.

3.1.2.1.4 Use case test setup

Main purpose of the test case is to demonstrate the usability and accuracy of machine learning models for location spoofing attack detection. Various scenarios will be used, to represent different types of driving profiles and their effect on spoofing detection. Simulation will be run using predefined scenarios. At a given time instance the location from the HackRF will be switched on and will replace the GPS data from the simulation to cause an anomaly in the data. If the test is successful, the TCN model will

recognize the anomaly and inform the operator that the vehicle's GPS is no longer reliable, by displaying a warning on the dashboard. In the simulation environment this warning will be displayed directly in the Model.CONNECT dashboard, which allows us to track the state of the vehicle in real-time during simulation. Figure 53 depicts the flow of spoofing attack detection in GPS in both attacked and normal scenarios.

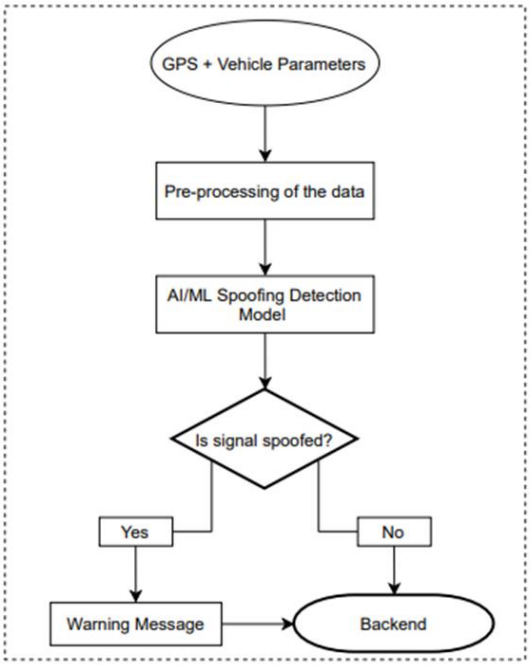


Figure 53: Flow of spoofing attack detection in GPS in attacked and normal scenario.

3.1.3 In-vehicle Location Spoofing Attack Detection

3.1.3.1 *Introduction*

The in-vehicle solution against GPS location spoofing attacks fuses multi-source data, readily available from the CAV's on-board sensors. In the prediction phase, multi-sensory data collected through the OBU and/or the CAN bus are used to compute the CAV's predicted location in the next time step given the previous CAV location estimate. This is achieved by projecting the location ahead of time using the sensor readings and a CAV mobility model. In the update phase, the CAV location measurements, provided by a GPS-free localization algorithm (e.g., based on cellular networks), are used to update the predicted location utilizing Bayesian filtering and derive a refined location estimate. Finally, in the attack detection phase, the CAV location provided by the GPS receiver is compared to the refined location. If their deviation, e.g., Euclidean or Bhattacharyya distance, exceeds a threshold, then an attack is signified.

The processing pipeline that implements the in-vehicle location spoofing attack detection solution is depicted in Figure 54. We simulate the moving vehicle, the onboard sensors and the real-time sensor data collection using the CARLA simulator and ROS integrated in a workstation PC. The embedded device (e.g., Jetson Nano) hosts both the Threshold selection algorithm and the Attack detection algorithm that comprise the attack detection solution. The attack detection threshold is selected during the Training stage, where the vehicle is moving under attack-free conditions (normal case), and then used to detect attacks in the Testing stage when location spoofing attacks are present (attack case).

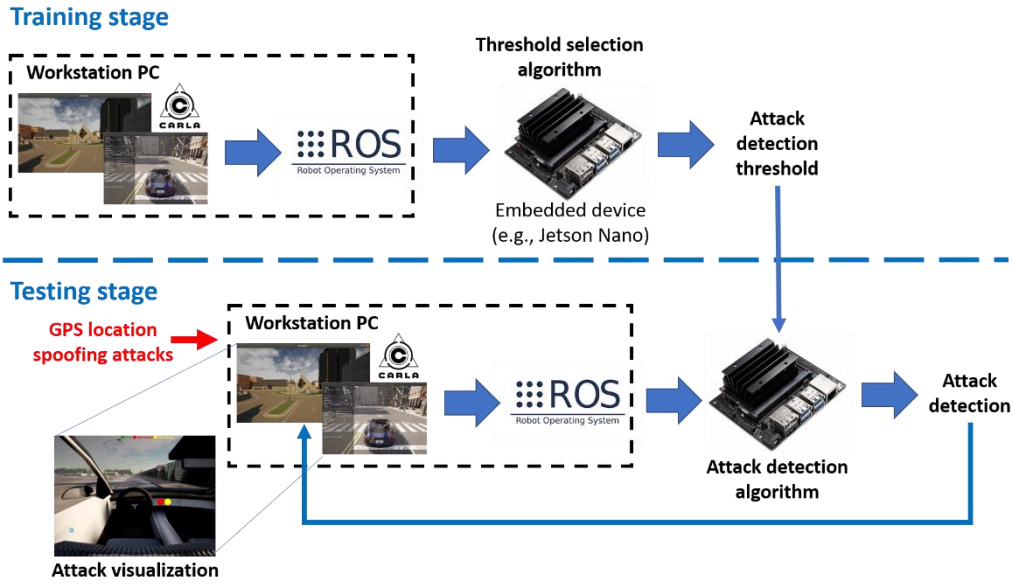


Figure 54 Processing pipeline of the in-vehicle location spoofing attack detection solution.

3.1.3.2 *Use-case setup*

The setup for the demonstration of the location spoofing use-case is illustrated in Figure 55, where the attack detection solution is coupled with the testbed setup. In particular, the simulation environment (i.e., CARLA and ROS) will be installed and running on an on-site laptop. The detection solution including the Threshold selection and Attack detection algorithms will be running on an embedded device (e.g., Jetson) that will emulate the Antihacking Device (AD) installed in the car. In the future, as part of a production-level solution, the AD would send an alert signal to the OBU inside the car in case an attack is detected to propagate it further to the backend/MEC for awareness and/or possible mitigation actions, e.g., revoking the PKI certificates of attacked vehicles.

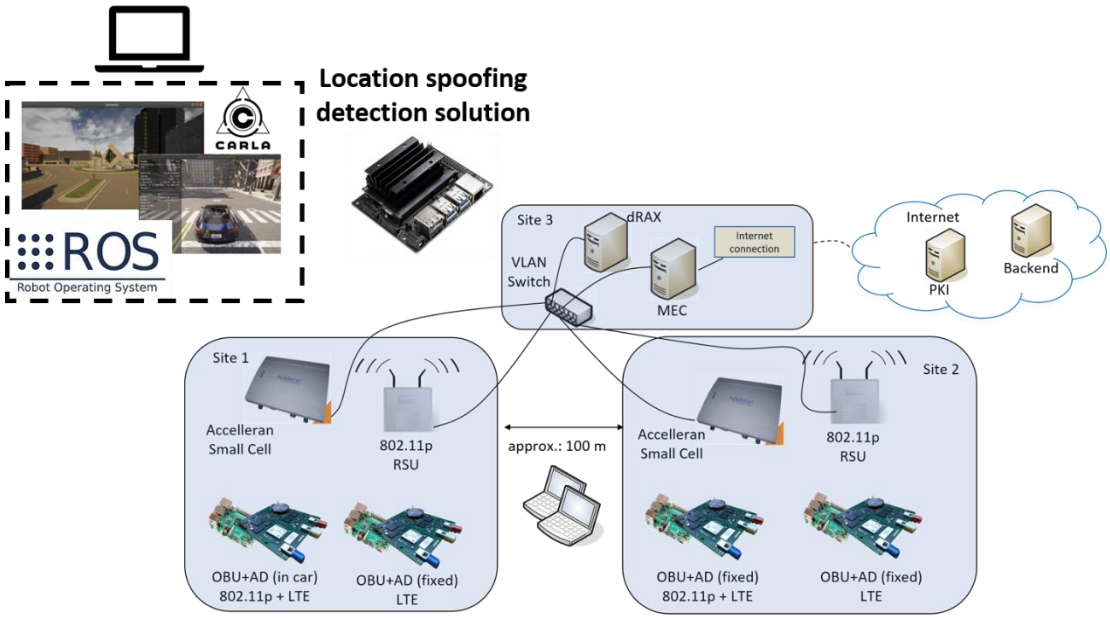


Figure 55 Setup for the location spoofing attack use-case.

3.1.3.3 *Use-case workflow and evaluation*

At the Training stage, the following workflow will be used:

1. A moving vehicle will be simulated in the simulation environment driving under normal conditions (i.e., no attack) along a predefined trajectory with user-selected noise profiles for the simulated GPS data and the GPS-free estimated vehicle locations (e.g., based on cellular networks).
2. The simulated sensor data will be collected and processed by the embedded device to compute the attack detection threshold.

At the Testing stage, the following workflow will be used:

1. A moving vehicle will be simulated in the simulation environment driving under mixed conditions (i.e., switching between normal and attack) along a predefined trajectory with similar noise profiles for the simulated GPS data and the GPS-free estimated vehicle locations.
2. The GPS location spoofing attack will be simulated by adding a user-selected constant bias to both GPS location coordinates.
3. The attack detection threshold computed in the Training stage will be provided as input to the Attack detection algorithm running in the embedded device.
4. When no attack is present, a 'green' light will be turned on within the vehicle in the simulation environment to indicate that the vehicle is driving under normal conditions.
5. When a location spoofing attack is present, the 'green' light will switch to 'yellow'. Subsequently, a 'red' light will turn on next to it to signify that the detected has been successfully detected, as shown in the left part of Figure 56.

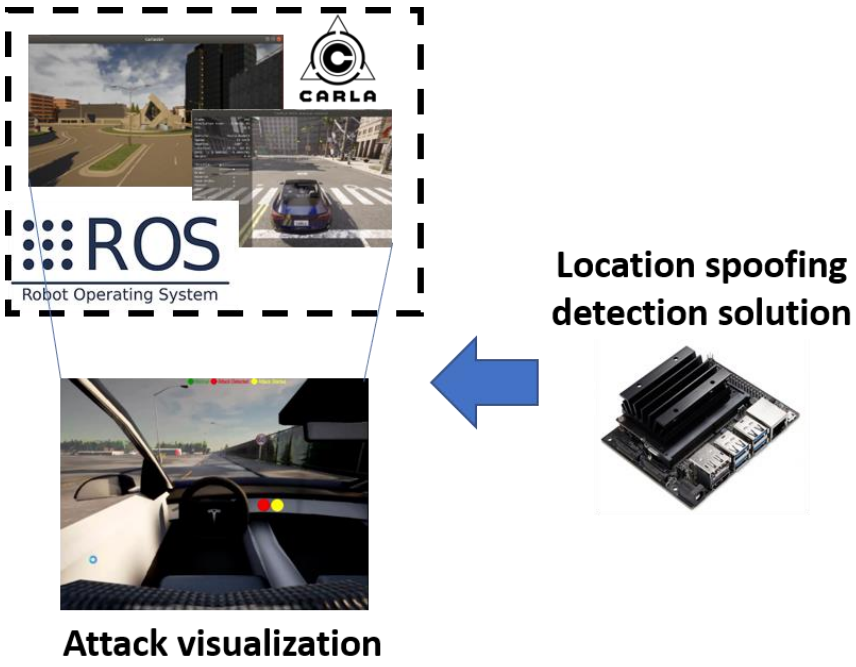


Figure 56 Visualization of the attack in the GPS location spoofing use-case workflow.

3.2 *Attack on the V2X Message Transmission: attack use case description and assessment*

3.2.1 Attack Description (UC2.2)

In this use case we demonstrate two operations:

- The interoperability between different radio technologies.
- The attack on the V2X Message Transmission.

The interoperability between different radio technologies is necessary when vehicles in the same region are using On-Board Units (OBUs) equipped with different standardized access technologies. Currently, it is possible to have OBUs with a single cellular connection (LTE-Uu), or others with this cellular connection plus a Vehicle-to-Vehicle (V2V) technology such as IEEE 802.11p, LTE-PC5, the new NR-PC5 or the not yet standardized IEEE 802.11bd.

As it has been described in previous deliverables, CARAMEL's testbed consists of two types of vehicles, the "LTE-Uu only" and the "LTE-Uu + 802.11p". To perform the interoperability, we deploy a fixed infrastructure consisting of an 802.11p RSUs network and a small private LTE network, both connected to a Multi-access Edge Computing (MEC) which, using different kinds of policies, forwards messages from a radio technology to the other.

The attack on the V2X message transmission consists of two different kinds of attacks:

- A malicious attacker transmits fake V2X messages. This is demonstrated in the testbed.
- A malicious attacker tries to track a specific vehicle. This is demonstrated by simulation.

CARAMEL addresses 5 types of fake V2X messages which are detected in the OBU and/or in the MEC:

- **Non-Signed messages:** When this event is detected the message is dropped.
- **Messages signed with a non-valid certificate:** This is the case where the certificate used to sign messages is not issued by a valid Certification Authority. When this event is detected, the message is dropped, and an alarm is triggered.
- **Non-authorized messages:** This is the case where an OBU of a "passenger car" transmits **V2X** messages specifying that its vehicle type is an "emergency vehicle". The Authorization Ticket of this car has been issued in such a way that the receiver detects the anomaly. When this event is detected, the message is dropped, and an alarm is triggered.
- **Replayed messages:** This is the case where an OBU captures a message transmitted by another OBU and retransmits it. A third receiver may receive both copies. When this event is detected, the message is dropped. It is also possible that the receiver only receives the replayed message. This case only represents a security problem if the message has been modified, which will be detected by the digital signature. Additionally, if the replayed message is received later than a threshold delay time, the message is dropped. An alarm is not triggered because this replayed message does not present any security problem, and it could also be replayed by the fixed infrastructure, which is, in fact, a compliant action.
- **Messages signed with a revoked certificate:** When this event is detected the message is dropped and an alarm is triggered.

In all previous cases, whenever an alarm is triggered, the backend receives a notification. The backend monitors alarms and performs statistics for management purposes.

The case of a malicious attacker trying to track a specific vehicle by sniffing its transmitted messages represents a passive attack. CARAMEL has developed an algorithm that computes when the best time is to change the vehicle's AT, trying to mimetize itself among multiple neighbor vehicles. To show how this algorithm works requires a relatively high number of vehicles, and it cannot be demonstrated in the testbed, for this reason it will be demonstrated using a simulation. In any case, the software to implement this algorithm has been integrated inside the anti-hacking device, but it is not being executed during the demonstration.

Figure 57, extracted from D2.4 "System Specifications and Architecture", shows the involved processes and actors.

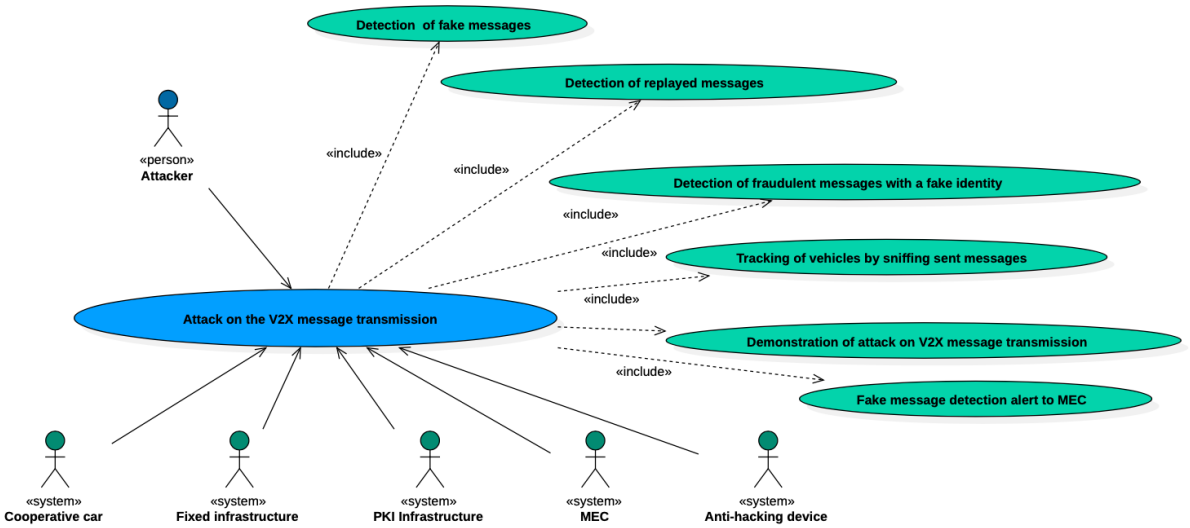


Figure 57: Use case attack on the V2X Message Transmission (UC2.2).

Involved actors:

- **Attacker:** In the demonstration setup, the attacker is represented by an OBU connected by Wi-Fi to a laptop, from where it is possible to activate the generation of the following fake messages: non-signed messages, messages signed with a non-valid certificate, messages signed with a revoked certificate, replayed messages and non-authorized messages.
- **Cooperative car:** In the demonstration scenario, the cooperative car is represented by an OBU plus its anti-hacking device. It constantly transmits Cooperative Awareness Messages (CAM), checks the validity of the received ones and, in case of detecting a fake message, drops it and, optionally, triggers an alarm. It is also connected to a laptop, from where it is possible to activate and to stop the process of CAM transmission, and visualize the messages received from other cooperative cars.
- **Fixed infrastructure:** In the demonstration setup, the fixed infrastructure is composed of two RSUs and one LTE small cell, all of them connected to the MEC using a VLAN capable Ethernet switch. Each RSU, covers a small geographical area. As small cells are supposed to cover larger areas, the small cell of the testbed covers the whole testbed area. These areas are used by the forwarding algorithm to showcase the forwarding of messages based on Region of Interest (RoI). Additionally, the Accelleran's small cell requires the Accelleran's dRAX™ Open Interface Radio Access Network (RAN), which provides an intelligent virtual RAN controller and it is implemented in a computer also connected to the Ethernet switch.
- **PKI infrastructure:** In the demonstration setup, the PKI infrastructure is implemented using servers in ATOS network which are reachable using a standard Internet connection. It is composed of multiple virtual containers executing the following processes: the forwarding of the certificates identifying the Root, Enrolment, Authorization and Revocation Authorities, the enrolment process, the authorization process and the revocation process. At the end of the whole process, the PKI client, executed in each cooperative car, obtains a group of ATs which are used to sign V2X messages.
- **MEC:** It is a computer with virtual containers executing the following processes: one V2X communication protocol stack for each RSU and the LTE network, the virtual Evolved Packet Core (vEPC) of the LTE network, the V2X forwarder, the Local Dynamic Map (LDM), the MQTT broker, the application which decides if a cooperative car will have its certificates revoked and application responsible of the Certification Revocation List (CRL) distribution.
- **Anti-hacking device:** One anti-hacking device is directly attached to each OBU to execute those processes that, due to their computational requirements, cannot be executed in the OBU. Specifically, these processes are the MQTT broker, the high-level applications for the demo test, the PKI client, the tracking avoidance algorithm, the alarm detector, and the alarm notification.

3.2.2 Use case setup

The testbed is deployed in Panasonic premises in Langen and is composed of communication devices and servers reachable through Internet (Figure 58):

- **RSU-1:** It covers one specific area with IEEE 802.11p.
- **RSU-2:** It covers another specific area with IEEE 802.11p. As we intend to show how vehicles from one area receive messages from vehicles of the other area through the infrastructure, and the test site is quite small, if all vehicles used the same channel, as it would be in a real case, all vehicles would see each other directly. To avoid this situation, we configure vehicles and RSU of both areas with different channels. So, the only option they have to receive messages is through the infrastructure.
- **Accelleran small cell:** It provides LTE coverage to the whole testbed area.
- **MEC:** It receives all messages transmitted by the vehicles and forwards them appropriately. It also deploys the core part of the LTE network and provides Internet connection to vehicles through this LTE network.
- **dRAX:** It is a specific server devoted to control Accelleran small cells. It is necessary to deploy the LTE network.
- Two ethernet switches connect all the infrastructure devices.
- Two types of vehicles: Vehicles with IEEE 802.11p and LTE radio interfaces, and vehicles with only one LTE radio interface. Each vehicle deploys one OBU and one anti-hacking device (AD).
- **PKI servers:** The PKIs servers will be reachable through Internet connection.
- **Backend monitoring server:** This server is reachable through Internet connection and receives all alarms triggered by the vehicles to perform monitoring functions.
- **Backend LDM server:** This server is used only for demonstration purposes. In a real case, every vehicle will continuously update its Local Dynamic Map (LDM), which is the data base for all Intelligent Transportation System (ITS) applications. In the testbed, all vehicles transmit the information acquired through V2X messages to this server and, using a frontend application developed in this project, it is possible to show what is happening in each vehicle by simply accessing this server from any computer.

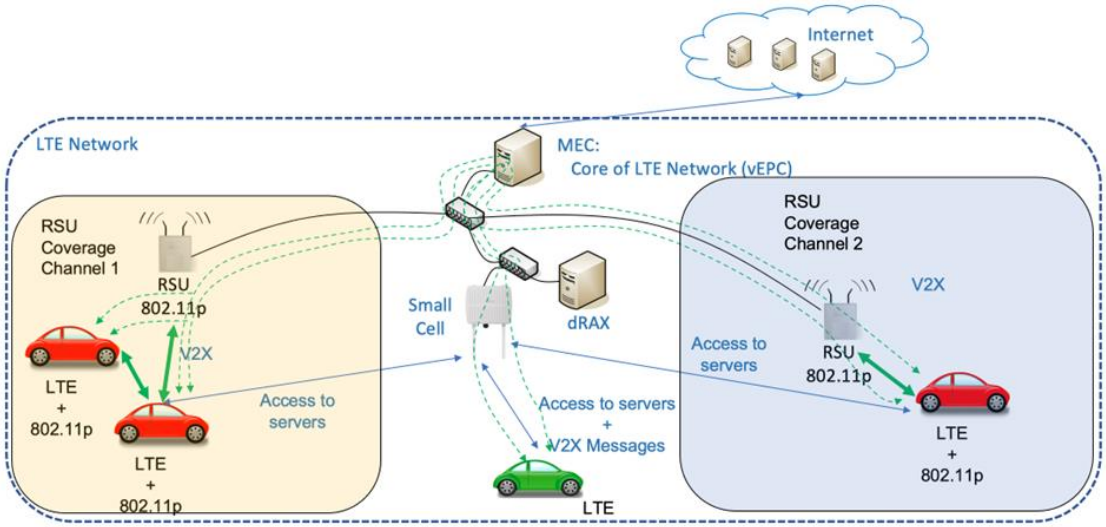


Figure 58: Testbed layout.

Requirements:

- Accelleran small cell and RSUs use Power Over Ethernet (POE). For the other devices it is required to have electrical power available.
- Ethernet cabling.
- Internet connection in the MEC. It is provided by a commercial LTE router.
- Regulatory approval for mobile network use by the German Agency for networks (RegTP), for the specific test frequencies, area, duration, and geographic area. 10 MHz of bandwidth in band B43.

3.2.2.1 *Local Dynamic Map server*

In order to perform demonstrations, the CARAMEL project has developed a cloud user interface backend to gather all the information regarding the V2X communications in the OBUs and show it on a web-based frontend. The OBUs maintain a permanent communication with the backend using their LTE interface.

Frontend Layout

The idea behind the design of the Frontend is to enable a global view, as well a specific view, of the V2X communications in a region and to be able to trigger attacks. The frontend has two working states:

- General view of the whole environment: It is the initial view where all nodes which are sending information to the backend are listed and drawn on a map in grey (Figure 59). They are located on the map according to their position obtained by their GNSS receiver.

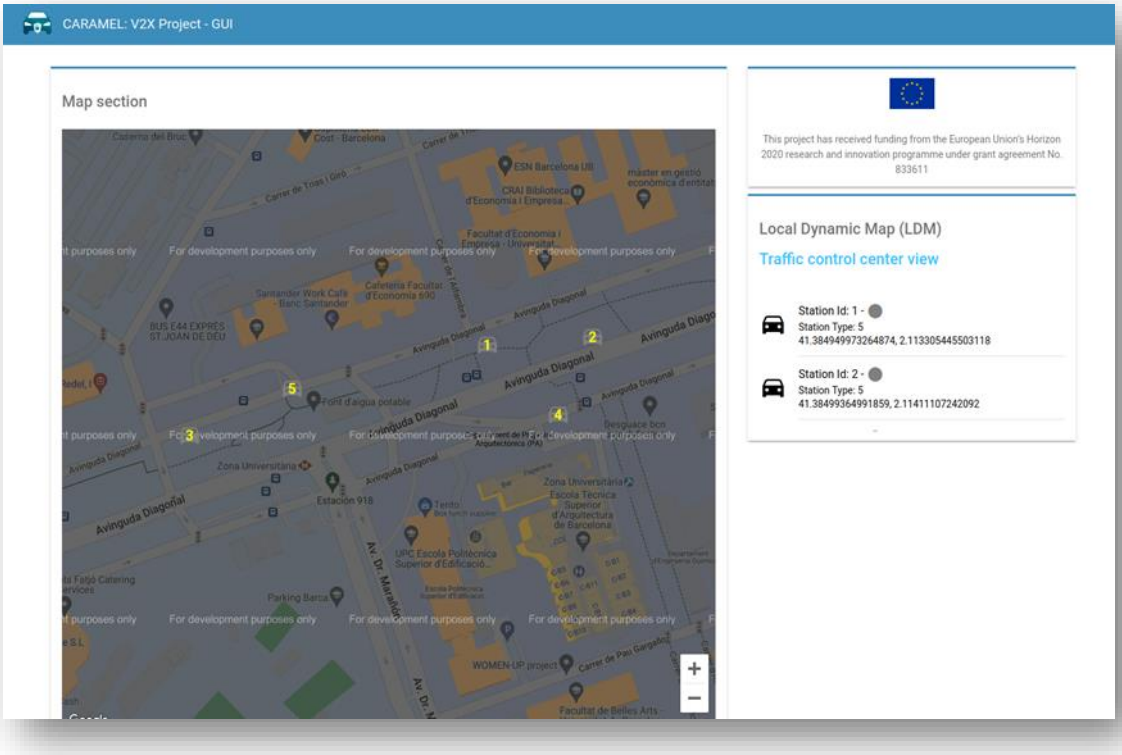


Figure 59: Global view of the frontend of the LDM server.

- Specific view of a vehicle: To see the V2X "sight" of a specific car (Figure 60), click the grey button on the right of its name on the list of nodes of the general view. Then, the selected node turns blue, and the surrounding nodes turn to different colours from green to orange or red. The colours of the surrounding nodes describe their status regarding the V2X communications security as they are perceived by the selected node. If the colour is green, it means that the security is completely fine, whereas orange and red mean that an anomaly or an attack has been detected.

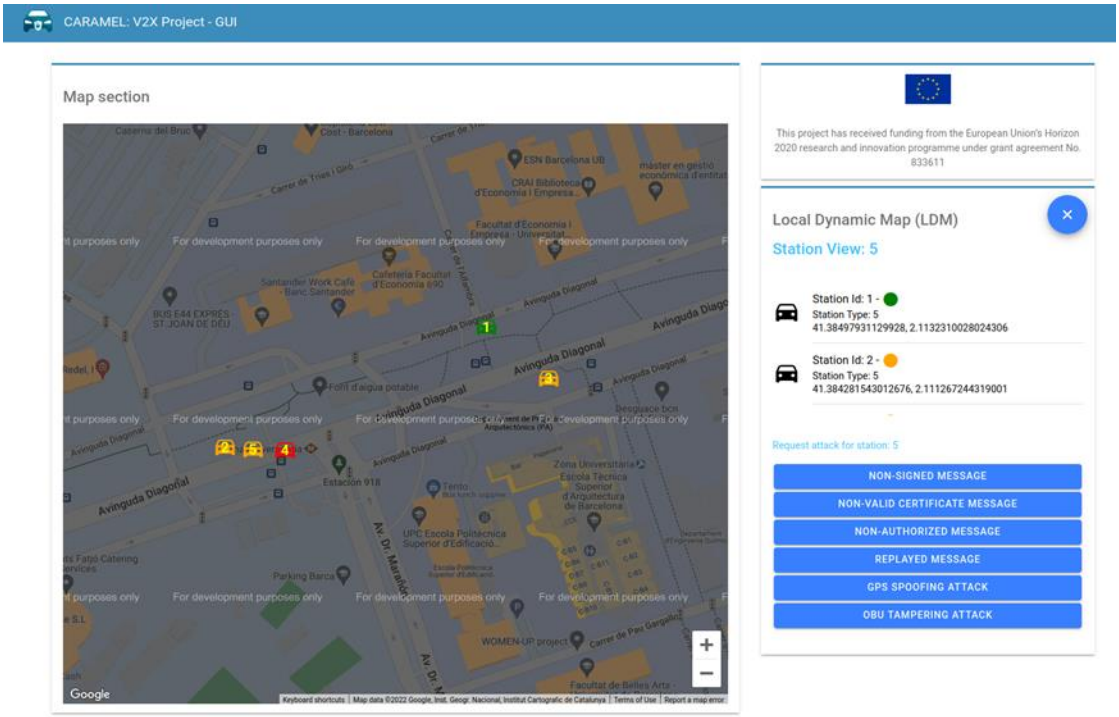


Figure 60: Specific view of the frontend of the LDM server.

Be aware that when we have a node selected, we are seeing its "view" through V2X communications, and some of the vehicles seen on the global view might disappear. This means that there is no V2X communication between these two nodes.

Finally, using the blue buttons that appear in the specific view of a node, it is possible to trigger attacks. If an attack is launched, it is possible to see that has been detected in the other nodes, if the color of the attacker in the other vehicle's specific view, changes to orange or red.

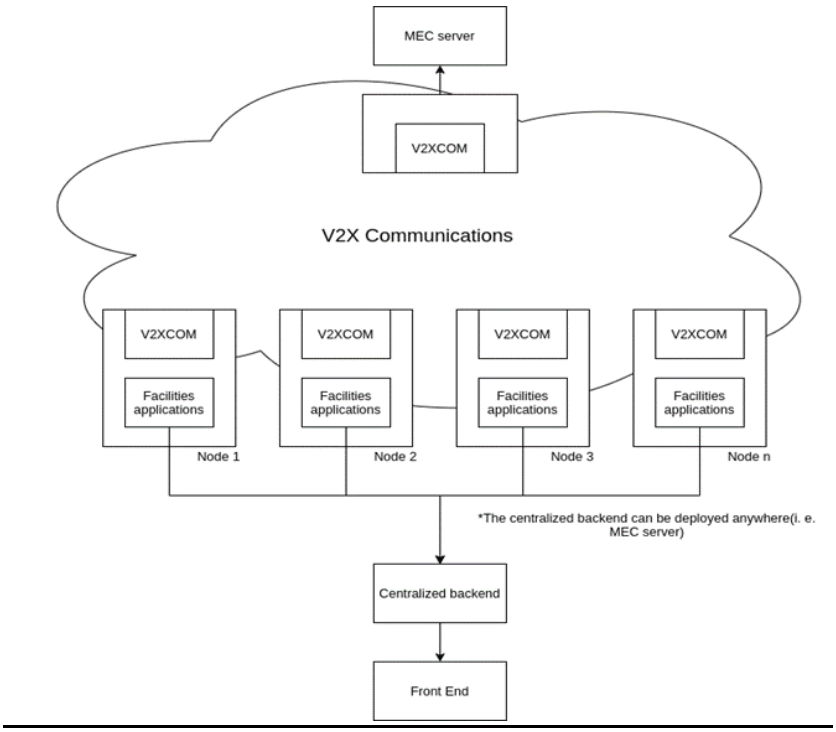


Figure 61: Architecture layout of the LDM server.

Backend module

The backend module runs on a virtual machine in the cloud. Its function is to gather all of the information from all the involved nodes, through TCP connections. Once the information is stored in the backend, it is served to the frontend, and it will be accessible to any client that might connect through WebSocket. The general architecture layout can be seen in Figure 61.

The communication between the backend and the OBU is done through a software module running at the Application/Facilities layer, being executed in the anti-hacking device, and using the LTE interface of the OBU to access Internet.

The centralized backend waits for IP/TCP client connections. Once this connection is established, the client continuously publishes all gathered information using a JSON structure with the following format:

```
{
  "station_id":2,
  "type":"latest|all",
  "position": {
          "latitude": 42.1,
          "longitude": 2.3
      } <-- Latest position gathered by the GPS Node
  "packets":[
    {
      "timestamp": 123456
      "type":"cam",
      "validity": true,
      "error":"error_type",
      "certificate":"hashedId8_certificate",
      "station_id": 2
      "message":"cam without header in xml"
    }
  ],
  "certificates":[
    {
      "hashedId8":"hashedid8 cert",
      "certificate":"cert in xml",
      "validity": True,
      "Error": "Error if not valid"
    }
  ],
  "alarms":[
     "id": 12,
     "message": "message of the alarm"
  ]
}
```

Where the meaning of the fields is:

- Station ID: Station ID of the node (published in the header of CAMs).
- Type: It can be "latest" or "all". Type "all" means that we are dumping all of the LDM and received messages to the centralized backend. Type "latest" means that we are updating the previously uploaded information.
- Packets: The received packets are placed in this list.
- Certificates: All the received certificates, with their corresponding validity and hash.

The centralized backend has to process all the information received by the nodes in order to be able to deliver the appropriate information to the front end.

The communication between the centralized backend and the frontend is done through WebSockets. The reason behind this choice instead of REST API is based in the fact that there is a continuous flow of information from the backend to the frontend. And, by using WebSockets, the continuous flow of information is possible without having to request each chunk of information.

On opening up the app, the layout prints all existing nodes (without taking into account the V2X sight). This means that all the nodes are drawn on the map. To get the information of all the nodes, the front-end connected to the backend through a WebSocket, uses the following petition:

```
{
  "request" : "general"
}
```

The response from the backend will be to periodically sent the list with all of the nodes and relevant information for the first-time printing. The refresh frequency is around one response per second. The transmissions of these messages will not stop unless a new request is sent or the communication is closed.

```
{
  "response": "general",
  "content": [
    {
      "station_id": 1,
      "station_type": 5 // <- The same enumerated as the CAM standard
      "position": {
        "latitude": 42.1
        "longitude": 2.3
      }
    }
  ]
}
```

When the end-user requests the specific view of one node, the following request is triggered:

```
{
  "request":"sightof",
  "content":{
    "station_id": 3 <- Station ID of the selected node
  }
}
```

The response only considers those vehicles seen by this node and displays them according to the colour code: green means all received packets are correct, orange means that generally all of the packets received are fine, but a few have been detected as problematic and red means all received packets are wrong, either because it's inconsistent or because is revoked.

```
{
  "response": "sightof",
  "content": [
      {
      "station_id": 1,
      "station_type": 5 // <- The same enumerated as the CAM standard
      "position": {
        "latitude": 42.1
        "longitude": 2.3
            "state": "green|orange|red|blue",
      "packets" : [ <- Last 10 packets received by the node
          {
          "timestamp": 123456
          "type":"cam",
          "validity": true,
          "error":"error_type",
          "message":"cam without header in xml"
```

```
          }
        ]
      },
    ]
  }
}
```


"sightof" responses are periodically transmitted, and only stop when another "sightof" or "general" request is issued, or the communication is closed.

3.2.3 Use case workflow

The demonstration is divided into four parts. Firstly, we check that all preconditions for the normal transmission of V2X communication between cooperative cars are working properly, then, we reproduce all attack and interoperability use cases.

3.2.3.1 *Initial requirements to set up the V2X communications testbed*

Subsystem	Operation	Success end condition
PKI servers	● Authorization certificate authority is reachable and signs ATs ● Enrolment authority is reachable and enrolls registered vehicles ● Authorization authority is reachable and authorizes vehicles according to their context ● Revocation authority is reachable and keeps track of revoked ATs	PKI servers are operative
Backend server	● Backend server is reachable - able to receive alarm notifications from the MEC and from vehicles - and displays alarms statistics	Backend server is operative
VLAN Ethernet	● VLAN Ethernet switch is configured with 4 VLANs as in Figure 62	VLAN Ethernet is operative
MEC	● MEC is configured with 8 virtual containers connected as in Figure 62 ● MEC is connected to Internet ● “V2XCom-11pRSU” modules receive V2X messages from the RSUs and push them to the MQTT broker ● “V2XCom-11pRSU” modules are subscribed to their forwarding topic of the MQTT broker, receive V2X messages from these topics and forward them to their respective RSUs ● “V2XCom-CV2XRSU” module receives V2X messages from the LTE network and pushes them to the MQTT broker ● “V2XCom-CV2XRSU” module is subscribed to its forwarding topic of the MQTT broker, receives V2X messages from this topic and forwards them to the LTE network ● All “V2XCom-*” modules check the signature of incoming messages and drop all non-compliant messages	MEC is operative

	● "MQTT broker" is configured with all necessary topics and distributes messages ● "V2X forwarder" module is subscribed to all incoming V2X messages MQTT topics and with their new information updates the LDM (Table 14) ● The forwarding rules table is configured with forwarding rules related to Region of Interest (RoI) and type of radio technology ● "V2X forwarder", on reception of new messages, forward them to cooperative cars according to the forwarding rules table ● "Certificate Revocation Decision" module receives alarms from OBUs ● "Certificate Revocation Decision", the algorithm to decide if an alarm is critical enough to revoke the certificates of a vehicle is out of the scope of the project. For demonstration purposes, we will activate the revocation manually. ● "Certificate Revocation Decision" is able to inform the "Certificate Revocation List Distribution" module about revocation decisions	
RSUs network	● Two RSUs are connected to the Ethernet network with their respective VLAN ● Two RSUs are forwarding V2X messages to and from 802.11p and Ethernet interfaces	RSUs network is operative
LTE network	● The small cell is connected to the Ethernet network within the same VLAN, which connects them to the MEC and to dRAX ● dRAX is operating and controlling the small cell ● vEPC is operating in the MEC and is has Internet connectivity ● vEPC provides connectivity to the OBUs while they are accessing to Internet and to the "V2XCom-CV2XRSU" module in the MEC	LTE network is operative
OBU "LTE-Uu only"	● OBU is connected to the anti-hacking device as in Figure 63 ● OBU can reach the MEC, PKI servers and backend through the LTE-Uu interface ● OBU is able to enroll, to be authorized and download ATs from PKI servers using the "PKI client" ● OBU is able to store private keys in the HSM ● OBU is able to generate, transmit and receive signed V2X messages through the LTE-Uu interface ● OBU is able to detect non-compliant V2X messages ● OBU is able to detect a tampering attack ● "Alarm notification" module is able to transmit alarms to the Backend ● "Traffic avoidance" module is able to choose at what time is best to change the AT and notify the "PKI client" to do so	OBU transmits and receives signed V2X messages

	• The “Facilities applications” (Demo backend client) module is able to reach the backend LDM server to show results to the user and to trigger attacks • The user controlling the demonstration testbed through a laptop is able to connect with the backend LDM server	
OBU “LTE-Uu + 802.11p”	• As the OBU “LTE-Uu only”, but the transmission and reception of the signed V2X messages uses the interface 802.11p instead of the LTE-Uu	OBU transmits and receives signed V2X messages

Table 13: Check list of initial requirements to set up the V2X communications testbed.

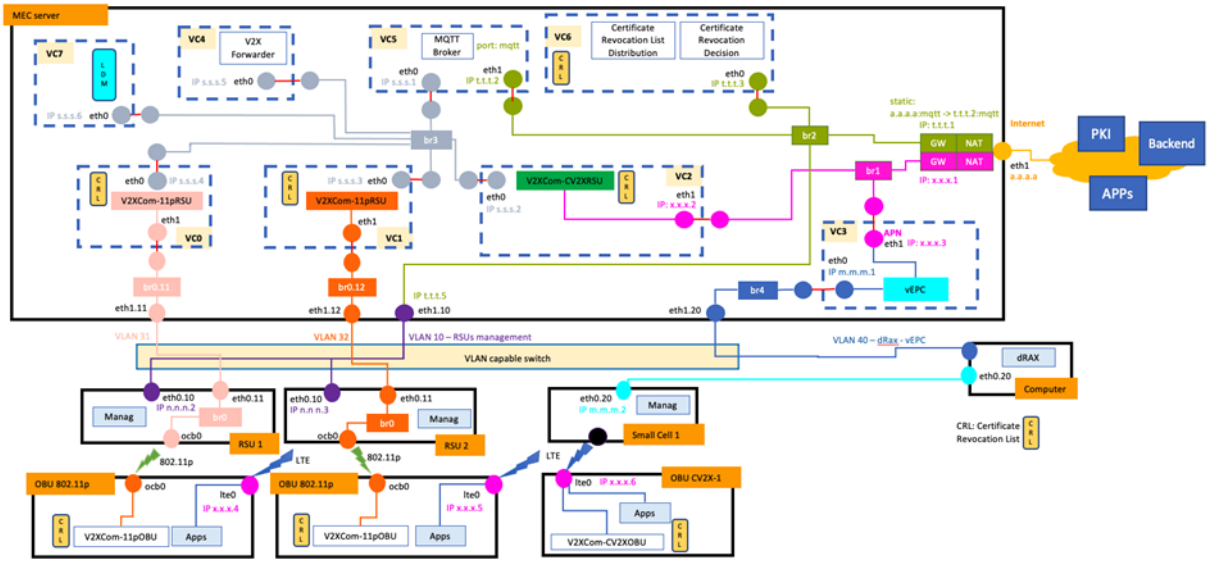


Figure 62: Architecture of MEC modules and connections.

Component	Parameters
RSU	• RSU identification • Coverage area
Neighbor cooperative car	• Station ID (can be a pseudonymous and change along time) • Geographical coordinates • Flag: 802.11p/LTE • RSU ID at which is attached (only for 802.11p connected OBUs) • IP address (only for LTE connected OBUs) • Flag: Valid AT / Non valid AT • Last seen timestamp • Type of vehicle

Table 14: LDM structure.

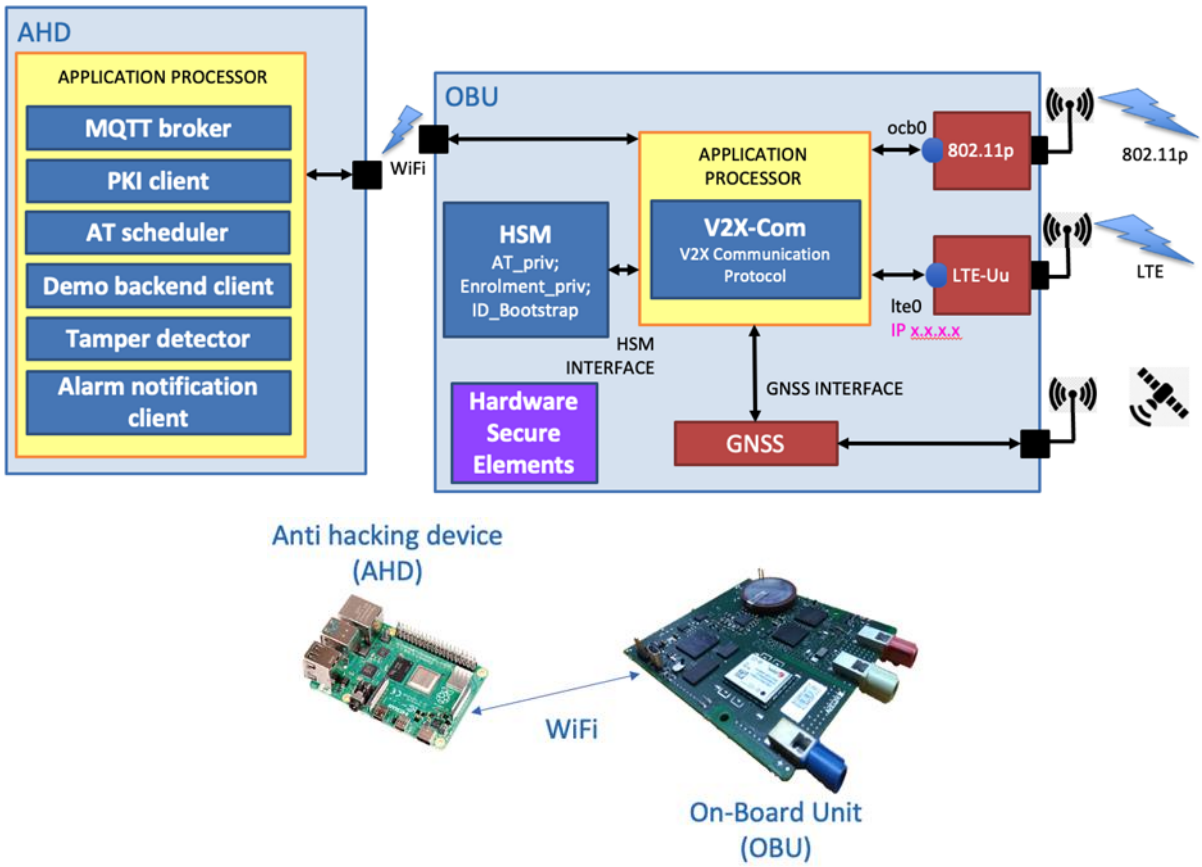


Figure 63: Architecture of connected car modules.

3.2.3.2 *Interoperability between different radio technologies*

The use case of "Interoperability between different radio technologies" supposes that cooperative cars already have ATs supplied by a correct registration into the PKI system and is demonstrated according to Figure 64 and Figure 65.

We have a system where all vehicles transmit CAM messages with a transmission frequency of 10 Hz, that need to be received by neighboring vehicles.

There are two different cases:

- **i)** when these CAM messages are transmitted from a vehicle which has 802.11p and LTE-Uu (Figure 64)
- **ii)** when the V2X messages are transmitted from a vehicle which only has LTE-Uu radio interface (Figure 65).

In case i), CAM messages are transmitted through the 802.11p interface, a real V2V communication interface, which enables neighboring vehicles under coverage to receive these messages. Nevertheless, vehicles "LTE-Uu only" or those that are far away do not receive them. The demonstration consists in doing that these vehicles also receive the messages using the forwarding mechanism through the MEC and the radio infrastructure.

The demonstration procedure workflow is:

1. One 802.11p vehicle transmits a compliant CAM which is received by the surrounding 802.11p vehicles and by the 802.11p RSU covering the area, for example RSU1 (note that in the testbed we have two RSUs: RSU1 and RSU2, covering two regions).

2. RSU1 forwards the message to its controller “V2XCom-11pRSU” module running in the MEC, which checks its validity.
3. If the message is non-compliant, it is dropped.
4. If the message is valid, it is sent to the “Forwarder” module, also in the MEC, using the MQTT broker.
5. The forwarder module, updates the LDM database of the MEC and checks the current forwarding policy.
6. The forwarding policy for our testbed is configured in such a way that messages received through an 802.11p interface, have to be forwarded to all “LTE-Uu only” vehicles and to all vehicles of the other region.
7. 7-Therefore, the “Forwarder” checks the LDM to consult the identifiers of other regions’ RSUs (in our testbed there is only RSU2) and, using the MQTT broker, forwards the CAM message to the “V2XCom-11pRSU” module controlling RSU2.
8. Also, the “Forwarder” checks the LDM to consult the “LTE-only” vehicles. Then, it forwards the CAM message and a list of all “LTE-only” vehicles that need to receive a copy of it, to the “V2XCom-CV2XRSU” module controlling V2X messages transmitted through the LTE network.
9. “V2XCom-11pRSU” module controlling RSU2 forwards the message to RSU2, which in turn broadcasts the message in Region 2 and it is received by 802.11p vehicles. One single message is sent from RSU2 because 802.11p is able to perform broadcast addressing.
10. “V2XCom-CV2XRSU” module controlling the LTE network transmits one copy of the message to each of the vehicles on the list. This transmission is done using the IP addresses of “LTE-Uu” only vehicles, which is stored in the LDM. We need to transmit one copy of the message per vehicle because the majority of LTE-Uu operators do not allow multicast transmissions.
11. The testbed user can check that all vehicles see the other ones using a computer connected to the backend LDM server (Figure 66).

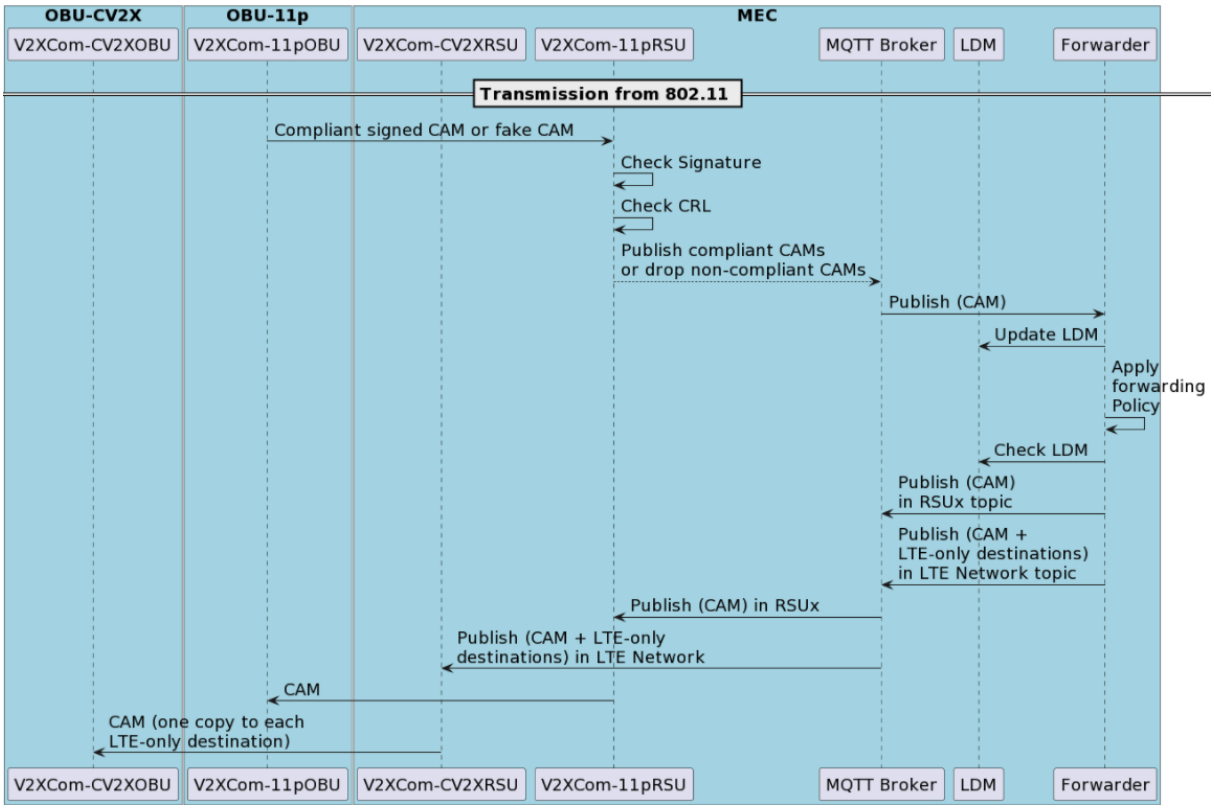


Figure 64: Workflow of use case “Interoperability between different radio technologies” when the transmission is initiated from a vehicle provided with 802.11p and LTE-Uu interfaces.

In case ii), CAM messages are transmitted through the LTE-Uu interface of an “LTE-Uu only” vehicle. These messages are not directly received by any other vehicle. All of them need to be forwarded through the MEC and the radio infrastructure. The demonstration consists in doing that all the vehicles in the testbed receive messages transmitted by LTE-Uu only vehicles.

The demonstration procedure workflow (Figure 65) is completely like the one of case i) with only these changes:

1. The transmission of the CAM message is done by “LTE-Uu only” vehicles.
2. The “Forwarder” module in the MEC forwards the message to all RSUs of the testbed.

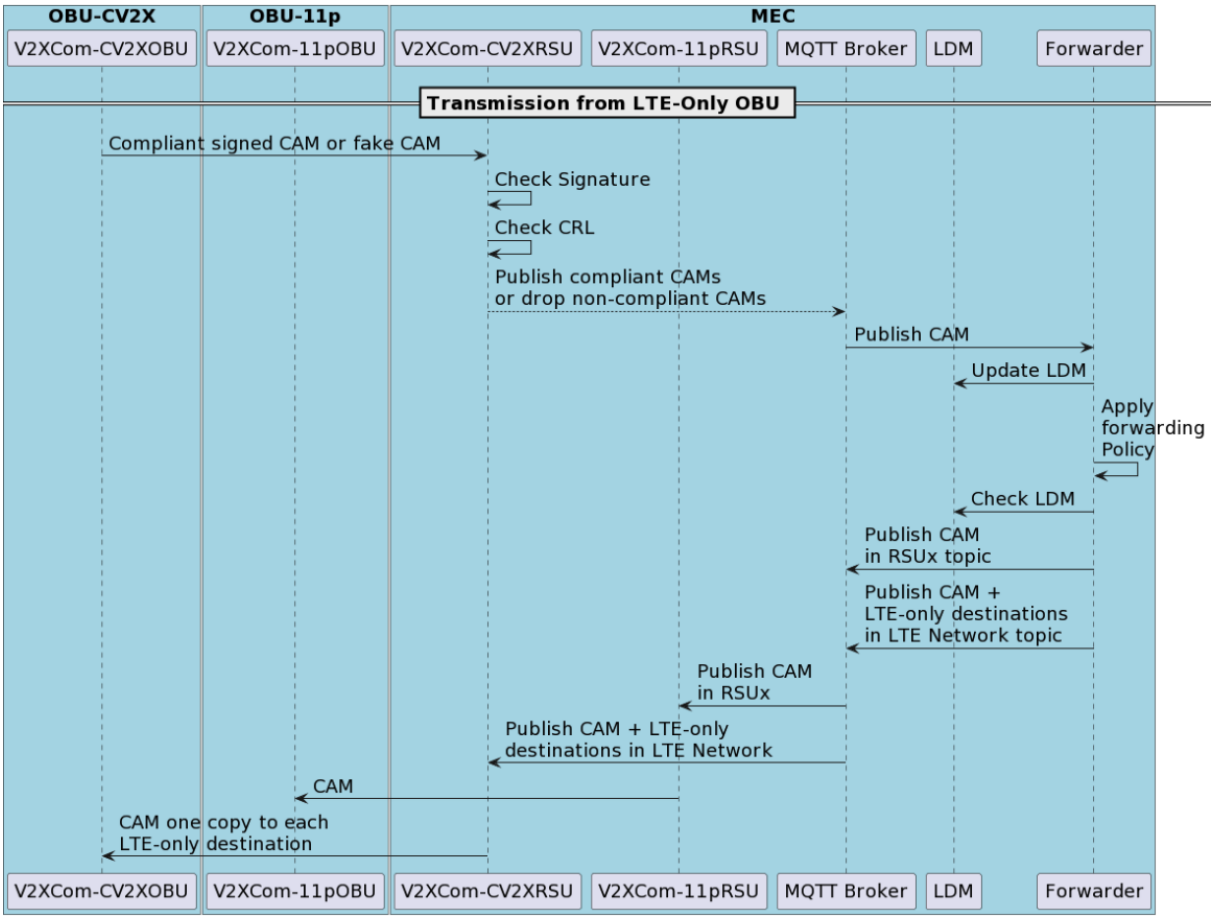


Figure 65: Workflow of use case “Interoperability between different radio technologies” when the transmission is initiated from a vehicle only provided with LTE-Uu interface.

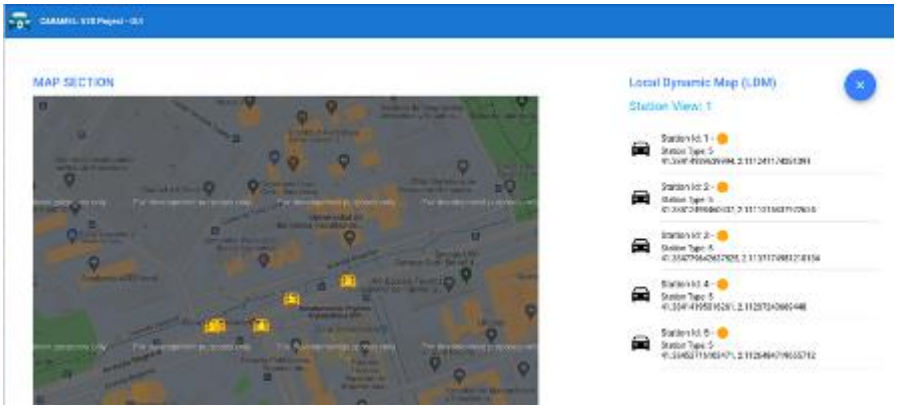


Figure 66: HMI to see the neighbors of one vehicle.

3.2.3.3 *Attack on the V2X Message Transmission*

The use case of “Attack on the V2X Message Transmission” supposes that cooperative cars already have ATs supplied by a correct registration into the PKI system. All vehicles transmit CAM messages with a transmission frequency of 10 Hz, that are received by neighboring vehicles and fixed infrastructure. On reception of these messages, the receiver vehicles and the MEC check the digital signature of the message. If the message is correct, it is sent to the applications running into the device. In our testbed, we have developed a user interface, through the backend LDM server, that allows the

testbed user to see the neighbours of one vehicle (Figure 66) and also, to trigger attacks using specific buttons (Figure 67). It is possible to generate the 4 different V2X message attacks described in section 3.2.1 (non-signed messages, messages signed with a non-valid certificate, replayed messages and non-authorized messages). The result can be seen using the same user interface.

Figure 67: Interface to trigger V2X message attacks.

Figure 68 and Figure 69 show the workflow of the attack response when the receiver of the non-compliant message is a vehicle. On the other hand, Figure 70 and Figure 71 describe the workflow of the attack response when the receiver of the non-compliant message is the MEC.

The demonstration procedures workflows depend on the type of attack, and they are described below:

a) Non-signed messages:

1. The attacking vehicle transmits a non-signed message.
2. The receiver device, either an OBU or the MEC, simply discards the message.

b) Messages signed with a non-valid certificate:

1. The attacking vehicle transmits a message signed with a certificate not generated by the Authorization Authority of the Caramel PKI system.
2. The receiver device, either an OBU or the MEC, performs two basic anti-reply operations which are checking if the message has already been received or if it is too delayed. As in this case the message is not replayed, the procedure jumps to the next step.
3. The receiver device checks if the certificate is present in the Certificate Revocation List (CRL). As this certificate is false, we assume that it is not present in the list.
4. The receiver checks the authenticity of the digital signature and the validity of the attached certificate. In the OBU, the attached certificate validation requires the assistance of the HSM where the Root Authority certificate, with which it is signed, is stored. In the MEC, as there is no HSM, this operation is uniquely done by the V2XCom-MEC module.
5. In this use case, the previous operation returns that the certificate is not valid and an alarm is triggered for statistical purposes to the backend.
6. The message is dropped.

c) Replayed messages:

1. The attacking vehicle captures a valid transmitted message and replays it.
2. The receiver device, either an OBU or the MEC, performs two basic anti-reply operations which are checking if the message has already been received or if it is too delayed. In this case, as the message is replayed, the module “V2XCom-*” detects it.
3. The message is dropped.

   Note: No alarm is triggered because the origin of the replayed message can be a retransmission from an RSU with overlapping coverage, or the message can be delayed due to network saturation conditions.

d) Non-authorized messages:

1. The attacking vehicle transmits a CAM message, stating that the vehicle is a “Emergency vehicle”, with a valid AT issued for a “Passenger car”.

2. The receiver device, either an OBU or the MEC, performs two basic anti-reply operations which are checking if the message has already been received or if it is too delayed. As in this case the message is not replayed, the procedure jumps to the next step.
3. The receiver device, either an OBU or the MEC, checks if the certificate is present in the Certificate Revocation List. As this certificate is not present in the CRL, the procedure jumps to the next step.
4. The receiver checks the authenticity of the digital signature and the validity of the attached certificate. In the OBU, the attached certificate validation requires the assistance of the HSM where the Root Authority certificate, with which it is signed, is stored. In the MEC, as there is no HSM, this operation is uniquely done by the V2XCom-MEC module.
5. In this use case, the previous operation returns that the certificate is valid, but the type of vehicle does not match.
6. An alarm is triggered for statistical purposes to the backend.
7. The message is dropped.

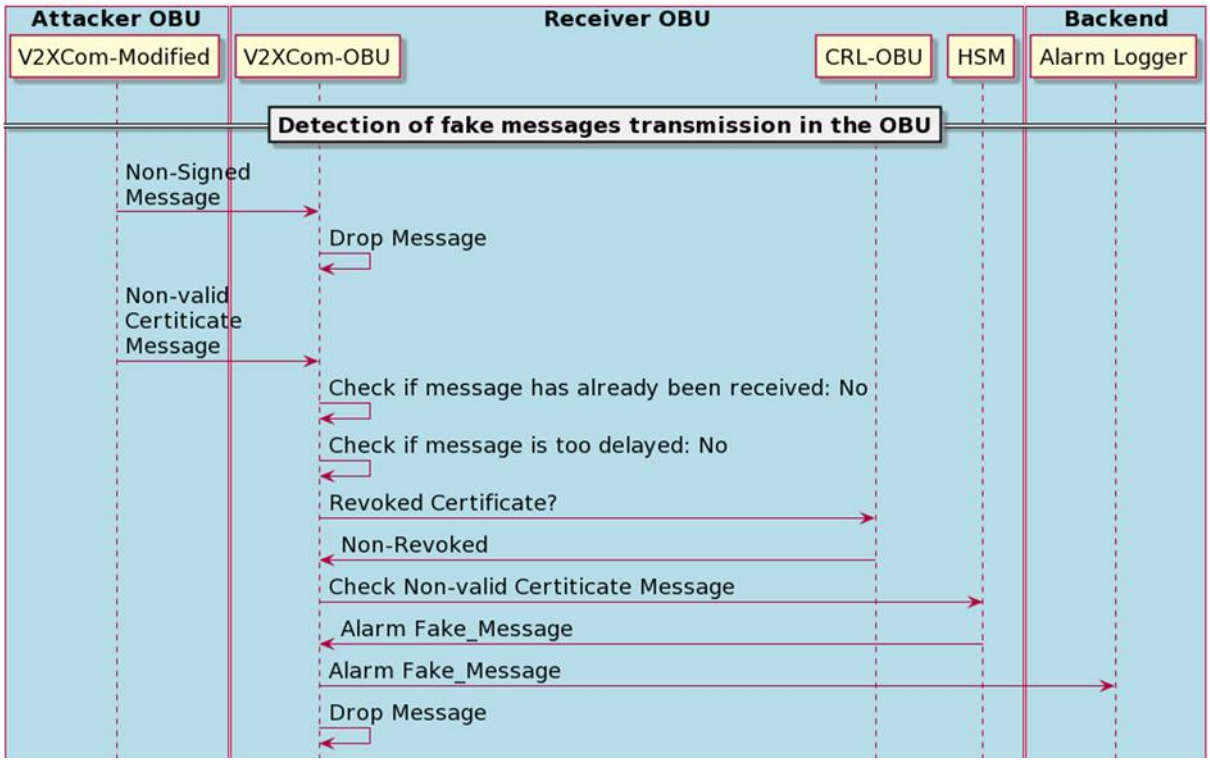


Figure 68: Workflow of use case “Attack on the V2X Message Transmission” - part 1 - when the detection procedure is performed in the OBU.

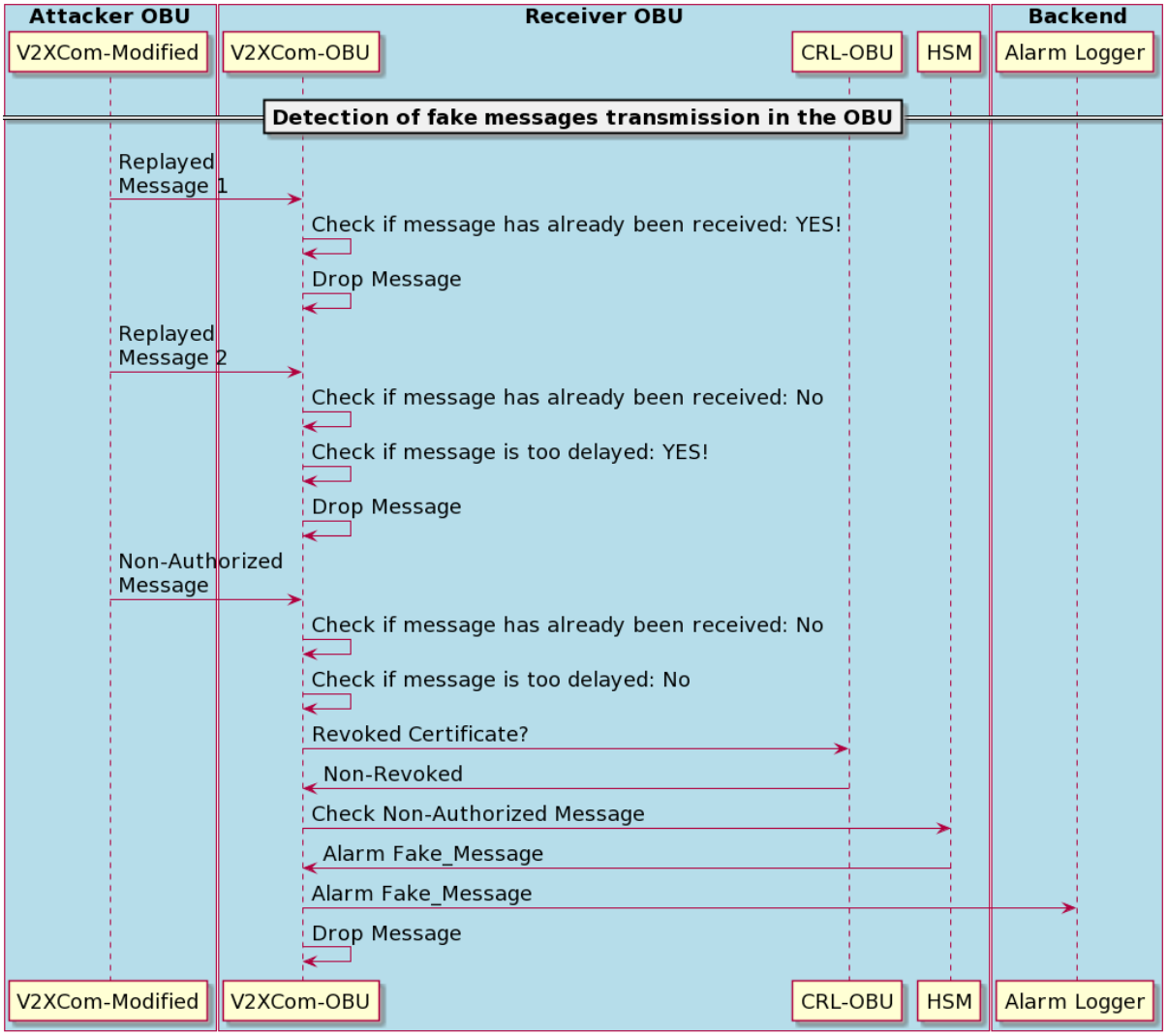


Figure 69: Workflow of use case “Attack on the V2X Message Transmission” - part 2 - when the detection procedure is performed in the OBU.

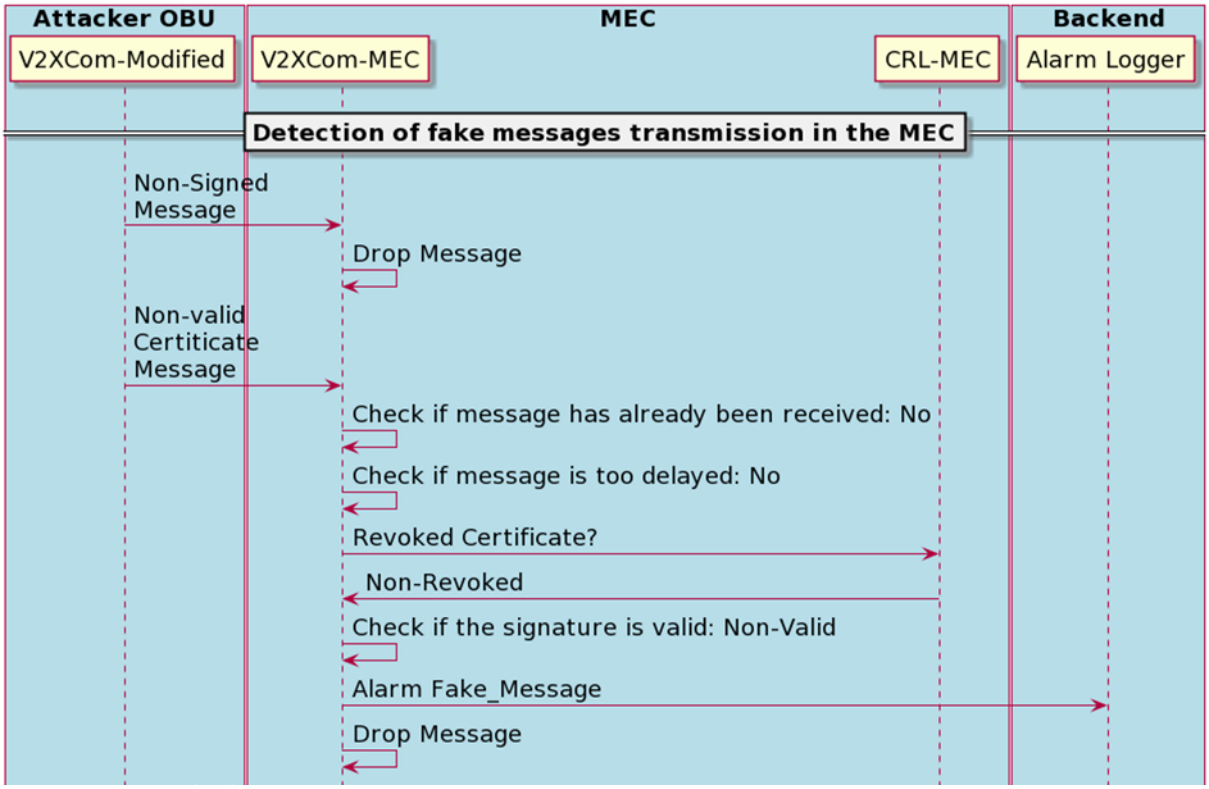


Figure 70: Workflow of use case “Attack on the V2X Message Transmission” - part 1 - when the detection procedure is performed in the MEC.

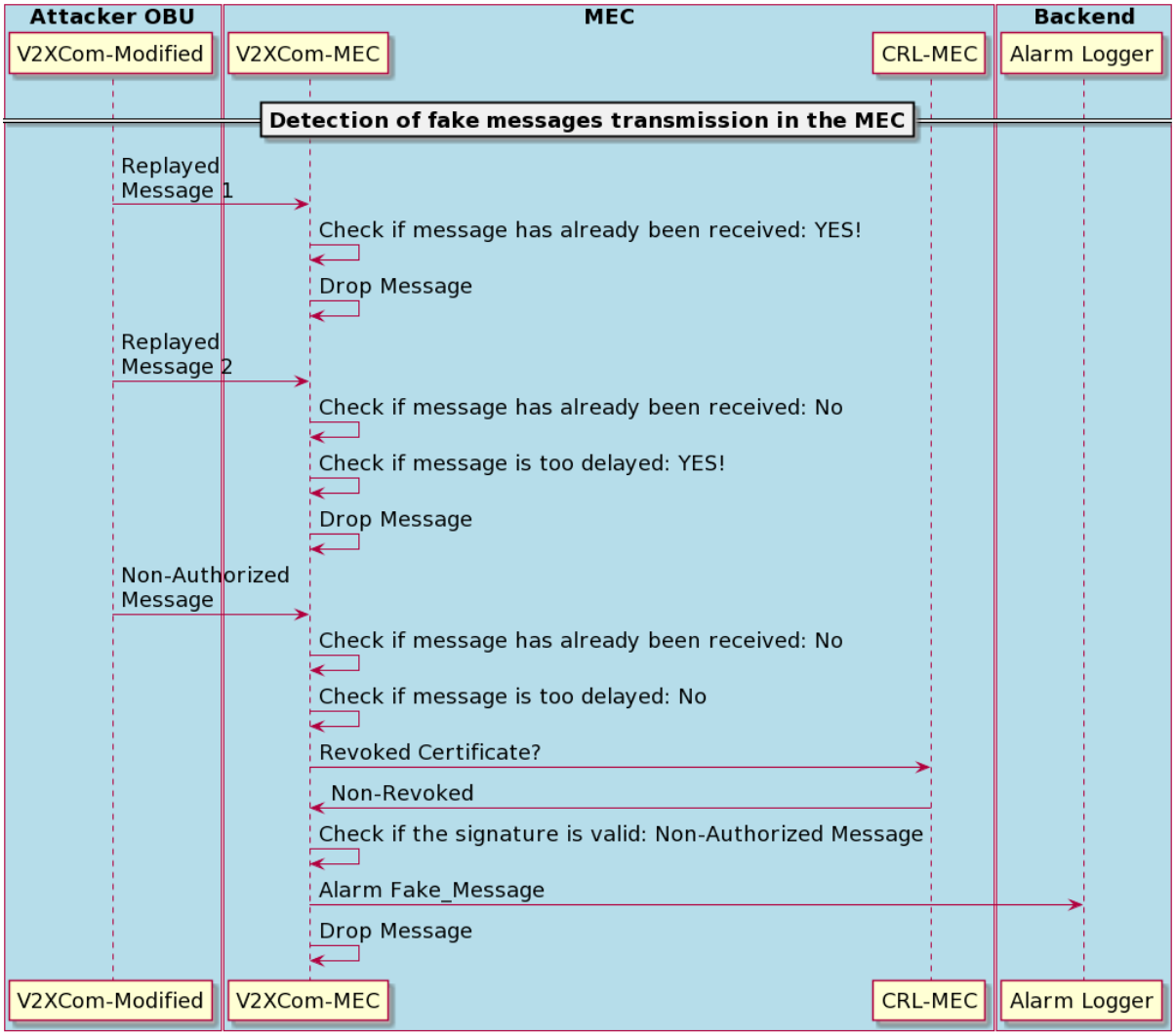


Figure 71: Workflow of use case “Attack on the V2X Message Transmission” -part 2- when the detection procedure is performed in the MEC.

3.2.3.4 *Tracking of vehicles by sniffing sent messages*

V2X messages are signed by an Authorization Ticket (AT) that uniquely identifies the sender vehicle. This creates a privacy issue, as a malicious sniffer could identify all the V2X messages sent with a specific AT and track the trajectory of that vehicle. One possible solution to avoid this tracking is to change the AT periodically. To minimize the probability of tracking, CARAMEL has developed a scheduler to choose which is the optimum moment to change the AT. This optimum moment depends on the V2X messages sent by the surrounding vehicles, as explained in the deliverable D4.1, section 4.4.1 “Attack Mitigation with the AT Scheduler”. To apply this scheduler, it is necessary to have multiple moving vehicles in the road sending V2X messages. As this is not possible in the testbed, this algorithm will be demonstrated using a simulation tool.

The tool simulates traces of multiple vehicles circulating in the city of Cologne, taken from the dataset already explained in D5.2. The AT Scheduler is installed in one of the simulated vehicles (target vehicle) and it receives the V2X messages of the surrounding vehicles (candidate vehicles). The test focuses on a region of Cologne that has a roundabout with multiple levels and intersections, as it poses a more challenging and interesting scenario compared to a straight road.

Alongside the simulation, i2CAT has deployed a visualization tool that allows to see from the data stored during the simulation by the AT Scheduler logs the positions of the cars in the simulation (taken from the simulated V2X messages), the position of the target car, the AT score, and when the AT scheduler decides to change the AT based on the previous variables. Figure 72 shows an example of this

visualization. In the upper plot, we can see the position of the cars inside the simulated scenario. The green dot represents the position of the target car, which has installed the AT Scheduler, and the blue dots represent the other cars in the simulation. All the cars are sending V2X messages every 100 ms approximately. In the lower plot, we can see a blue line that relates the tracking score (Y axis) over the time of the simulation (X axis) and the green vertical lines represent the timestamp where the AT Scheduler system has decided to request an AT change. In this example, the tracking score is always 0.

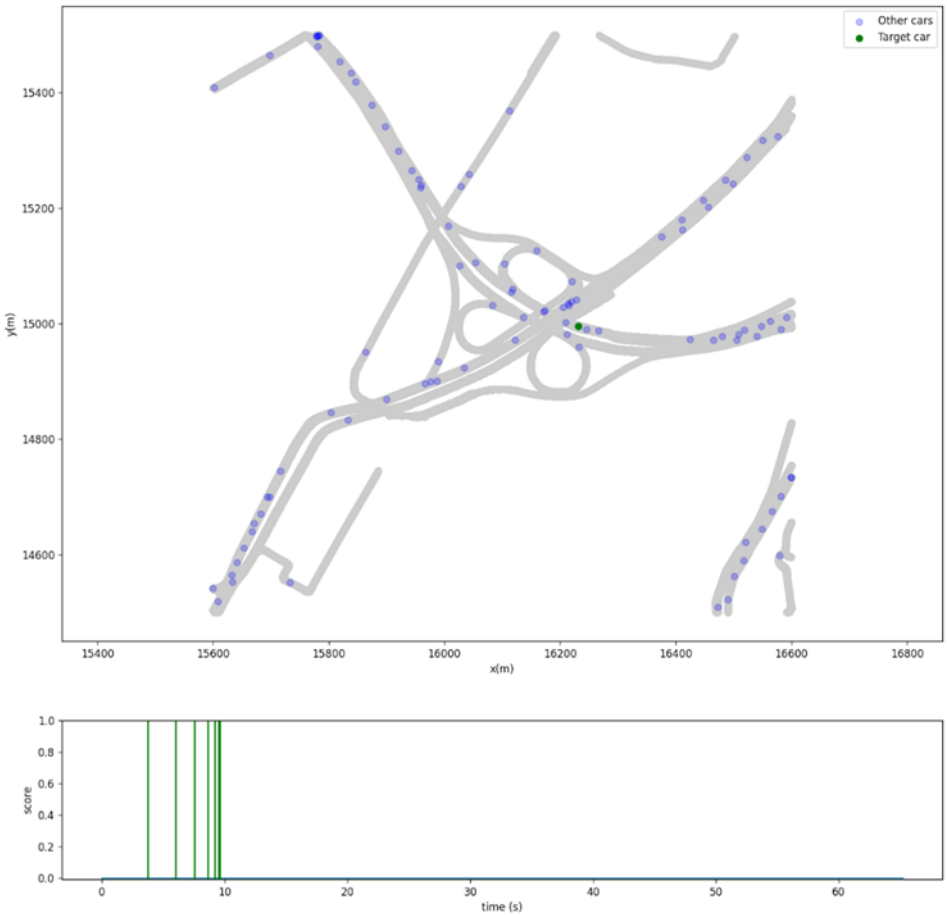


Figure 72: Visualization of the tracking avoidance tool showing a) the position of the target and surrounding cars (plot on the top) and b) the tracking score and the times when an AT change has been requested (plot on the bottom).

3.2.4 Initial results in lab

The testbed has been deployed in the lab (Figure 73). Its components are:

- 1 NUC computer acting as a MEC.
- 1 NUC computer acting as a dRAX.
- 2 RSUs.

- 1 small cell.
- 4 OBUs with 4 anti-hacking devices, plus 1 OBU for backup. They need to be feed by power supplies (in lab) or batteries (outside lab).
- 1 LTE router to have access to Internet.
- 1 Ethernet switch and cabling.
- 3 servers hosted in Internet: PKI, monitoring backend and LDM backend.

Figure 73: General picture of the testbed with GPS antennas going out through the window.

Integration and problems that have appeared:

- The modules hosted in MEC (V2XCom, vEPC, forwarder, PKI client, Certification revocation software and LDM) have been successfully integrated in virtual containers and connected through virtual networks inside Kubernetes. The configuration is stable.
- The RSUs are correctly connected to their V2XCom modules hosted MEC using Ethernet VLANs. They correctly forward messages between 802.11p and Ethernet interfaces. The configuration is stable.
- The LTE network is formed by 4 different components: i) the UE which are the LTE modems integrated in the OBUs, the small cells, the dRAX and the vEPC. Its configuration works but it presents instability problems that we have been unable to solve.
    - The project acquired 2 small cells, but in the last month, one of them began to misbehave, it disconnected User Equipments (UE) that were already associated. For this reason, we decided to deploy a testbed with a single small cell. The demonstration is not affected as it is possibly to demonstrate all CARAMEL use cases.
    - When several UEs try to get associated to the network at the same time, the LTE network does not associate them correctly. They do not have access to Internet. We have found that we have to activate the UEs sequentially, one by one. Nevertheless, after some uncertain time (several hours), the network disconnects them and the UEs need to be restarted. The software to control the network is, for one hand the Accelleran's dRAX v2.1 (Cloud-Native Open RAN software) and, for the other hand, the open source vEPC (virtual Evolved Packet Core) Open5GS v2.1.7. Our impression is that some of these two components has some bug, probably the vEPC.
- The software module V2XCom is properly integrated in the OBU hardware and connects to its Hardware Security Module (HSM). This has been one of the main problems of the integration since V2XCom relays in Vanetza open-source framework, which is very wide and not optimized for small capabilities devices like the OBU. Nevertheless, the integration has succeeded and works correctly.

- The integration of the software modules inside the anti-hacking device has been done correctly: The modules MQTT broker, PKI client, AT scheduler, Demo backend client, Tamper detector and Alarm notification client have been correctly integrated on the anti-hacking device platform specified in deliverable D5.5. All applications run correctly except for the AT scheduler that requires a larger computation capability than the provided by the Coral Dev platform. This aspect was already discussed in D5.3 "Machine Learning based Detection of Attacks into Anti-Hacking Device".
- The Wi-Fi connection between the OBU and the Anti-hacking device is done properly, uses the security functions of the standard 802.11, but messages are not signed by HSMs. In the first design of the CARAMEL architecture, it was planned that messages transmitted between the OBU and the anti-hacking device, a part of the standard encryption system of the WiFi, they would be additionally signed by the HSM attached to each device. The problem is that the HSM of the anti-hacking device has not been designed to sign hundreds of messages per second, as it is the HSM of the OBU. Therefore, although this signature system has been programmed and tested, it is not used in the testbed. We propose that future commercial versions of this devices are connected by cable (Ethernet), instead of by WiFi.
- The forwarder is correctly integrated with the 802.11p and the LTE networks. It can parse and generate V2X messages and forward them to OBUs connected through 802.11p or LTE networks. It also updates the MEC's LDM and the forwarding rules can be modified.
- The alarms transmitted by the OBU are correctly integrated in the monitoring backend. This sever receives and understands all alarms generated by OBUs.
- The information transmitted from the OBUs at the LDM backend server is correctly integrated. This server, build for demonstration purposes, enables to show the lists of vehicle neighbors and trigger V2X messages attacks.
- The V2XComm module is correctly integrated with the PKI client. Both are able to manage/interchange/request digital certificates and Authorization Tickets (AT) and inform if an AT is revoked.
- The AT scheduler is correctly integrated with the V2XCom module. The scheduler receives CAM messages from the V2XCom module and decides when is best to change the AT. Then, it notifies the V2XCom to change AT.
- The open-source framework Vanetza has arisen multiple problems, basically related with the management of the security functions:
    - Implementation of the V1.3.1 version of security: The base project of Vanetza was using the obsolete and currently deprecated version of the ETSI v1.2.1 to implement the security on the sent messages. With the CARAMEL project, there has been implemented the v1.3.1 in retro compatibility with the previous ones. This new version, which also regards of Release 2 of the ETSI C-ITS standard, uses the same data structures as the IEEE 1609.2 protocol. This means that opens a window of compatibility with the WAVE stack.
    - Aligned the ASN.1 files version: The ASN.1 files describing how the data structures must be when serializing, usually get problems with the alignment of the versions of all of the ASN.1 files. Usually, all the standards are not published altogether, and getting all the asn.1 files with the perfect alignment can pose a huge challenge.
    - Fix serialization problems regarding the asn1c generated files: The asn1c utility generates plain C code regarding the ASN.1 described data structures it has been run with. The problem with the usage of this library with complex structures, like the ones described with the ETSI C-ITS or IEEE 1609.2, is that the memory allocation with the usage of multiple levels of pointers pointing to other pointers, makes any code developed with such a solution prone to segmentation faults and, obviously, huge memory leaks.
    - HSM library endianness: The processors used within the HSM use little-endian codification when transmitting numbers higher than one byte. This has posed a problem for the project because external modules assumed a big endianness on the codification. Along with this, an extra layer of base64 encoding had to be added to ensure the byte arrays are properly sent to the python modules.

Implications of the known problems:

- Problem of having an LTE network unstable: Vehicles need an LTE connection to get access to the servers hosted in the cloud, and also to use this connection to transmit V2X messages in case of not having a native V2X radio interface (802.11p, LTE-PC5, 802.11bd or NR-V2X). We could have used a commercial LTE operator for the first purpose of accessing to the Internet servers, but in order for the MEC be able to push V2X messages to the OBUs over LTE, it is needed to know their IP addresses, which are not provided by commercial operators. So, the reason of having deployed our own LTE operators if having knowledge of the IP addresses of the OBUs in the system. Commercial operators use Network Address Translators (NAT) at the border of their network which masquerade the IP addresses of the OBUs. In case of willing to use a commercial LTE operator for V2X communication, it will be necessary to redefine the network architecture, which can introduce additional delays.
- The capacity of the OBU and the Anti-Hacking device: The applications that CARAMEL plans to execute in both, the OBU and anti-hacking device, require more computational capability and memory. In the case of the OBU, the V2XCom module, based in Vanetza software, is not designed to be embedded in automotive computers. The solution is to increase the capacity of the OBU or optimize the V2XCom code. In the anti-hacking device, the main problem is that all basic required applications are already using all its computational capacity. When the AT scheduler starts, all the system is slowed down. With this architecture, the change of AT has to be scheduled without an intelligent algorithm.

3.2.5 Deployment in Panasonic premises in Langen

On the days 8th and 9th of June, pillar 2 testbed on V2X communication was deployed in the parking lot of Panasonic premises in Langen, Germany (Figure 74).

Figure 74: General view of the deployment of pillar 2 in Langen (Germany).

The deployment behaved in the same way than the test in the lab:

1. The MEC was deployed correctly with its virtual containers running (Figure 75 and Figure 76).

Figure 75: MEC, dRAX and small cell deployed in Langen.

```
ubuntu@caramel-edge-server:~$ kubectl get deploy,statefulset -n test-caramel
NAME                                        READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mqtt-broker                 1/1     1            1           5d22h
deployment.apps/v2x-certificate-revocator   1/1     1            1           5d22h
deployment.apps/v2x-forwarder               1/1     1            1           5d22h
deployment.apps/v2xcom1rsu                  1/1     1            1           5d22h
deployment.apps/v2xcom2rsu                  1/1     1            1           5d22h
deployment.apps/v2xcom3cv2x                 1/1     1            1           5d22h

NAME                    READY   AGE
statefulset.apps/ldm    1/1     5d22h
```

Figure 76: Kubernetes pods running the V2X Stack.

2. The RSUs were deployed and connected to the MEC through the switch (Figure 77)

Figure 77: RSUs.

3. The LTE network components were deployed: dRAX and small cell can be seen in Figure 75, the vEPC OpenG5S running in the MEC (Figure 78). Also, Figure 79 shows the radiofrequency parameters of the small cell and Figure 80 shows the 3 UEs of the OBUs associated to the network.

```
ubuntu@caramel-edge-server:~$ kubectl get deploy,statefulset -n open5gs
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/open5gs-nrf         1/1     1            1           22h
deployment.apps/open5gs-webui       1/1     1            1           22h

NAME                                 READY   AGE
statefulset.apps/mongo               1/1     43d
statefulset.apps/open5gs-hss         1/1     22h
statefulset.apps/open5gs-mme         1/1     22h
statefulset.apps/open5gs-pcrf        1/1     22h
statefulset.apps/open5gs-sgw-c       1/1     22h
statefulset.apps/open5gs-sgw-u       1/1     22h
statefulset.apps/open5gs-smf         1/1     22h
statefulset.apps/open5gs-upf         1/1     22h
```

Figure 78: Open5GS pods running in Kubernetes.

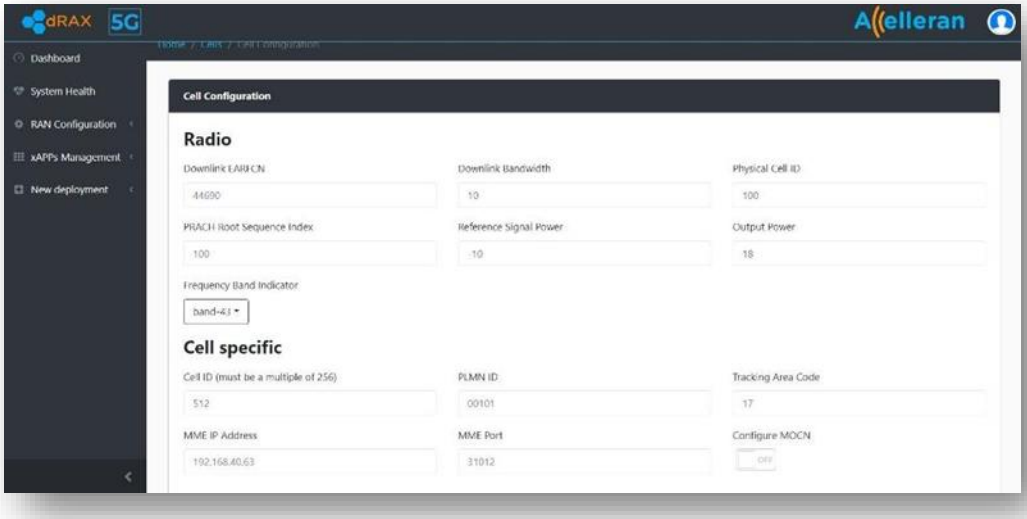


Figure 79: Small cell radio configuration.

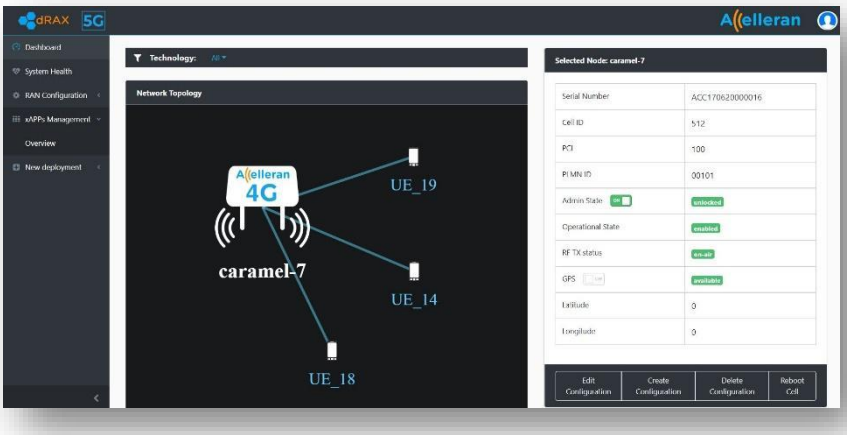


Figure 80: 3 UEs connected to the Small Cell. View from dRAX Dashboard.

4. Four different OBUs with their respective anti-hacking devices plus power sources were configured and deployed (Figure 81). In order to be able to configure the OBU and the anti-hacking device, previously to its operative phase, we connect the anti-hacking device with an Ethernet cable to the switch. In this way, we can enter into these two devices with a *ssh* and perform all necessary setup. After this phase, the Ethernet cable is unplugged and the OBU can work standalone.

Figure 81: OBU with anti-hacking device and batteries.

After performing several tests, we reached to the same conclusions as in the lab:

- The RSUs network is stable.
- The LTE network presents problems of instability. UEs get disconnected after a random time.
- The integration between the V2XCom and the OBU's hardware and HSM works correctly.
- The integration between the V2XCom and PKI servers works correctly, and messages are sent correctly signed.
- The interoperability between 802.11p and LTE technologies works correctly.
- The integration between the V2XCom and the monitoring and LDM servers works correctly.

3.3 *Tamper attack on Vehicle's OBU: attack use case description and assessment*

As introduced within the Deliverable D2.1, the attacks on-board systems could also be physical if the attacker gets physical access to the On-Board-Unit (OBU) by accessing the car. As a point to have in mind, the attacker could even have acquired another OBU (e.g., aftermarket sample) with the aim to study potential vulnerabilities beforehand.

Since the OBU is the gateway to the vehicle's network communications, its protection should be mandatory and the top priority, so, as to prevent it from becoming the weakest link in the vehicle's security chain.

Hardware securitization has been carried out by identifying the potential threats on the OBU and then designing a protection strategy to avoid attacks on the identified vulnerabilities. The design of the V2X unit from the OBU is done having in mind HW and SW point of view, and is self-protected against different attacks shown in the Table 15 below:

Title	Description	Solution developed
HW tamper attack	The attacker can get physical access to an OBU by accessing the car. The attacker could also have acquired another OBU (e.g., aftermarket sample) in order to study potential vulnerabilities beforehand.	i) Open box detection switch. ii) Wire-mesh protection of secure signals. iii) Zeroization of cryptographic keys.
SW tamper attack	The attacker gets access to the internal ports of the OBU to download malicious software	Implement trust of chain (secure boot)
HW manipulation	The attacker gets access to the internal OBU and replaces HSM and/or memory to use malicious software downloaded into them.	Mutual authentication Chain of trust Secure boot

Table 15: Summary of possible attacks with direct access at the OBU.

When some of the previous attacks are performed, the OBU unit is disabled for its normal use and there will be impossible to send any V2X signed message. Then the system detects the attack and informs the MEC that the OBU has been compromised.

3.3.1 Use case description (UC2.3)

In this section in advance, we will focus on the two use cases defined for the final project demonstration at the Panasonic parking lot in Germany, which are focused on the HW as the entrance door to the system. This means that the attacker is able to get physical access to the OBU and exploit vulnerabilities.

The use case for the tamper attack on vehicle's OBU implies that the attacker tries to get access to the OBU electronics in order to manipulate it. D3.6 chapter 3.6.3 explain the different countermeasures and anti-tamper techniques developed to detect tamper attacks.

In order to detect different hardware attacks, we have developed an active protection covering different sensible areas:

- Open enclosure: detected by a switch
- System clock oscillators: protected by epoxy resin
- V2X signals: protected by active wire-mesh
- GNSS signals: protected by an enclosure with active wire-mesh

Although attacks to the system clock, V2X signals and GNSS signals are destructive tests, the use case for tamper attack will be focused in the open enclosure.

Figure 82, extracted from D2.4 “System Specifications and Architecture”, shows the involved processes and actors.

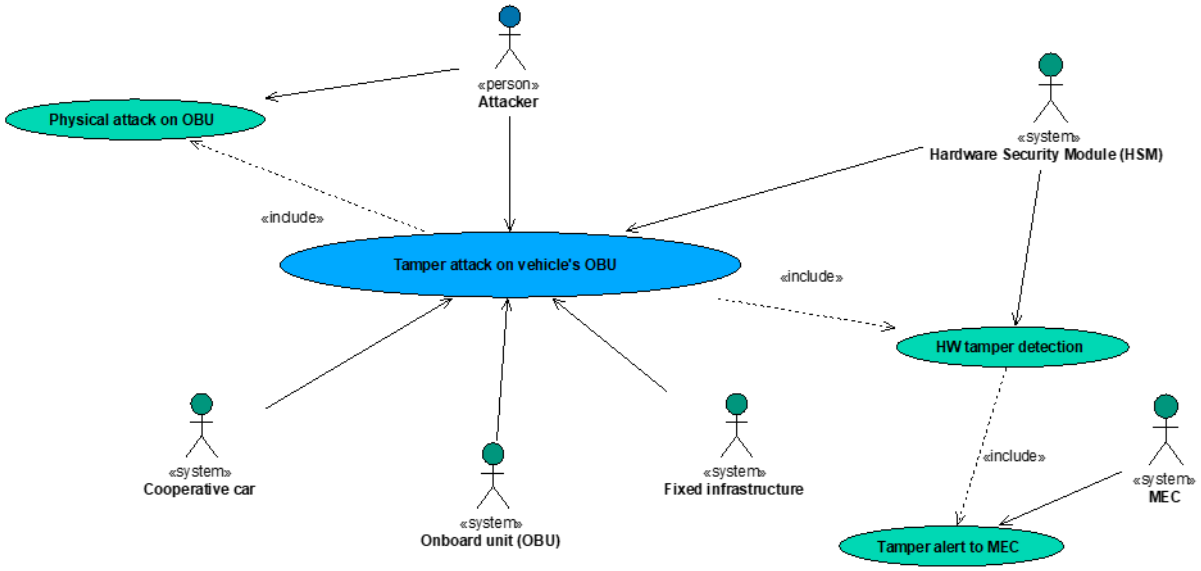


Figure 82: Workflow Use case for the tamper attack on vehicle’s OBU.

Involved Actors:

- **Attacker:** The attacker is a person who opens physically the OBU enclosure.
- **Cooperative car:** In the demonstration scenario, the cooperative car is represented by an OBU plus its anti-hacking device. It is connected to a laptop by WiFi, from where it is possible to visualize the messages received from other cooperative cars.
- **Onboard unit (OBU):** Is the communication unit inside the car, and it has the hardware and software needed for V2X, LTE and WiFi secure communication, as well as secure message signatures and secure keys management.
- **Fixed infrastructure:** The PKI infrastructure has the servers in ATOS network which are reachable using a standard Internet connection. It is composed of multiple virtual containers executing the following processes: the forwarding of the certificates identifying the Root, Enrolment, Authorization and Revocation Authorities, the enrolment process, the authorization process and the revocation process. At the end of the whole process, the PKI client, executed in each cooperative car, obtains a group of ATs which are used to sign V2X messages.
- **MEC:** It is a computer with virtual containers executing the following processes: one V2X communication protocol stack for each RSU and for the LTE network, the virtual Evolved Packet Core (vEPC) of the LTE network, the V2X forwarder, the Local Dynamic Map (LDM), the MQTT broker, the application that receives alarm notifications from cooperative cars and from other MEC processes, the application which decides if a cooperative car will have its certificates revoked (for demonstration purposes, it is an user interface where the user can choose between two options YES/NO) and the application responsible of the Certification Revocation List (CRL) distribution.
- **Hardware Security Module (HSM**): The HSM is the hardware element brought into the OBU microcontroller and is the responsible of storing private keys and sign messages.

3.3.2 Use case test setup

The testbed for tamper attack test setup is the same used for the attack on the V2X message transmission (see section 3.2.2): it will be deployed in Panasonic facilities and will be composed of three different sites plus several servers reachable through Internet and laptops to activate and visualize the demonstrations. This is because the tamper attack will be detected and the attack notification to the

MEC works in a similar way that the V2X message transmission attack and needs the same infrastructure.

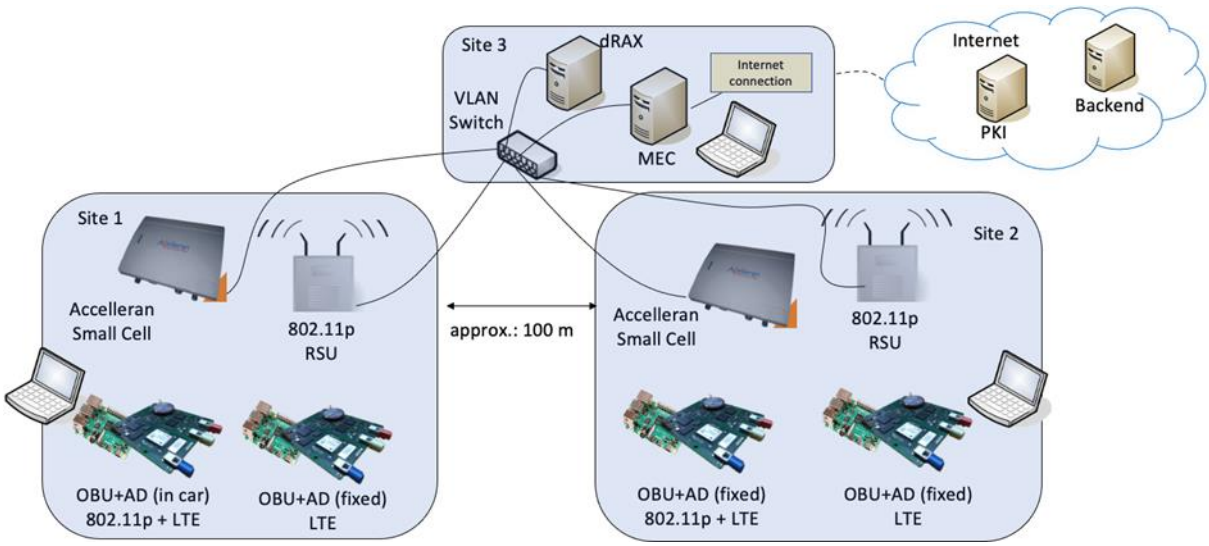


Figure 83: Testbed Layout.

The tamper attack on the vehicle's OBU use case will be demonstrated by opening the OBU enclosure, based on the use case description on D2.4 (UC2.3).

Figure 84: CARAMEL OBU with enclosure showing antenna connectors.

3.3.2.1 *HW tamper attack workflow and evaluation*

The procedure workflow when a tamper attack is detected in the different protected areas is the same. In the demonstration, we will detect the attack when the enclosure is opened:

1. In normal operation, the anti-hacking device pings the OBU in order to check there's no alarm.
2. An attacker opens the OBU's enclosure.
3. The detection switch detects the enclosure opening and triggers an alarm.
4. The processor detects the alarm and internal keys are deleted immediately.
5. Access to the HSM to sign messages is not allowed because internal keys are deleted.
6. Anti-hacking device is not able to communicate with the OBU, assuming the OBU is compromised, and sends an alarm notification to the MEC.
7. The MEC notifies the attack to the PKI server.
8. PKI decides if the vehicle's certificate must be revoked.

9. The MEC also sends an alarm notification to the backend for statistical purposes.

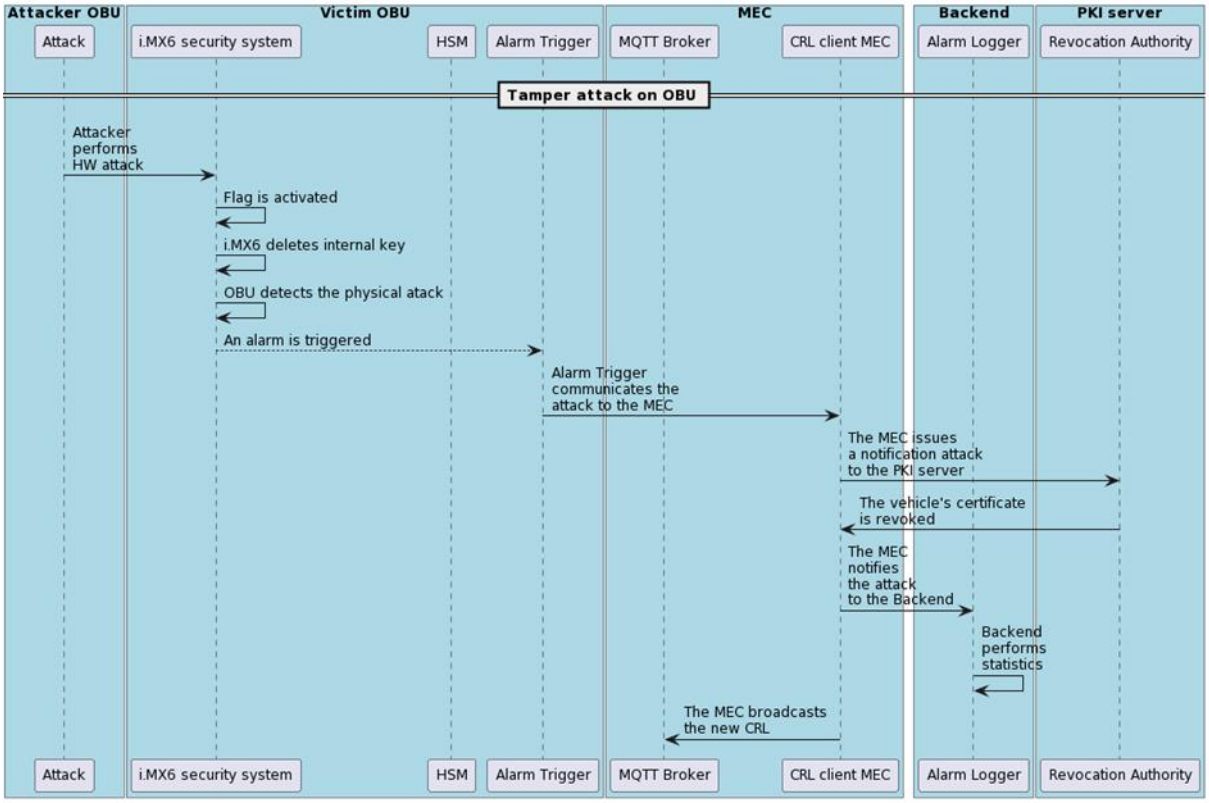


Figure 85: Workflow of the Use Case “Tamper attack on OBU”.

3.3.3 Test Results

With the test setup described, key files needed to communicate with the HSM are located in **/v2xconf** folder. These files are:

- keyset0: used for normal operation with the HSM
- keyset1: used for V2X private keys injection

When we open the box enclosure, we can see the files containing the communication keys are permanently deleted (Figure 86), so communication with the HSM becomes impossible and the OBU is not able to sign any message with the secure keys.

```
root@caramel-OBU2:/v2xconf/conf/nxtm#
root@caramel-OBU2:/v2xconf/conf/nxtm# ls -al
drwxr-xr-x    2 root     root             0 Feb  2 13:15 .
drwxr-xr-x    5 root     root             0 Jan  1  1970 ..
-rw-r--r--    1 root     root          6686 Feb  2 13:15 caramel-tamper-detection.log
-rwxr-xr-x    1 root     root          1086 Feb  2 13:15 keydata.txt
-rwxr-xr-x    1 root     root           933 Feb  2 13:15 keydata_factory_reset_commands_only.txt
-rwxr-xr-x    1 root     root            99 Feb  2 13:15 keyset0.txt
-rwxr-xr-x    1 root     root            99 Feb  2 13:15 keyset1.txt
-rwxr-xr-x    1 root     root           190 Feb  2 13:15 v2xscppalutil-ks1.bin
-rwxr-xr-x    1 root     root           190 Feb  2 13:15 v2xscppalutil.bin
root@caramel-OBU2:/v2xconf/conf/nxtm# ls -al
drwxr-xr-x    2 root     root             0 Feb  2 14:42 .
drwxr-xr-x    5 root     root             0 Jan  1  1970 ..
-rw-r--r--    1 root     root          6812 Feb  2 14:42 caramel-tamper-detection.log
root@caramel-OBU2:/v2xconf/conf/nxtm# ls -al
drwxr-xr-x    2 root     root             0 Feb  2 14:42 .
drwxr-xr-x    5 root     root             0 Jan  1  1970 ..
-rw-r--r--    1 root     root          6812 Feb  2 14:42 caramel-tamper-detection.log
root@caramel-OBU2:/v2xconf/conf/nxtm#
```

Figure 86: OBU’s Linux command console showing keys before and after activate tamper switch.

Once the keys are deleted, a tamper alarm notification is seen also in the frontend/dashboard, as it can be observed in Figure 87.

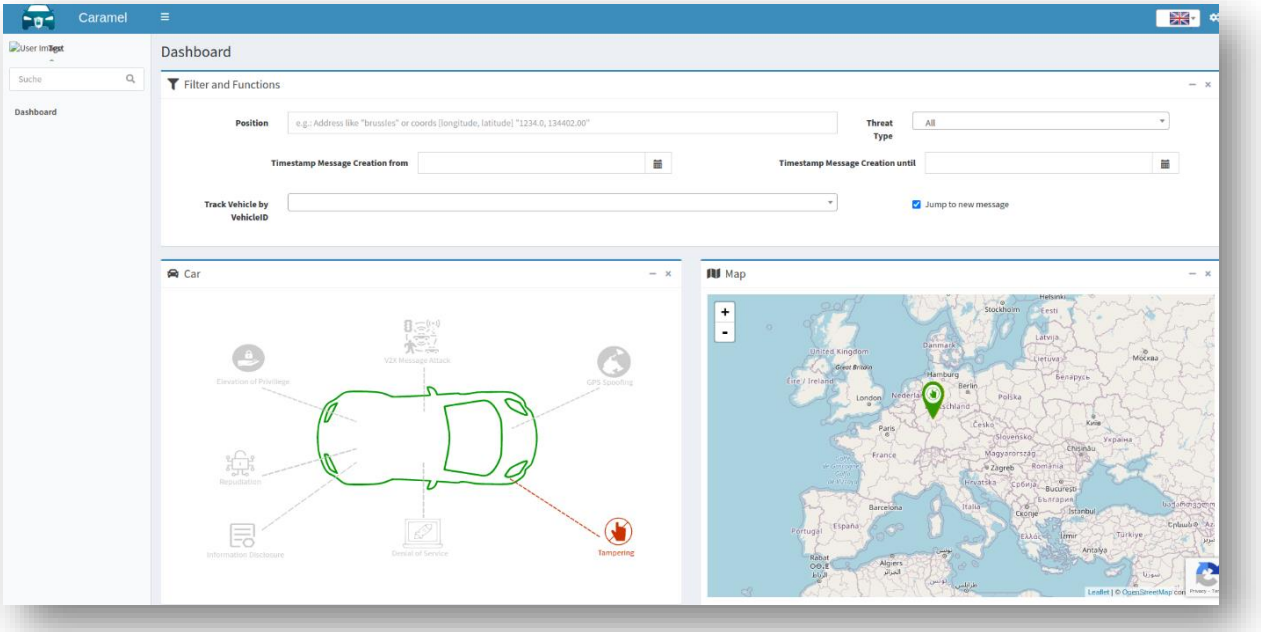


Figure 87. Frontend/dashboard's notification showing the tamper attack .

3.4 *Certificate Revocation: attack use case description and assessment*

3.4.1 Description

In this use case we aim at demonstrating the mechanism that revokes certificates from malicious actors once they are detected by the MEC. The reasons behind the decision revocation request are out of the scope of CARAMEL but, for demonstration purposes, we assume that attacks on V2X messages do not revoke certificates, and a GPS spoofing attack would revoke the certificates.

As the GPS spoofing attack is demonstrated using a simulator, also for demonstration purposes, we have prepared a user interface that can trigger the alarm in the OBU as if the vehicle was under GPS spoofing attack (Figure 88).

Pushing the button “GPS spoofing attack”, produces an alarm in the OBU, that informs the “Certificate Revocation Decision” module in the MEC, which decides to revoke the certificates of the vehicle producing the alarm.

Figure 88: User interface with a button to simulate a GPS spoofing attack in the OBU, which produces an alarm that revokes the certificates of the vehicle under attack.

The revocation of a certificate and the consequent broadcasting of the updated Certificate Revocation List (CRL) is the process which allows vehicles to recognize and discard messages transmitted from malicious actors. An updated CRL is issued when the PKI servers receive a request from the MEC identifying a new vehicle identity to revoke.

This use case is the continuation of the UC2.2 and therefore the involved actors are the same: attacker, cooperative car, fixed infrastructure, PKI infrastructure, MEC, anti-hacking device.

For more details, please, refer to section 3.2.

3.4.2 Use case setup

The use case setup is the same as the one presented in UC2.2.

Two geographical regions equipped with Accelleran Small Cells and 802.11p RSU, connected to the VLAN switch, the MEC and the dRAX. Two OBUs, one with only LTE-Uu radio and the second with 802.11p and LTE-Uu radio interfaces, the PKI servers, the Backend server, and the laptops.

For more details on the test setup of the use case, please, refer to section 3.2.

3.4.3 Use case workflow

In the following Figure 89, it is explained the Certificate Revocation workflow:

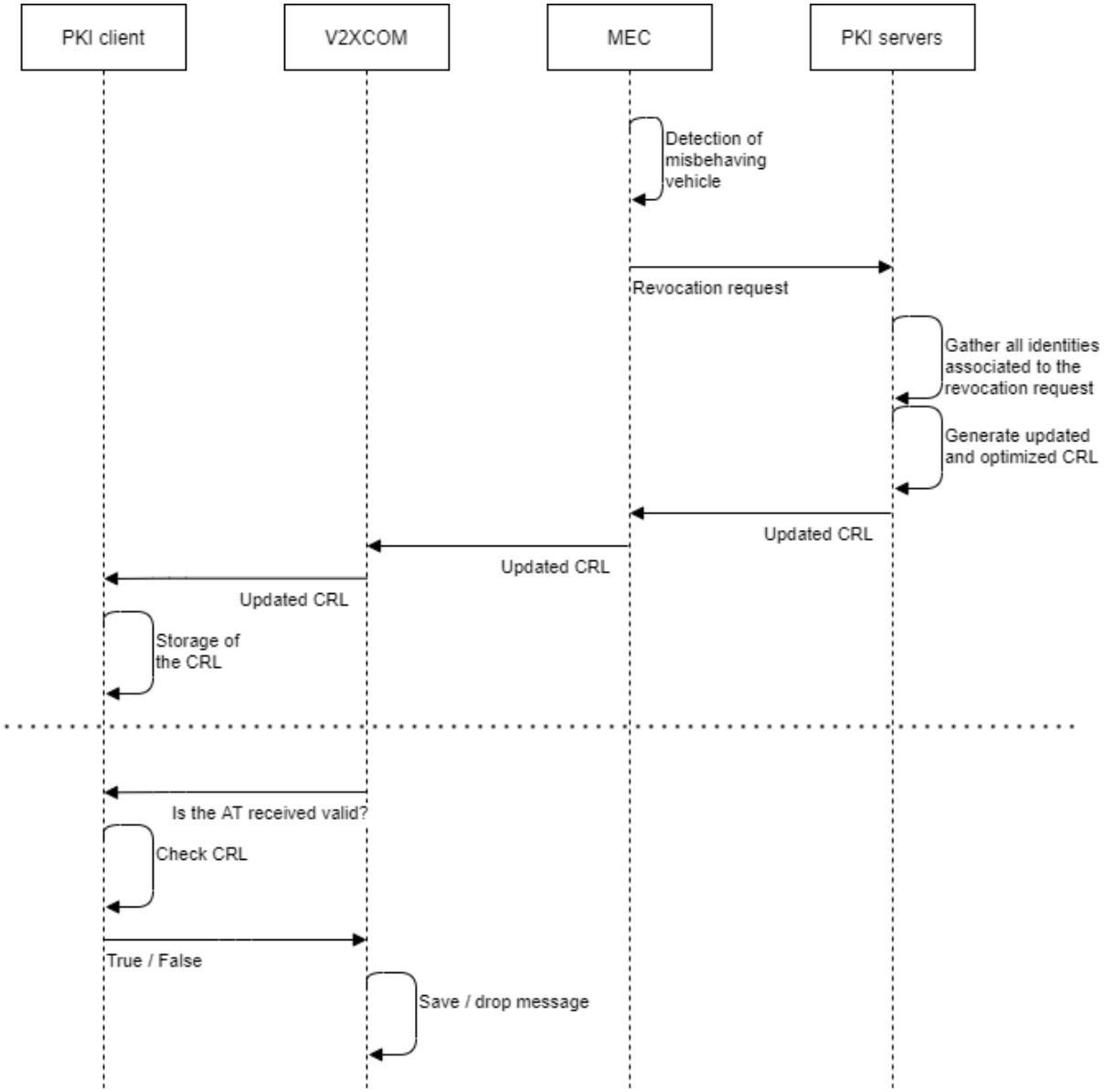


Figure 89: Certificate Revocation workflow.

3.4.4 Results

3.4.4.1 *Requirements*

Secure association. The system shall be able to establish a communication channel between itself and another ITS station such that they can exchange messages according to negotiated security parameters, i.e., message digital signature using Authorization Tickets must be applied for V2-X communication.
Identity Management. The system shall support simultaneous change of communication identifiers (like station ID, network ID, MAC address) and credentials used for secure communications, within the ITS station.
Secure transmission of ITS messages (CAM, DENM) between vehicles (V2V or V2I2V) and between vehicles and infrastructure (V2I), using BTP and GeoNetworking protocols. Every ITS transmitted message shall be signed using one Authorization Ticket.

3.4.4.2 *Successful end condition*

- PKI servers produce an updated CRL containing all the identities assigned to the revoked vehicle.
- Vehicles drop fake messages and prevent safety applications from being misinformed.

4 Pillar 3 scenario driven attacks

Apart from investigating the impact of CARAMEL's engine in robustizing the immunity of autonomous vehicles from cyber-attacks oriented towards the perception and communication layer, CARAMEL has also contributed solutions related to detecting and mitigating attacks at the charging layers of electric vehicles. Chapter 4 discusses the methodology developed, the use cases and the evaluation protocol followed to assess the efficacy of the mitigation strategies, developed in Pillar 3.

4.1 *Smart Charging Abuse: attack use case description and assessment*

The attacker(s) occupy (physically or remotely) the available charging stations starting and proceed timely in connection/disconnection actions creating an enormous load to the power grid.

4.1.1 Introduction

In this use case we aim at demonstrating a machine learning pipeline that is capable of detecting anomalies in communication between an EV charge station and its remote back office.
Of the charge stations that are currently installed worldwide, a part consists of outdated hardware. These legacy charge stations can no longer always be updated (hardware or software) so that they meet the latest security standards. It is, therefore, possible for an attacker to either locally take over the charge station or to intercept the communication in one way or another and thereby, for example, be able to send false smart-charging control signals to such a charge station.
It is therefore important to be able to remotely detect irregularities in the behavior of a charge station. It is typical for data from charging stations that these are relatively many messages, of which only a few can be labelled as outliers. To be able to detect these properly, a semi-supervised machine learning pipeline has been developed and its workflow can be seen in Figure 90.

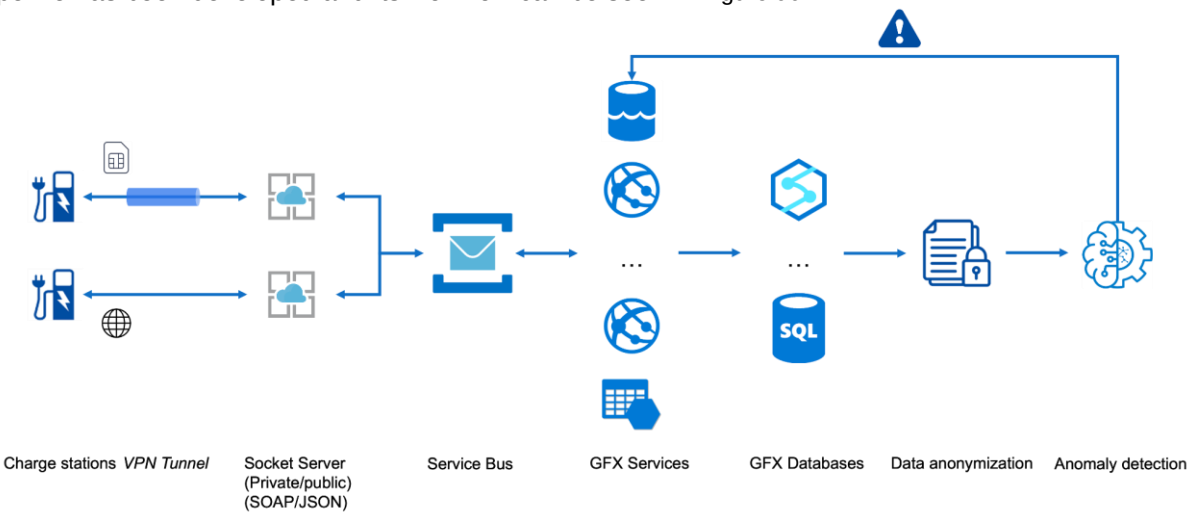


Figure 90: Overview of the demonstration pipeline.

4.1.2 Involved actors

The involved Actors are:

- **Attacker:** In this test setup the attacker is represented by a Greenflux (GFX) employee who can:
  1) Tamper with a (virtual) charge station.
  2) Inject fake data in communication with back office.
- **EV:** A physical EV is required to participate in a charging session. A simulator will not suffice because it cannot draw enough power, which would result in unrealistic data.
- **Charging Infrastructure:** These are the other charge stations. During the test, these will generate data, but they are not the target of an actual attack.

- **GFX Platform:** This is the cloud platform that communicates with all charge stations. Incoming and outgoing messages are processed and enable remote control of the charge station. It also automatically forwards the incoming messages to the anomaly detection tool.
- **Anomaly detection tool:** This is the semi-supervised machine learning pipeline developed within CARAMEL that should detect strange messages. It runs on a server in Berlin.
- **Operator:** This is a person who operates the EV Portal and can thus manage the charge stations remotely. He is also able to disconnect from a compromised charge station.

4.1.3 Machine learning (ML) pipeline test setup

Test facility at GFX office in Amsterdam: There is a parking lot with 8 charge stations from different manufacturers next to the GFX office in Amsterdam. These charge stations can be used for various test purposes. During the test, one of the charge stations will be attacked.

GFX Platform: The attacked charge station is connected to the platform. For each charge station, a certain sequence of messages is expected at a certain interval.

Databases & API: Data from the charge stations is pre-processed, stored in one of the databases and forwarded via the platform APIs.

ML tool: The ML tool processes incoming messages and determines whether there is unexpected behavior. If this is indeed the case, an outlier is recognized and reported to the operator.

Operator: With an incoming alert, the operator will have to determine whether there is indeed an attack and, if necessary, he can break communication with the charge station and inform the corresponding CPO.

4.1.4 Flow of activities

The workflow is:

1. Previous days to the attack: Data collection to be used as normal data for reference.
2. On the day of the demonstration: Attack to the charge station.
3. After attack: Sending the generated data to the ML tool in the same way.
4. Result: If it works, it will be able to attack the charge station.

4.1.5 Success end condition and test results

Machine learning tools can detect:

1. Data generated from tampered charge station.
2. Fake data inserted by attacker.

4.2 *EV Scheduling Abuse: attack use case description and assessment (UPAT)*

4.2.1 Use case description

This theoretical use-case aims to demonstrate the effectiveness of coordinated EV charging against cyberattacks, in a PC simulated environment.

Consider that an EV load aggregator of a utility company tries to coordinate the charging of several EVs. Valley-filling task is a common objective that it can be achieved through coordinated charging. In the valley-filling problem, the goal is to flatten the given load demand profile of the aggregator as much as possible, by filling the overnight valley in the load demand with the demand caused by the EVs. Let M be the EVs that need to be charged over a charging time T, comprised of consecutive time slots such as $T = \{1, \dots T\}$. The duration of each time slot is the same and equal to ΔT. Let $e_m(t)$ be the energy charge of EV m ($m = 1, \dots M$) at time $t \in T$. The energy charge can range from zero to its maximum value $e_m(t)$. Let $e_m(t) = [e_m(1) \dots e_m(T)]^T \in R^T$ be the charging profile of vehicle m. Moreover, we

define set E_m, such as $e_m \in E_m = \{e_m : {e_m}^T 1 = R_m, 0 \leq e_m(t) \leq \underline{e}_m(t), \forall t \in T\}$, where R_m is the total energy needed by EV m. The desired energy R_m is equal to:

$$R_m = \frac{(B_m \times (s_m(T) - s_m(0))}{c_m \Delta T}$$

where $B_m, s_m(T), s_m(0), c_m$ and ΔT are the battery efficiency, the expected state of charge at the end of charging horizon, the initial state of charge, the charging efficiency and the time slot duration of charging EV m, respectively.

Finally, the optimal EV charging problem can be defined as follows:

$$\underset{e_m}{\text{argmin}} \quad C(\{e_m\}) = \sum_{t=1}^{T} C_t\left(d(t) + \sum_{m=1}^{M} e_m(t)\right)$$
$$s.t \quad e_m \in \mathcal{E}_m, \ \forall m = 1 \ldots M$$

where energy costs $C_t(x) = x^2/2$ are convex and differentiable for all t, while $d(t)$ capture the based load for the EV aggregator.

A decentralized infrastructure supporting the communication between the aggregator and the EVs is preferable to tackle the optimal EV charging problem, due the lower computational requirements. CVX software, Frank-Wolfe (FW), Projected Gradient Descent (PGD) or Alternating Direction Method of Multiplies (ADMM), can be utilized to estimate the optimal charging profile for each EV.

FW and PGD algorithms [14][15], summarized in Algorithm 1 and 2, are state-of-the-art decentralized charging protocols that try to minimize the EV charging problem under FDI attacks on the individual EVs. They both directly solve the optimal problem in a decentralized manner. ADMM based approach [16][17], summarized in Algorithm 2, reformulates the EV charging problem to a decentralized exchange problem between EVs and the aggregator.

According to Algorithm 2, $F_a(\cdot)$ is the aggregator's convex objective function, e_a is the aggregated charging profile of EVs, $F_m(\cdot)$ is the convex objective function of the individual EV, E_a is the constraints set of the aggregator, $\rho > 0$is the augmented Lagrangian multiplier, $e_{avg}$is the average charging profile of the agents (both EVs and aggregator) and $u$is the dual variable. Parameter $\gamma$is in fact the trade-off between the objective goals of the aggregator and the EVs, represented by the two objective functions. It should be noted that in the valley-filling task, the individual EV goals should be ignored. In that case $\gamma = 0$ and $F_m = 0$. Moreover, the main objective of the aggregator is, in fact, to minimize the EV charging problem. As such, $F_a = C(\{e_m\})$. Finally, $E_a = \emptyset$. Both the state-of-the-art decentralized charging protocols are depicted in Figure 91.

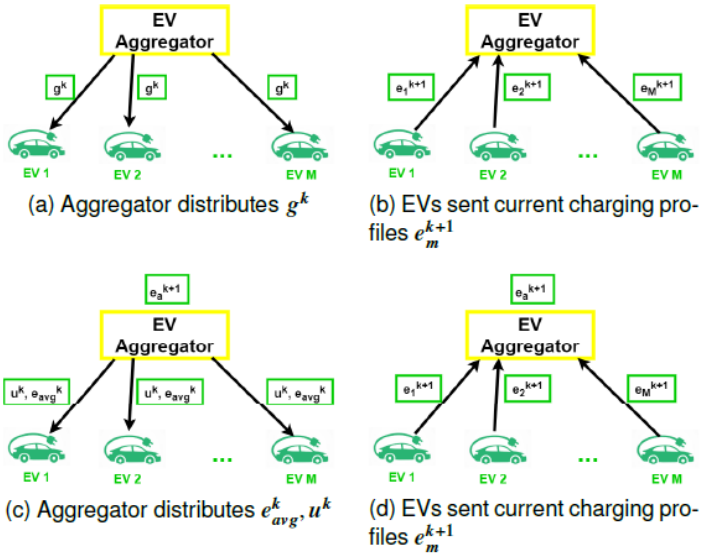


Figure 91: Decentralized charging protocols with FW and ADMM.

The three approaches are in fact iterative methods, in which the aggregator and the EVs constantly exchange their estimations in order to minimize the EV charging problem. A malicious cyber attacker can take advantage of it, since it can modify what EVs send on every iteration and destroy the entire charging process. This modification could be performed by the instalment of a malicious software on EVs. As such, we define the FDI attack on EV m as the modification of ${e_m}^k(t)$ at iteration k. This attack has an impact on the performance of the two protocols, since the quantities g^k, ${e_{avg}}^k$ and u^k also change. Note that t must belong to the time where m was to be charged, namely starting over midnight and ending early in the morning (valley-filling). Moreover, according to [16], if $e_m(t) < 0$, then EV m is feeding energy (discharge) back to the power grid.

Motivated by that, and by the modelling of FDI attacks [18] (as adding a bias to an estimated quantity), we define two types of attacks aiming to discharge, instead of charge, the EVs:

Algorithm 1: FW

Input: $M, \mathcal{E}_m, R_m, T, K_{max}$
Output: $e_m, \ \forall m = 1 \dots M, e_{total} \in \mathbb{R}^T$

for $k = 0 \dots K_{max}$ **do**
  $\eta_k = 2/(k+2)$;
  $g^k = \nabla_{e_m^k} C(\{e_m^k\}) = \sum_{t=1}^{T_m} \left(d(t) + \sum_{m=1}^{M} e_m^k(t)\right)$;
  for $m = 1 \dots M$ **do**
    $r_m^k = \underset{r_m}{\operatorname{argmin}}\, r_m{}^T g^k, \quad s.t \ r_m \in \mathcal{E}_m$;
    $e_m^{k+1} = e_m^k + \eta_k(r_m^k - e_m^k)$;
  end
end
for $m = 1 \dots M$ **do**
  $e_m = e_m^{K_{max}}$
end
for $t = 1 \dots T$ **do**
  $e_{total}(t) = \left(d(t) + \sum_{m=1}^{M} e_m(t)\right)$
end

$$Attack\ 1: \quad {e_m}^k(t) = \begin{cases} -{e_m}^k(t), & if\ t \in [00:00,\ 08:00] \\ e_m(t), & otherwise \end{cases}$$

$$Attack\ 2: \quad {e_m}^k(t) = \begin{cases} -R_m, & if\ t \in [00:00,\ 08:00] \\ e_m(t), & otherwise \end{cases}$$

With **Attack 1**, the compromised vehicle simply discharges, while in **Attack 2** the vehicle may become non-operational since it is required to feed back to the grid an amount of energy equal to that of its desired energy.

Algorithm 2: PGD

Input: $M, \mathcal{E}_m, R_m, T, K_{max}$
Output: $e_m, \ \forall m = 1 \dots M, e_{total} \in \mathbb{R}^T$

$\acute{\eta} = 1/M$;
for $k = 0 \dots K_{max}$ **do**
  $g^k = \nabla_{e_m^k} C(\{e_m^k\}) = \sum_{t=1}^{T_m} \left(d(t) + \sum_{m=1}^{M} e_m^k(t)\right)$;
  for $m = 1 \dots M$ **do**
    $e_m^{k+1} = \underset{e_m}{\operatorname{argmin}} \left\| e_m - (e_m^k - \acute{\eta} g^k) \right\|_2^2, \quad s.t \ e_m \in \mathcal{E}_m$;
  end
end
for $m = 1 \dots M$ **do**
  $e_m = e_m^{K_{max}}$
end
for $t = 1 \dots T$ **do**
  $e_{total}(t) = \left(d(t) + \sum_{m=1}^{M} e_m(t)\right)$
end

Algorithm 3: ADMM

Input: $M, \mathcal{E}_m, R_m, T, K_{max}$
Output: $e_m, \ \forall m = 1 \dots M, e_{total} \in \mathbb{R}^T$

$\rho > 0$;
for $k = 0 \dots K_{max}$ **do**
  for $m = 1 \dots M$ **do**
    $e_m^{k+1} = \underset{e_m}{\operatorname{argmin}} \gamma F_m(e_m) + \frac{\rho}{2} \left\| e_m - e_m^k + e_{avg}^k + u^k \right\|_2^2, \quad s.t \ e_m \in \mathcal{E}_m$;
  end
  for *Aggregator* **do**
    $e_a^{k+1} = \underset{e_a}{\operatorname{argmin}} F_a(-e_a) + \frac{\rho}{2} \left\| e_a - e_a^k + e_{avg}^k + u^k \right\|_2^2, \quad s.t \ e_a \in \mathcal{E}_a$;
  end
  $e_{avg}^{k+1} = \frac{1}{M+1} \left(e_a^{k+1} + \sum_{m=1}^{M} e_m^{k+1}\right)$;
  $u^{k+1} = u^k + e_{avg}^{k+1}$;
end
for $m = 1 \dots M$ **do**
  $e_m = e_m^{K_{max}}$
end
for $t = 1 \dots T$ **do**
  $e_{total}(t) = \left(d(t) + \sum_{m=1}^{M} e_m(t)\right)$
end

4.2.2 Use case experimental setup

We evaluated the three protocols using the 2014 average residential load profile in the service area of South California Edison as the base load $d(t)$. A day long charging horizon time starting at midnight, was divided into $T = 96$ time slots and thus, $\Delta T = 15min$.We choose $M = 80$, $\underline{e}_m(t) = 3.45kWh, c_m = 0.9kWh, s_m(0) \sim U(0.85, 0.95)$ where U is the uniform distribution.

The parameters were selected according to the state-of-art approaches [14][15][17], while choosing a different M [14] (i.e., 59 or 120), doesn't seem to seriously affect their performances.

CVX solver is used to tackle (1) in a centralized manner and to minimize the sub-problems of the three Algorithms.

4.2.3 Use case evaluation

In Figure 92, the total load demand produced by the three protocols and the centralized scheme of CVX, without FDI attacks, is depicted. Total (or aggregated) load demand is defined as the sum of aggregator load demand and the charging profiles of EVs. It is obvious that valley-filling has been achieved, since the load demand curve has been flattened and EV can be charged efficiently during night. In the following, we apply the two types of attacks. We assume that a subset of EVs can be compromised, and we measure the relative total load demand error between the decentralized protocols respectively and the intact centralized scheme of CVX, which acts as the baseline protocol.

In Figure 93 and Figure 94, the total demand under the attacks on 40% of EVs is presented. It is evident that the performance has been seriously degraded since the curve does not flatten at all. Clearly, the impact of attacks affects the overnight charging.

However, in all cases, except Figure 94(a), the corresponding curves are close enough to that of Figure 92. In Figure 94(a) and under Attack 2, FW seems that fails to perform the valley-filling task, as the resulting total load curve is clearly affected by the attack. Furthermore, we demonstrate in Table 16 and Table 17, the relative total demand errors, under the two types of FDI attacks on 10%, 20%, 30% and 40% of EVs, respectively. Under both Attacks, and as expected, the errors of the protocols are quite small and become larger as the number of compromised EVs increases. The value of the error characterizes the impact of the attack which is reduced and significantly mitigated when the error is small.

At this point it should be noted that the ADMM error in the Attack 1 scenario where the 30% and 40% of the vehicles are attacked, is the smallest one between those achieved by the other two approaches, while when 10% and 20% of the vehicles are attacked the errors of all the approaches are almost identical.

In the Attack 2 scenario, FW error is by far the greatest, and therefore, the valley-filling task totally fails to be performed. Once again, ADMM seems to outperform PGD, since its error is very small and remains almost constant, regardless of the number of attacked vehicles.

In all cases, ADMM and PGD have been executed efficiently succeeding to perform the valley-filling task during EV charging, proving their robustness under different FDI attack scenarios. The fact that the robustness has been achieved without any extended modification of the initial schemes, is crucial to the safety and economic cost of EVs, since special hardware doesn't seem to be an important requirement.

Algorithm	10% under attack	20% under attack	30% under attack	40% under attack
FW	8	18	29	33.6
PGD	8	18	28	37.2
ADMM	8	18	23	27.7

Table 16: Error of EV charging under Attack 1 (10^{-3}).

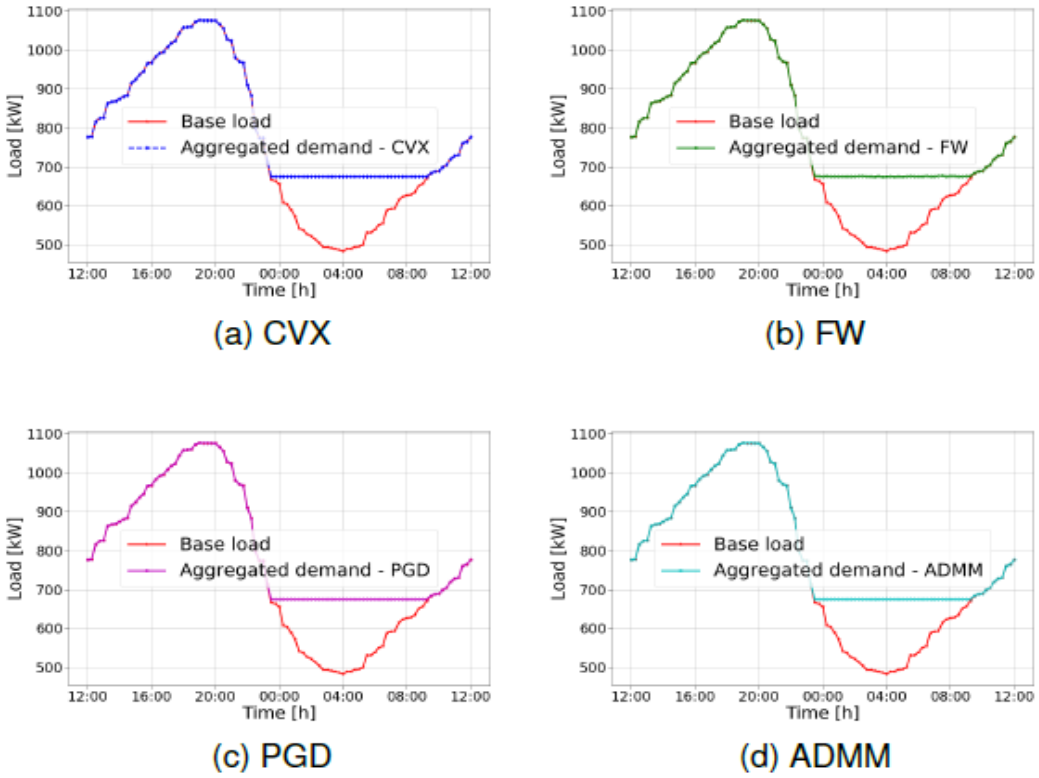


Figure 92: Total or aggregated load demand with CVX, FW, PGD and ADMM, without attack.

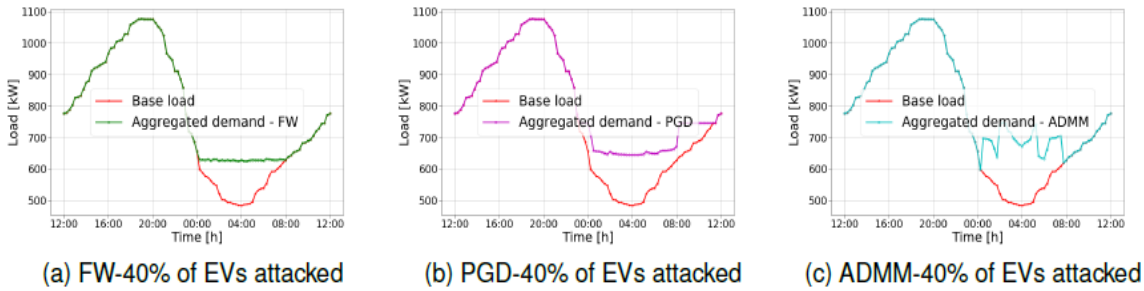


Figure 93: Total load demand under Attack 1.

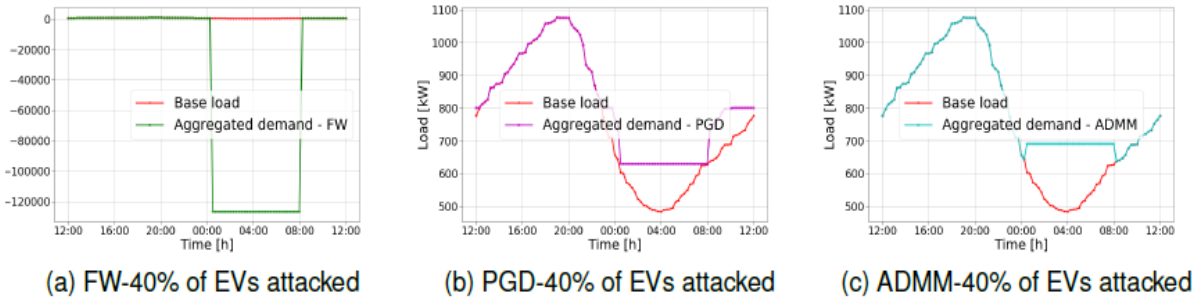


Figure 94: Total load demand under Attack 2.

Algorithm	10% under attack	20% under attack	30% under attack	40% under attack
FW	19506	41689	60934.7	87105
PGD	9	25.3	38.1	58.8
ADMM	15	12.6	14	12.2

Table 17: Error of EV charging under Attack 2 (10^{-3}).

5 ANNEX: Threat Awareness Dashboard and Backend Infrastructure.

Autonomous driving is an emerging safety-critical technology, where vehicle software is tasked to navigate based on data inputs, while at the same time defending against foreign malicious cyberattacks. While onboard attack detection and mitigation is important as the first line of defense, an external supportive cybersecurity platform can provide additional layers of security. Such platforms, called Security Operation Centers (SOCs), can provide important cybersecurity analytics, and can organize a faster attack response with relevant stakeholders.

It would be rather optimistic to assume that in the future all vehicles will be equipped with the technology to detect any possible cyber-threat. Also, the detection of certain threat types could be unreliable, for instance due to sensor errors. This raises the need for a collaborative cybersecurity approach, with SOCs as aggregators and service providers. Attack detections can be collected from multiple vehicles on the street, cross-validated, filtered and assigned a confidence level. SOC service providers can then push their advanced threat intelligence back to their customers with a high degree of confidence.

5.1 *Demonstrator elements*

Within the framework of the CARAMEL project, Capgemini Engineering developed the following components.

5.1.1 Backend

The backend can monitor a fleet of vehicles, real-world or simulated. Vehicles can communicate with the backend to submit and retrieve data and threats. The backend collects threat information from multiple sources, analyses the data and assigns a confidence level to each threat based on the observation frequency and threat type. This enables a form of collaborative awareness, which improves the reliability of threat detection by crowdsourcing information from multiple vehicles. More information about the backend can be found in the document “D4.4 Report on the Fallback Actions for Minimal Risk Conditions.” This infrastructure was offered to all CARAMEL partners to support their respective demonstrators.

5.1.2 Dashboard

The main CARAMEL dashboard aggregates and displays all threats that are collected by the vehicles. For this demo, an additional in-vehicle HUD was created in the simulated environment to display threats that are retrieved from the backend.

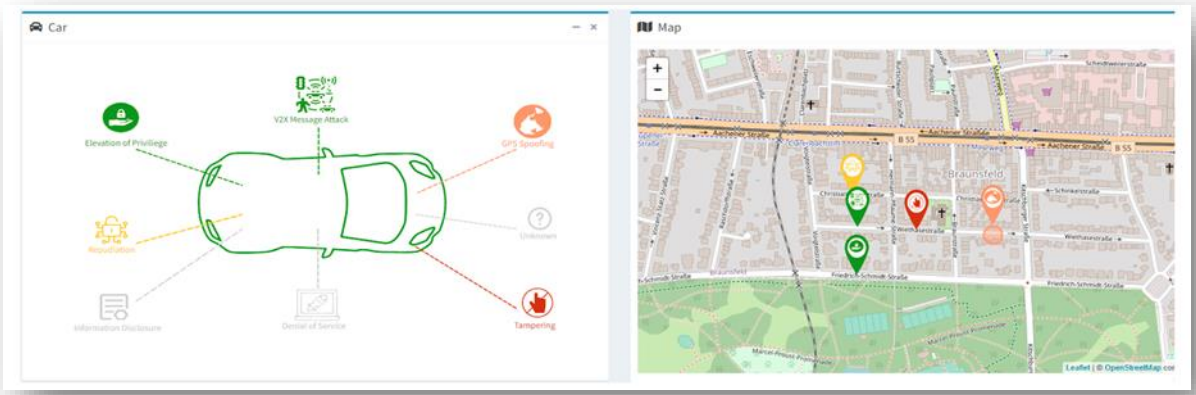


Figure 95: The CARAMEL dashboard with vehicle-specific threat display (left) and location-specific threat intelligence (right).

5.1.3 Simulated Environment

To accelerate development and to recreate various scenarios, a simulated environment was generated using the CARLA framework replicating real-world driving conditions, including buildings, traffic signs, roads, etc. There, virtual vehicles can be deployed with simulated RGB and GPS sensors. To submit and retrieve application relevant data the simulation is connected to the internet. In addition, the communication takes place per the SIEP protocol as defined in the document “D4.4 Report on the Fallback Actions for Minimal Risk Conditions.” The required libraries are also installed in the environment. For ease of use-, demonstration-, and testing purposes the following supportive features were also integrated:

- Keyboard-controlled traffic sign changes.
- Keyboard-controlled tampering of traffic signs.
- Threat alert visualization.
- Built-in painting tool to manually tamper the traffic signs.
- Automatic vehicle navigation along a predefined route.
- Predefined regions where simulated GPS spoofing attacks take place.

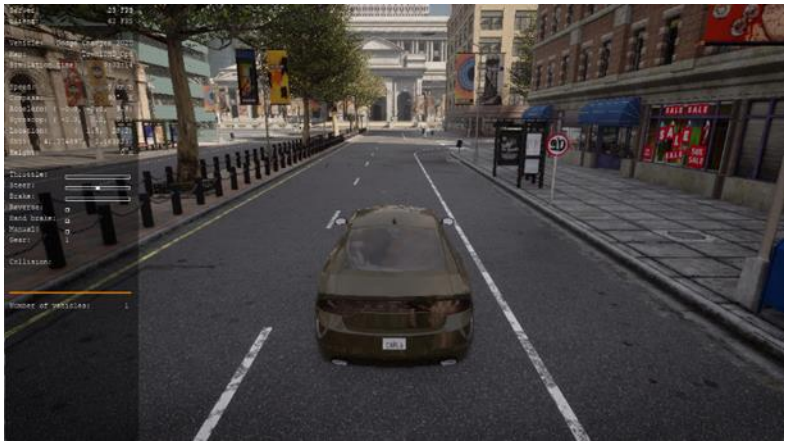


Figure 96: Vehicle navigating the CARLA virtual environment.

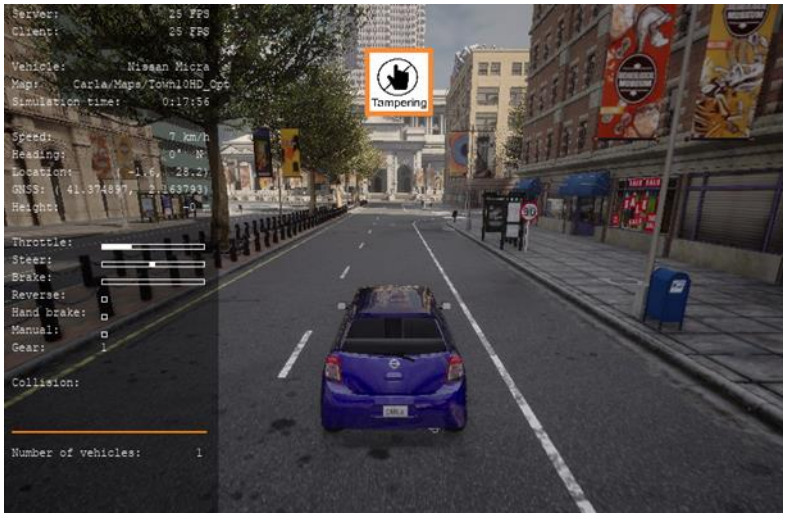


Figure 97: Simulated threat display.

5.1.4 Attack Detection & Generation - Sign Tampering

An object detection system that allows the recognition of traffic signs was integrated into the simulation. The TensorFlow-based Object Detection API was used to process and infer the collected images. The

open-source AI model “Faster R-CNN Resnet 101” was used to recognize traffic signs within the CARLA simulator. Detected tampered signs were submitted to the backend according to the SIEP protocol.

Figure 98: Simulated tampered sign attack within the CARLA environment.

5.1.5 Attack Detection & Generation – GPS spoofing

Demo vehicles equipped with onboard GPS spoofing attack detection capabilities submit the threats they detect to the backend. There, the information is processed and overlapping areas are stitched together into aggregated threat zones. The threats are then rated in terms of detection confidence, based on the frequency of observations from the vehicles. If a significant percentage of vehicles crossing through a suspected spoofing area detect and submit the threat, then the backend assigns a high confidence level to it. Similarly, if only a small fraction of vehicles detects it, then the threat is treated with low confidence. Only medium and high confidence threats are pushed to vehicles subscribed to the backend. On the threat awareness dashboard, a color scale is used to visualize the different threat levels.

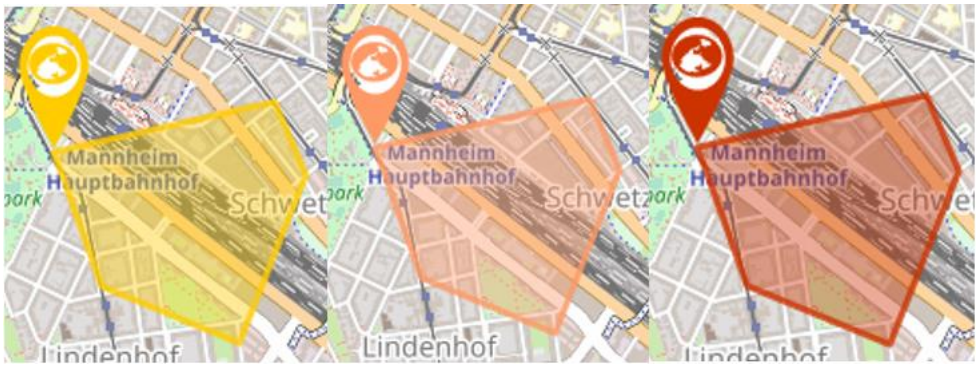


Figure 99: Various detection confidence levels of a GPS-spoofing attack in the CARAMEL dashboard. Low confidence in yellow, medium confidence in orange, high confidence in red.

5.1.6 Small displacement spoofing (SDS) awareness dashboard Proof of Concept

Vehicles can determine their location in several ways. GPS is the most common localization technology, but more sophisticated methods, such as combining information from high-definition maps with object detection techniques, offer more accurate ways of localization. By comparing and combining information from different location sources, vehicles can reliably determine their position on the road. GPS signals can be spoofed. Vehicles can detect large anomalies in their GPS position by fusing location data from various sources, such as nearby Wi-Fi and cell tower signals. This is not the case for small displacements, which fall within the range of typical GPS accuracy of around 5 meters. At this point, vehicles cannot distinguish whether these deviations occur due to sensor malfunction, physical obstacles, or a malicious attacker. Under the assumption that under normal circumstances the difference between the GPS position and other location sources is random and that under attack the deviations of each spoofed vehicle follow a common direction pattern as shown in Figure 100. Capgemini

Engineering created an interactive visual dashboard as a proof of concept to demonstrate the collaborative method in which the team envisions the collection and visualization of SDS attacks. In Figure 101 the blue area demonstrates the scanner function that iterates through the map area for similar deviations. Work on this concept will possibly continue as part of the GAIA-X 4KI project.

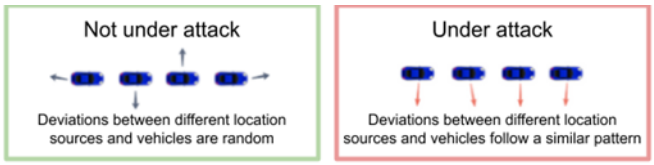


Figure 100: Vehicle location deviations under different scenarios.

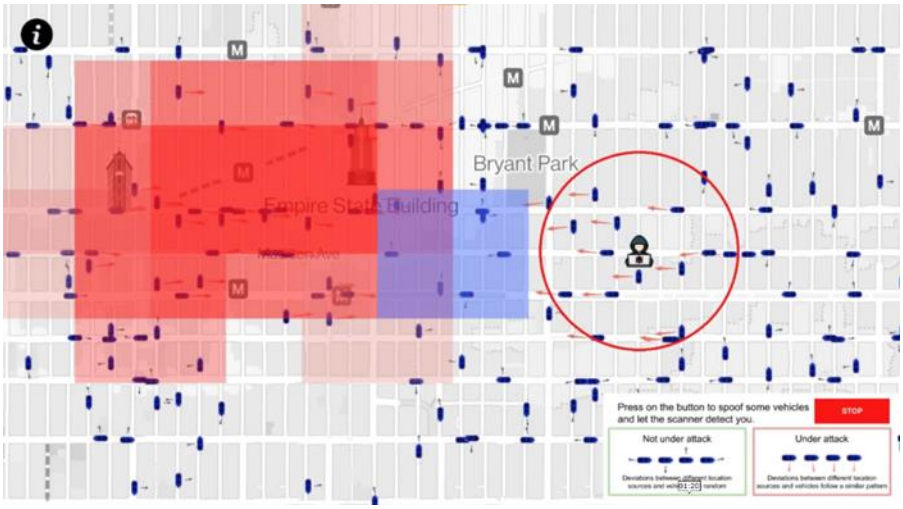


Figure 101: SDS Proof of Concept dashboard. The red area color intensity indicates the detection confidence level.

5.2 *Attack Vectors*

In the scope of this work, the following two attack scenarios are considered:

5.2.1 Sign Tampering

Autonomous vehicles rely on traffic sign information collected through onboard cameras to make safety-critical decisions. Tampered traffic signs, regardless of malicious intent or not, can trick the vehicle into taking wrong and dangerous driving decisions, for instance, assuming a false speed limit.

5.2.2 GPS Spoofing

GPS technology is now a mature technology and the standard way for vehicles to efficiently navigate between locations. Still, GPS devices can be vulnerable to cyber-attacks through GPS spoofing. Spoofing happens when a malicious actor uses a radio transmitter to emit a counterfeit GPS signal to a receiver antenna, such as the one within a vehicle, to counter and overpower a legitimate GPS satellite signal. Most GPS-based navigation systems are designed to use the strongest available GPS signal, and so they are vulnerable to spoofing attacks. The attack can be subtle enough for the driver or vehicle to not realize and it could lead them to steer off course without any coercion.

Recently, autonomous driving technologies have created the necessity for more accurate and reliable ways of localization. For example, high-definition maps together with object detection techniques can help vehicles position themselves on the street more accurately. These methods usually have multiple input sources and so they are more robust against attacks as is demonstrated by other partners. Nevertheless, it is expected that these technologies will complement and not replace traditional pure-

GPS localization, due to the additional computational effort and availability of data. Here, we assume that vehicles transmit two location streams.

5.3 *Demonstrated Use Cases*

The added value of the Capgemini Engineering backend solution will be demonstrated through the following two use-cases.

5.3.1 Attack awareness – Sign tampering and GPS spoofing

We will demonstrate how smart vehicles with threat detection capabilities can notify the backend about attacks they have witnessed in route. After processing, the backend can then push threat alert notifications to subscribed legacy vehicles that are less equipped to detect threats. We show how these vehicles can be informed in advance about attacks in their vicinity. Both road sign tampering and GPS spoofing attacks will be used to demonstrate.

5.3.2 Attack detection through collective intelligence – GPS spoofing

Here, the ability of a monitoring backend solution to process both hyperlocal and larger surface area threats from multiple sources will be demonstrated. A simulated GPS Spoofing attack will be used to demonstrate.

Smart vehicles with onboard GPS spoofing attack detection capabilities submit location data abnormalities that they detect to the backend. The backend aggregates the data into distinct danger zones and assigns a danger level based on the consensus of the observations. The more ground truth vehicle data is available, the more confident the prediction. In practice, if the majority (or other suitable statistics/thresholds) of smart vehicles detect a GPS spoofing attack, then the corresponding area is marked as dangerous and other incoming legacy vehicles are informed of the threat.

5.4 *Software Pipeline*

The process pipeline and the software building blocks are similar for both use-cases. Vehicles driving in a simulated environment, the CARLA Simulator, submit their data to the backend through a Python-based communication API. Traffic signals are detected using image recognition techniques such as “faster RCNN (resnet 101)” implemented in TensorFlow. For the GPS spoofing use case, only the locations of the vehicles are submitted to the backend. This use case will be discussed in more detail in the following section. The Python-based backend verifies and processes the submitted messages before forwarding them to the custom-built CARAMEL dashboard. Identified threats along the trajectory are published to an MQTT communication broker from where legacy vehicles can retrieve and display them on an in-vehicle HUD. The scenarios can be run both in a simulated and a real-world setting.

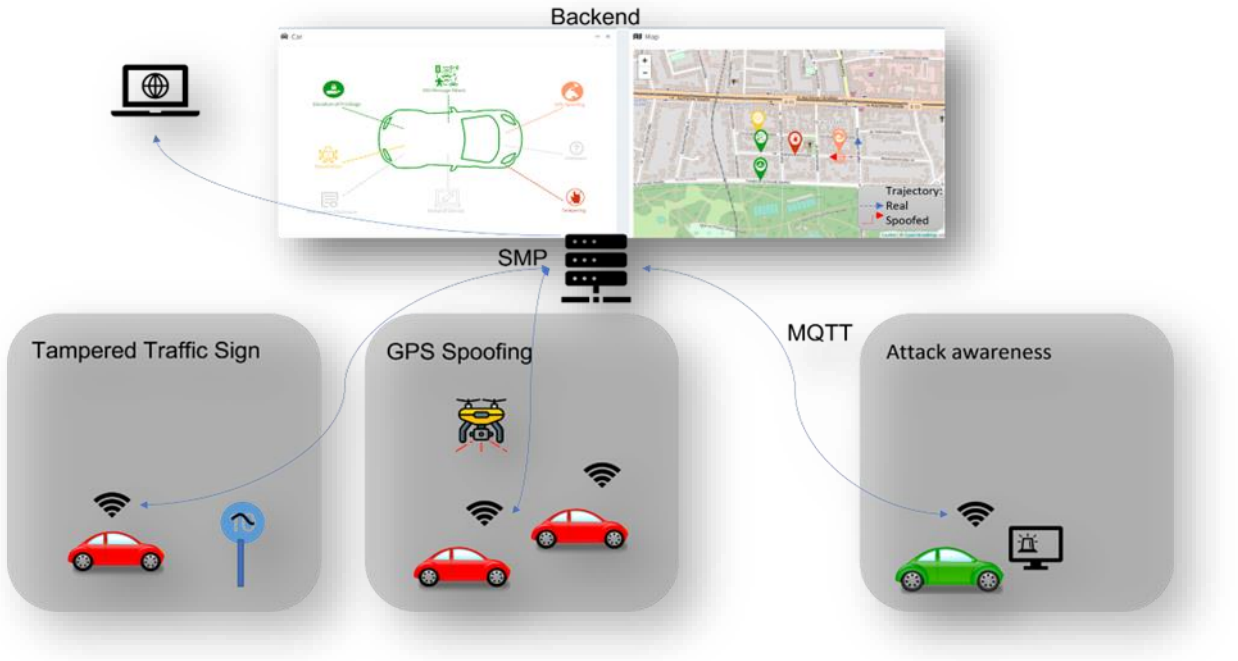


Figure 102: Communication architecture with a Security Message Protocol (SMP).

5.5 *Demonstrator storyboard*

To demonstrate the use cases, three simulated vehicles are employed. We showcase the following capabilities:

- Threat collection.
- Threat analysis, aggregation, and severity ranking.
- Threat alert notifications.
- Threat awareness and visualization dashboard.
- Road sign detection.
- Communication and data exchange between the tools.

The following storyboard outlines the series of events that we have orchestrated to highlight the full value of our work:

5.5.1 Scene 1 – Smart Vehicle 1

Smart Vehicle 1 is assumed to be equipped with an advanced sensor package and map matching technology, that allows it to generate an untamperable location data stream (at least in environments with suitable landmarks) and hence detect GPS spoofing attacks. The vehicle is also connected to our backend service.

While driving downtown on the highway of a simulated city, the vehicle enters an area, where its GPS signal does not match its untamperable location. The vehicle notifies the backend of this discrepancy. The area is flagged as suspicious. The backend does not assign a high-risk level to the threat because it lacks enough data points. The data needs to be cross-validated with other smart vehicles first, which is where the backend-powered statistical intelligence shines. Besides, the observation could have potentially been a sensor miscalibration or signal interference with the surrounding buildings.

Further ahead, a traffic sign has been sprayed with paint to trick the vehicle's autonomous driving system. The vehicle detects the threat, as it is equipped with onboard tampered sign detection technology. The threat is sent to the backend, where it gets stored in the database. As this kind of threat

can be reliably detected from a single vehicle, the backend immediately assigns a high-risk level to the threat.

A few meters ahead an attacker has placed a transmitter device nearby, distorting GPS signals. The vehicle enters the area and again detects the discrepancy. The data is sent to the backend and again the area is flagged as suspicious. The threat database is starting to build up. At the same time, all threats are displayed on the threat awareness dashboard.

5.5.2 Scene 2 – Smart Vehicle 2

A few minutes later Smart Vehicle 2 enters the same street. Like Smart Vehicle 1, it too is equipped with an advanced sensor package, and it is also connected to the backend.

The vehicle enters the first suspected spoofed area but does not detect any discrepancies, indicating no spoofing activity. Since there is no consensus between this and the previous vehicle, the backend does not elevate the risk level of this threat.

The vehicle moves on and enters the second suspected spoofed area. Here, the discrepancy is detected. The backend processes the data from both vehicles, readjusts the suspicious attack zone and elevates the threat level to high risk. In a realistic, large-scale scenario, it is expected that significantly more vehicle observations will be required to update the risk profile of a threat.

5.5.3 Scene 3 – Legacy Vehicle

A third, legacy vehicle enters the street. It lacks advanced onboard sensors and is vulnerable to GPS spoofing and sign tampering attacks. The driver has subscribed to a cybersecurity service powered by the backend for extra protection.

As the vehicle drives through the first suspected GPS spoofed area the backend does not notify the vehicle, because the confidence level is low. As it approaches the tampered sign though, the backend pushes a threat alert. Later, as the vehicle approaches the second GPS spoofed area the backend again sends an alert in advance. Both vehicle and driver are now aware of the dangers and can exercise caution.

6 Roadmap for future evolution of the CARAMEL achievements

6.1 *Introduction.*

Self-driving and connected vehicles are rapidly evolving and will eventually become the main mode of transportation for people and cargo. While from a mechanical point of view little has changed compared to traditional cars, software is at the core of this rapid transformation. These new capabilities allow for use cases, products and applications that were unviable a few years ago, but also surface problems that require careful research and design. Vehicles, whether autonomous or not, remain a safety-critical application that must guarantee the safety of their passengers. The increasing integration of software and the connectivity of vehicles with their environment creates attack surfaces that were previously unknown and must be carefully studied. It was CARAMEL's goal to address some of those challenges, provide advanced methods to mitigate them and to explore further problems that are still to be solved. In the following we will present how the roadmap of CARAMEL's pillar achievements could look.

In particular, the element of (edge-)cloud is of increasing interest in the pan-European cybersecurity ecosystem because it is the glue between stakeholders, machines, users as well as the cybersecurity risks that spill over between previously disjoint use-case playing grounds of (cyber)attacks, some of which previously not categorized as such (e.g., Pillar 1). It is also apparent that the local-minded, once-and-for all type approval process of to-be-sold devices and vehicles is insufficient. The UNECE 155, for example, advises a Cyber Security Management system to all stakeholders for good reason. So far, the law is not implemented.

Some of the solutions presented in CARAMEL are backend oriented, so it could be considered as a "Seeing is believing" style, since the implementation of cross-stakeholder, "online" messaging about threat awareness is hampered by

- political issues within and between the industrial players,
- fragmentation of the public players,
- unclear business case.

The more the path to collaborative threat awareness is laid out the more we hope to expect actual implementation across the cloud and subsequently also for companies like Capgemini and Atos (two partners from CARAMEL's Consortium) who mostly perform like integrators between the sectors, partly via the cloud.

6.2 *CARAMEL's modularity approach.*

Pillar 1 - Autonomous Mobility (AVL/UPAT/UCY/0INF/PANA/CAPGEMINI).

Over the last few years there has been a great interest in developing autonomous vehicles aiming to reduce disastrous accidents and offer improved time management during traveling. However, such systems also impose strong safety and security requirements and therefore a second safety-ensuring sub-module is essential besides the main driving components. In this project, we focused on designing and developing techniques to detect and diagnose anomalies in the environment focusing on traffic signs aiming to reduce the possible implications and consequences of such malicious actions. In the near future an extension of the system for possible environmental attacks or alterations on road lanes, parking lines and traffic lights is considered, by extending the current anomaly detection algorithms. Similar to that, anomaly detection models can be extended to handle vehicle behavior patterns utilizing also the system logs to provide further security guarantee to autonomous driving systems. This extension is aimed mainly to improve the current functionality including more environmental attacks or unexpected and unwanted alterations that may endanger the passengers or pedestrians and other vehicles. As a result, it will help to build a more complete system with improved functionality.

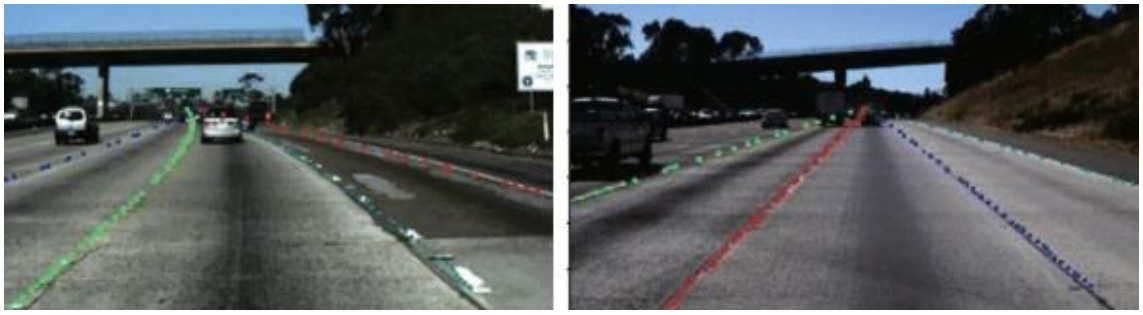


Figure 103: Road Lane line detection for environmental attacks or alterations.

Furthermore, another evolution of the CARAMEL system that is currently only for environmental attacks detection and recognition is to be modified and used for road and environment monitoring such as road damage detection for maintenance e.g., holes. Road infrastructure is an essential piece of modern society and is one of the most important public assets. This additional functionality will help cities, countries and transportation departments struggling with budget and labor limitations. These challenges affect everyone, as bad road conditions cause public safety problems and cost drivers who repair pothole damage to their vehicles. Computer vision and deep learning models will be designed and trained to interpret image data from vehicle cameras. The models for the detection and classification of pavement and road deterioration will be finally deployed and tested with the aim of calculating an index or metric. This metric will quantitatively evaluate the condition of the pavement for each section of the road.

Figure 104: Example of road quality analysis.

Finally, the roadmap of these tools and components is to be integrated into a single benchmarking system. Utilizing the CARLA car simulation - that allows dynamic and controlled environmental alterations - traffic signs and roads will be generated including normal and attacked or altered objects. This benchmarking system will allow the fair comparison of AI models in terms of performance, accuracy but mainly their robustness against environmental attacks. As a result, this integrated system could be also utilized as a cybersecurity certification process for smart and autonomous vehicles.

On another note, almost all large European car manufacturers and suppliers are customers of DT-Sec or Deutsche Telekom. DT-Sec has identified automotive security as a growth area, given that more and more domains in the automotive sector are governed by IT – may it be the connected car itself, the roadside infrastructure, or the backend systems. This increasing reliance on data communication and processing inside the car, the edge, and the cloud dramatically widens the attack surface and therefore the market for security solutions. Given this background, DT-Sec focuses on delivering tailor-made secure IT solutions to the connected vehicle such as the anti-hacking device or the embedded secure element (or HSM, hardware security module.

DT-Sec runs the Deutsche Telekom trust center for the whole Deutsche Telekom group and many external customers. Additionally, DT-Sec has its own product line of smart card and secure module products based on the Telekom Card Operating System (TCOS) that is used for the German identity and healthcare cards, for example. DT-Sec has the engineering and software development ability to integrate this HSM solution into a range of embedded devices – the anti-hacking devices of the CARAMEL project – and to help partners deploy this technology into their respective security scenarios and showcases developed for the project. DT-Sec is also running the Deutsche Telekom CERT (Computer Emergency Response Team) for the whole Telekom group which is also networked to the ENISA CSIRT network.

Results and experiences of the CARAMEL project will influence the roadmap of the DT-Sec product portfolio development for years on end and will drive the development of innovative product offerings for the automotive security market. Specifically, DT-Sec will re-use the methodology of integration of hardware security elements into in-car control units as showcased in the anti-hacking device. Leveraging its membership in the 5GAA (5G Automotive Association), specifically the working group 7 on security and the work in the misbehavior detection work item to integrate results from CARAMEL into forthcoming 5GAA documents. Additionally, DT-Sec will steer the direction of development of the TCOS platforms to enable more compatibility with automotive standards, eg. by supporting elliptic curve cryptography to enable secure handshakes with automotive control units and on-board units such as showcased in the project. The multi-layered approach to security taken in the anti-hacking device will be adapted as a blueprint for the development of future hardware solutions for the automotive product portfolio and forthcoming customer projects.

The Architecture & Innovation department inside DT-Sec carrying out the work is committed to constant dissemination of work results to the whole Deutsche Telekom Group in Germany and internationally. Members of the CARAMEL project team also take part in the internal product portfolio process and meet with members of the board on a weekly basis. Therefore, transmission of project results into the portfolio and the product roadmap is guaranteed. Project results will be preserved and made available to all of Deutsche Telekom group employees by using the widely accepted internal knowledge management platform “YAM United” where they are prominently displayed and easily be found by the advanced search functions of this platform.

Capgemini, partner with collaboration within Pillar 1 and 2, is considered an established contractual supplier of experts and solutions for Telecoms and has experts conducting worldwide projects with several major automotive manufacturers and telecom operators. This expertise in conjunction with acquired knowledge from CARAMEL’s core topics enables CAPGEMINI's portfolio diversification in areas like 5G technology, cybersecurity, artificial intelligence, machine learning, embedded software, among others. The gained experience allows proliferation of innovative technologies or consulting services in the most relevant fields of development such as autonomous driving simulations, digital twins, cybersecurity, risk mitigation, among others.

Within the activities of pillar 1, UCY has developed a deep learning approach referred to as Drive Guard, which aims to provide a solution towards detecting and mitigating cyberattacks on the camera sensors of autonomous vehicles. This approach has proven promising in robustifying perception tasks in autonomous vehicles. Building on this work the UCY team will further investigate the spatio-temporal aspects of the proposed solution by expanding it with the use of vision transformer models which can capture longer range dependencies and thus would be more useful in extracting patterns from video. Furthermore, we will also explore multi-modal approaches to train machine learning models to have a holistic situational awareness and be able to deal with uncertainties of various sensors and exploit their complementarity. Finally, an additional mechanism to develop regards quantifying the uncertainty of the perception models as well through probabilistic Monte Carlo techniques in order to have a second line of defense in case the primary solution fails with detecting and mitigating the cyberattack. A horizontal challenge across all these research directions is the strict run-time requirement. As such, we will provide further improvements on currently developed models and techniques that will be simultaneously robust but also lightweight and fast.

Throughout CARAMEL’s development and experimentation phase, it was verified that the cyber-attack detection and mitigation engine, developed during the project, can provide immunization of the system at a level of high caliber. The consortium’s solution in pillar 1 was mainly geared towards sensor-oriented approaches for mitigating the attack. While this has proven to be efficient in terms of the provided accuracy, the computational resources needed to be allocated for this type of mitigation strategy is quite

high. According to the gained experience, as a further step for robustifying cyberattack mitigation efficiency, the community should provide solutions, where data exchanged by neighboring traffic agents are being used to mitigate the attack. In this way scene understanding measurements contributed by the neighboring traffic agents can be used as priors in the process of reversing the attack. Thus, embodying IEEE 802.11p in the cyberattack evaluation and mitigation phase.

Due to the very strong demand for safety solutions by the equipment manufacturers, we expect a significant increase in the business in this area. Innovative solutions and competence demonstrations are added on top. Since we are product-neutral, we expect the proliferation of CARAMEL technologies and thus further indirect business.

Pillar 2 - Connected Mobility (i2CAT/ATOS/UBIWHERE/NEXTIUM by Idneo/UPAT/CAPGEMINI).

Pillar 2 presents solutions for three hot topics about the basis on which Intelligent Transportation Systems (ITS) relay. These are the implementation of ITS security models defined by the ETSI, the co-existence of different radio technologies to interchange V2X messages and the veracity of the positions that vehicles are announcing.

i2CAT, by developing the components of which it is responsible for, has set the roadmap of its participation in future V2X projects and services. The main component that i2CAT has developed is the ETSI ITS protocol stack containing different V2X messages, BTP and GeoNetworking protocols, its interface with a PKI architecture to manage ATs and the integration with different kinds of HSM. In the market, there are two main consumers of such a protocol stack, the car manufacturers that deploy OBUs containing the protocol stack in their vehicles, and road infrastructure operators that require the stack to be executed in servers to provide ITS services to drivers and administrations. In CARAMEL we have used the same version of the stack for both segments, in the NEXTIUM by Idneo's OBU, and in the MEC. As the deployment of the whole stack departing from zero is a huge task, we have relayed in the open-source Vanetza framework, modified and upgraded conveniently to be adapted to the specificities of CARAMEL's use cases. We have seen that commercial versions of this stack provided by Commsignia, Cohda Wireless or Lacroix are much more efficient in the OBU side since their stack has been programmed very specifically for automotive computers, which have small memory and small computation capabilities. Therefore, i2CAT's stack cannot compete in this segment. Nevertheless, these commercial stacks are closed and focused to high level functionalities, the programmer can not access to specific functions or change the parameters of the protocols, they are not flexible, and they are hard to be adapted to different scenarios. On the other hand, these are, precisely, the characteristics of the ITS protocol stack developed in CARAMEL which adapts very well when used in a MEC or other servers in the infrastructure. We have all the code, and we can adapt it to any particular situation and requirement. For instance, it can be used to build digital twins, traffic management centers, to develop any kind of ITS application. Moreover, it can be used in other types of environments, for instance to build small size OBUs, using Raspberry Pi platform, for bicycles or electric scooters. Our roadmap is to continue developing this piece of software and use it in several upcoming projects as EU project Podium starting next October, or in contracts that we are discussing with Idiada or Abertis Autopistas España.

The second module that i2CAT has developed is the system architecture that contains a MEC, connected to fixed infrastructure of different types of radio technologies, with the necessary software components to allow vehicles, using these different types of radio technologies, to communicate among themselves. In CARAMEL we have demonstrated the case of interoperability between the V2X native standard IEEE 802.11p and LTE that provides non-native V2X communication, having to use V2X messages over IP. These components are valuable for i2CAT's upcoming projects because radio technologies will continue evolving, and we need to adapt our systems to this evolution. i2CAT's immediate roadmap contains the plan to implement interoperability with a new radio infrastructure based in LTE-PC5 and, when available, also based in NR-V2X and 802.11bd. Additionally, the fact of having developed the necessary components to transmit messages over IP over LTE, allows it to switch from LTE to 5G with very little effort.

Another challenge of cybersecurity in V2V connected mobility is location spoofing. The latter attack aims to compromise the self-positioning ability of vehicles. A location spoofing attack attempts to fool a GNSS receiver by broadcasting false satellite signals, focused on resembling a set of normal satellite signals. These spoofed signals may be modified in such a way to cause the receiver to estimate its location even kms away from its actual position. The impact of this attack is more devastating if we consider a

group of connected vehicles, which exchange their location measurements in order to coordinate their actions. Broadcasting falsified GNSS positions, then severe traffic accidents are more likely to take place, injuring drivers, pedestrians, cars, etc. UPAT has developed the associated algorithmic mitigation and detection solution. The main task of this collaborative defense mechanism is to run dedicated cooperative AI solutions that work on the data transmitted to the leader ego vehicle. It is responsible to receive the measurements of the cluster's vehicles, identify and mitigate the possible attacks on the GNSS receivers and feed the re-estimated positions back to the involved vehicles. UPAT's module has been deployed within the CARLA-ROS simulated framework, using CARLA simulator to generate data from cars' sensors and ROS nodes as the data consumers. Within this approach, it is of future plan to evolve the interoperability of the defense mechanism, by integrating a realistic V2V communication simulator which models network delay. In addition, it will be explored how the current centralized implementation will be transformed to a distributed based solution, without the need of a central node, which enables scalability, lower computational and deployment cost, as well as mitigation and detection ability.

UCY developed the in-vehicle location spoofing attack detection solution. It uses a threshold-based approach (i.e., static threshold value that is selected during the training phase) for detecting suspicious deviations between the current GPS location and the GPS-free vehicle location estimate that relies on vehicle measurements and absolute vehicle location (e.g., through cellular network localization solutions). In the immediate future, the UCY team will explore Machine Learning techniques to obtain a dynamic threshold that is adapted to changing conditions. For instance, as the vehicle moves from a rural to a suburban to an urban environment the GPS location accuracy and precision degrades (because of increasingly obstructed satellites), while the accuracy of cellular-based methods that are used to estimate the GPS-free vehicle location is better due to increasing cell tower density. This dynamic threshold selection approach is expected to robustify the UCY attack detection solution, i.e., reduce false positives due to variable environmental conditions. Another research direction will be towards enhancing the solution not only to detect the location spoofing attack, but also to mitigate the attack by reconstructing the attacked GPS locations. To this end, denoising autoencoder techniques will be investigated to remove any bias introduced in the GPS locations as the result of an attack.

From NEXTIUM by Idneo side, as a solutions partner who developed SW & HW securitization methods to protect the HW (OBU) against tampering and attacks, an evolution for the hardware protection could be the usage of Physical Unclonable Functions (PUF) for the securisation. A PUF is a physical object that for a given input and conditions provides a physically defined "digital fingerprint" output. In semiconductors, due to the submicron manufacturing process variations, every transistor from the integrated circuit has slightly different physical properties that can be measured (transistor threshold voltages, gain factor, parasitic capacitances…). These variations are not controllable in the manufacturing process and using these inherent variations as inputs for certain algorithms, the silicon fingerprint is turned into a cryptographic key that is unique for that individual chip and is used as its root key.

The same principle can be used for protection against tamper attacks for the electronics of the OBU without using a battery. For example, if we were able to measure the capacitance between traces in the wire-mesh used for protection, these capacitances will be unique in each device due to the manufacturing process and could be used to extract a cryptographic key. If the device is modified in any way, even if the wire-mesh is bypassed, the inherent capacitances will be different, obtaining a different cryptographic key and then detecting a tamper attack.

From software level, we could suggest extending the chain of trust to rootfs for future products related.

Pillar 3 - Electromobility (GFX/SID/CLS/TSYS).

Within Pillar 3, CARAMEL developed cyber threat detection techniques for plug-in Electrical Vehicles (EVs). The focus was on detecting irregularities in the communications to and from the charging infrastructure. The smart charging infrastructure cybersecurity analysis carried out by CARAMEL indicated several potential attack vectors due to its complex nature and the interactions between the entities involved creating a communication scheme susceptible to a number of security threats on different levels. Some examples of the communications occurring at a smart charging architecture include the metering and payment for energy, communication interruptions between the EV battery management system and the charge point that is followed by a communication mechanism between the CP and a central management system, and finally the establishment of a communication channel

between the CS and the energy suppliers (Distribution System Operators (DSO), Transmission System Operators (TSO), smart grids, etc.).

Smart Charging attacks can have a direct or indirect impact on every component of the energy ecosystem (charging infrastructure, TSO, DSO, etc.). The reliability and security of the whole energy supply network can be jeopardized due to the lack of security mechanisms in the charging stations for identifying and preventing potential attacks and threats. For example, a potential attack vector in the smart charging infrastructure could affect the DSO and as a result a partial energy blackout could take place. As Charging Point Operators (CPOs) and e-Mobility Service Providers (eMSP) struggle to integrate Artificial Intelligence (AI) approaches to modernize their services, the interconnection with different actors using different technologies opens doors to threats and vulnerabilities.

Within CARAMEL, a Machine Learning (ML) pipeline has been developed, deployed, and tested on a real dataset of a standard EV charging enterprise identifying abnormal activity in the charging process. The security concerns that have been raised by this pipeline can alert software developers, security administrators, and electrical engineers for potential threats to the smart charging infrastructure. An aspect that should be explored in the future is the interconnection between different actors in real-time informing for breaches, known vulnerabilities and zero-day attacks. An extension of this exchange of information is the creation of a common repository that stores, updates, and ranks known threats and attacks. Another extension of the proposed ML pipeline could be the exploitation of the Open Charge Point Protocol (OCPP) packets creating a library of signature-based attacks inside the private Azure vNET that the GFX socket server resides capable of blocking any malicious packets. Moreover, through the analysis of the OCPP packets, additional headers could be used providing a more detailed overview of the incoming transactions.

.

7 Conclusions

Next, we summarize the key findings and takeaways obtained from all the tests and developments done in the context of Pillar 1, 2, and 3 of the CARAMEL's project:

- The detection and mitigation of the cyber-attacks were quite efficient in restoring the attack in numerous Autonomous Driving Functions including fully automated Parking.
- The range of operation of the remote control needs to be extended.
- Regarding the in-vehicle location spoofing attack detection solution, it is important to understand and consider the anti-spoofing mechanisms already present in commercial GNSS receivers that are integrated into autonomous vehicles (e.g., timestamp checks to detect time synchronization attacks). The proposed solution should be enhanced to go beyond the existing mechanisms for detecting and importantly mitigating (see the in-vehicle location spoofing solution roadmap) fine-grain attacks that introduce small bias in the GPS locations to avoid detection.
- With regards to cyber-attacks on camera sensors it is important to understand the extent at which such attacks can be recovered from a single sensor and when additional sensors need to be considered. With regards to the embedded platforms used there still needs more work on supporting multiple deep learning models and facilitating their parallel execution in real-time.
- Regarding the anti-hacking device for the environmental attacks, different devices were considered and tested. The Jetson Nano AGX (anti-hacking device) performed at the best frame rate allowing real time processing during the integration. Of course, it consumed considerably more voltage but this can be mitigated by using the built-in ethernet and connecting via a wired local area network.
- Both real and synthetic datasets were used for the deep learning models during the training and validation. Performing transfer learning and fine-tuning significant benefits in terms of performance can be achieved using both modalities. The real data overcomes the lack of realism in the synthetic, and the synthetic ones offer more balanced/unbiased datasets including scenarios that are not common in real life under a controlled and safe environment.
- The overall performance of the traffic sign anomaly detection system can be improved significantly by introducing a reconstruction component. This deep network aims to improve the quality of the attacked traffic sign before the recognition and classification offering significant increase in accuracy and precision both for real and synthetic datasets.
- One of the main objectives achieved in pillar 2, it has been to develop a functional, flexible and easy to extend ETSI ITS protocol stack. We departed from the Vanetza open-source framework, and we modularized it in order to be able to commercially exploit it. We also contributed to the open community providing the new version of the ETSI security that was not implemented in the original framework. This protocol stack has been successfully tested in different platforms: i) standalone computers, ii) Single Board Computers that act as RSUs or OBUs, iii) in virtual containers as part of a MEC architecture, and iv) embedded in an automotive grade OBU. While in the three first cases, the deployment was easy, in the fourth case (the automotive grade OBU) the task has been very arduous and much time consuming. The reasons for this situation were that the Vanetza framework uses many C++ libraries and dependencies, and was not designed for optimizing memory space and, on the other side, the automotive grade OBU was designed to be economically competitive, and its memory and CPU capabilities were too low. Therefore, we had to prune many of the Vanetza superfluous dependencies and deploy some functionalities in the anti-hacking device. In a future version, we recommend not to use a low capabilities OBU plus an additional device, but to design a more powerful OBU and optimize the protocol stack.
- Related with the previous point, pillar 2 achieved the objective of a complete integration of the ETSI ITS protocol stack with the management of the required Authorization Tickets (AT) provided by a PKI architecture. The main goals have been: i) the management of ATs inside the OBU's HSM, to produce public/private key pairs and signature of messages, ii) the attachment of digital signatures plus digital certificates in the V2X messages, and iii) the selection of the time when the AT must be changed. Emphasizing this third point, one of the

known flaws of CAM messages broadcasting is a thread in the vehicle's privacy. Meaning that any eavesdropper can follow the trajectory of a vehicle. To try to mitigate this situation, the ETSI proposes to change the used AT and vehicle's identifiers from time to time. Pillar 2 has investigated this aspect and reached the conclusion that, even without analyzing the used AT, the fact of transmitting the vehicle's position in periods of 100ms, is enough to be able to track a vehicle. Therefore, there is no point of changing the AT. Nevertheless, the computational power to perform this operation is a critical aspect to consider. In consequence, pillar 2 proposed an algorithm to change the vehicle's AT considering when the vehicle has more chances to be "lost" among other vehicles.

- Another objective achieved in pillar 2, it has been bringing out the problematic of having different ITS radio technologies and developing a solution to solve it. Currently, we are in a phase that there is a mature and wide available technology (IEEE 802.11p), an already consolidated technology but not available as commercially Network Interface Cards (NIC) to be connected to any computer (LTE-PC5), and a new technology, not yet commercially available (NR-V2X). A part of these native V2X radio technologies, it is also possible to transmit ITS messages over IP over a public cellular network. Therefore, in the first stages of ITS services deployment, we need a solution as the one provided in pillar 2, that sets up fixed radio infrastructure of all required technologies, and using a forwarding software, this time executed in a MEC architecture, forwards messages to all involved vehicles. In CARAMEL, we successfully deployed an 802.11p RSUs network, with the protocol stack controlling them virtualized in a Kubernetes MEC. Additionally, we also set up a complete LTE network with the Open5GS open-source EPC and Accelleran small cells. With this configuration, the forwarding module is able to solve the interoperability problem and, in a near future, it can be extended to consider LTE-PC5 and NR-V2X technologies. The main problems that we faced building this architecture have been, firstly, the complexity of IP network addressing with Virtual Local Area Networks (VLANs), successive Network Address Translators (NATs) managed with different iptables routes and, secondly, the deployment of the Open5GS with the Accelerant's dRAX and small cells, that presented numerous instability problems.
- A further achievement of pillar 2, it has been the development of an architecture to deliver alarms detected in the vehicle (tampering alarm, GPS spoofing attack, etc.) to servers in the edge or in the cloud that can trigger mitigation actions or simply, monitoring the heartbeat of the whole system.
- Related to the HW, the release G2 of i.MX8 processor used was not able to detect the HW tamper attack when the device is powered off. The release G3 has implemented tamper monitoring in low power mode, that means powered by battery, which is able to detect tamper attacks when the device is powered off.
- The overall performance of the anomaly detection system that has been developed within Pillar 3 of the CARAMEL project with the aim to improve the resilience of the EV charging stations against smart charging abuse and EV scheduling abuse attacks can be enhanced through dimensionality reduction. Applying techniques such as Pearson correlation and Random Forest can significantly reduce the complexity of the incorporated ML models and avoid overfitting, improving all four-evaluation metrics (namely accuracy, precision, recall and F measure) of the devised anomaly detection system.

References

[1] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, 'CARLA: An Open Urban Driving Simulator', *arXiv:1711.03938 [cs]*, Nov. 2017, Accessed: Nov. 16, 2020. [Online]. Available: http://arxiv.org/abs/1711.03938

[2] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, 'MobileNetV2: Inverted Residuals and Linear Bottlenecks', 2018, pp. 4510–4520. Accessed: Nov. 17, 2020. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html

[3] Andreas Geiger, Philip Lenz and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[4] Jingwen He, Chao Dong, and Yu Qiao. Modulating image restoration with continual levels via adaptive feature modification layers. CoRR, abs/1904.08118, 2019.

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017.

[6] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud, 2019.

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017

[8] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network, 2017.

[9] Anurag Arnab, Ondrej Miksik, and Philip H. S. Torr. On the robustness of semantic segmentation models to adversarial attacks, 2018.

[10] N. Piperigkos, A. S. Lalos and K. Berberidis, "Graph based Cooperative Localization for Connected and Semi-Autonomous Vehicles," in 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2020.

[11] C. Vitale and et al., "The CARAMEL Project: a Secure Architecture for Connected and Autonomous Vehicles," in 2020 European Conference on Networks and Communications (EuCNC), 2020.

[12] C. Vitale, N. Piperigkos, C. Laoudias and et al., "CARAMEL: results on a secure architecture for connected and autonomous vehicles detecting GPS spoofing attacks," EURASIP Journal on Wireless Communications and Networking, vol. 2021, 5 2021.

[13] S. Bai, J. Z. Kotler, V. Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. (19.4.2018). Available at: https://arxiv.org/pdf/1803.01271.pdf [Accessed: 10.3.2021]

[14] L. Zhang, V. Kekatos and G. B. Giannakis, "Scalable Electric Vehicle Charging Protocols," in IEEE Transactions on Power Systems, vol. 32, no. 2, pp. 1451-1462, March 2017.

[15] L. Gan, U. Topcu and S. H. Low,"Optimal decentralized protocol for electric vehicle charging," in IEEE Transactions on Power Systems, vol. 28, no. 2, pp. 940-951, May 2013.

[16] J. Rivera, P. Wolfrum, S. Hirche, C. Goebel and H. Jacobsen,"Alternating Direction Method of Multipliers for decentralized electric vehicle charging control," 52nd IEEE Conference on Decision and Control, Florence, 2013, pp. 6960-6965.

[17] J. Rivera, C. Goebel and H. Jacobsen,"Distributed Convex Optimization for Electric Vehicle Aggregators," in IEEE Transactions on Smart Grid, vol. 8, no. 4, pp. 1852-1863, July 2017.

[18] L. Liu, M. Esmalifalak, Q. Ding, V. A. Emesih and Z. Han,"Detecting FDI Attacks on Power Grid by Sparse Optimization," in IEEE Transactions on Smart Grid, vol. 5, no. 2, pp. 612-621, March 2014.