

CVS: Design, Implementation, Validation and Implications of a Real-world V2I Prototype Testbed

Alessandro Marchetto*, Panagiotis Pantazopoulos[‡], András Varádi[§], Silvia Capato[¶] and Angelos Amditis[‡]

* Centro Ricerche Fiat, Trento, Italy. Email: alessandro.marchetto@crf.it

[‡]Institute of Communication and Computer Systems, Athens, Greece. Email: {ppantaz, a.amditis}@iccs.gr

[§]Commsignia Ltd, Budapest, Hungary. Email: andras.varadi@commsignia.com

[¶] Swarco Mizar s.r.l., Torino, Italy. Email: silvia.capato@swarco.com

Abstract—A Connected Vehicle System (CVS) is a cyber-physical system of highly-equipped infrastructure-connected vehicles interconnected with road-side units and cloud-based services to offer safer driving. Despite the ever increasing relevant research, the testing of CVS functionalities and communication features remains problematic; computer-based simulation requires detailed system models while field-testing is expensive, focusing on few components and may raise safety concerns.

In this paper, we present a full CVS prototype testbed enabled to realize a broad set of timely Vehicle-to-Infrastructure (V2I) use-cases and serve as the basis for automotive cyber-security testing. The testbed designed and built in the context of the H2020 SAFERtec project, is described in terms of its hardware requirements, software design and implementation. The conducted testing activities on its capability to realize the considered V2I use-cases and support cybersecurity testing is explained. More importantly, the paper details ‘take-home’ lessons derived from the CVS implementation experiences aiming to assist future development of automotive prototype testbeds.

I. INTRODUCTION

Connected vehicles, regardless of their automation level, leverage the latest achievements of in-vehicle hardware and vehicle to everything (V2X) communications to achieve safer driving conditions and improved comfort [1]. When the vehicle is considered closely-coupled with the (wireless or mobile) interfaces to infrastructure and the exchanged data, the emerging *Connected Vehicle System* (CVS) constitutes the heart of the upcoming automotive landscape. The successful adoption of such technology is mainly dependent on the offered reliability. In that sense, testing and validation of the CVS’s performance is of paramount importance.

However, testing the connected vehicles functionality is subject to limitations [2]; standardized testing approaches are lacking while field operational tests in real traffic conditions come with high cost [3] and considerable safety challenges. As such, a promising alternative for testing and validation of the vehicular technology is to resort to laboratory experimentation using prototype testbeds. One particular aspect of the mandatory technology validation involves the cybersecurity of vehicle to infrastructure (V2I) communications and applications [4] which is the rationale for building the studied testbed.

Related work on compiling automotive testbeds is limited. In [5], a prototype testbed of short range communications uses IEEE 802.11p and Ethernet interfaces to realize a simple scenario of packets broadcasting. The aim is to measure

the maximum communication range (between transmitter and receiver) as well as the corresponding delay.

When the focus is on cybersecurity testing, testbed works span across three axes: (a) those describing architectural aspects, such as [6] which overviews the state-of-the-art about automotive security mechanisms and it envisages how the experience of IT security can be migrated to the automotive domain; (b) those driven by modeling, such as [7] which first analyzes automotive cybersecurity threats and then evaluates them on automotive components by considering theoretical analyses and functional testing; (c) those focusing on certain in-vehicle modules such as [8] and [9]. The former paper presents a testbed for security automotive research and training. It replicates the interactions between hardware components and control software of an automotive system supporting fuzzy and replay tests on ECUs [10]. The latter paper presents a testbed built by an ECU and a CAN [11] network simulator to study cyber-attacks and countermeasures and it has been used to evaluate security threats on a vehicle.

In contrast to these works, we present a *full* V2I testbed realizing *all* involved V2I parts. Our contribution amounts to the design, implementation and testing of a real-world testbed capable of reproducing a broad set of automotive use-cases. The required hardware/software to realize both the vehicle part (*i.e.*, on-board units) and the infrastructure part (*i.e.*, road-side unit and cloud services) is detailed while standardized protocols implement the corresponding software for wireless and mobile communications. We finally highlight the gained experience from the testbed compilation, made for security evaluations [12] in the SAFERtec project [13]; even if so, the presented testbed is able to serve any testing purpose. By sharing our experiences in this paper we aim at filling the literature gap on the development of *full* V2I testbeds which increase the engineering quality of automotive products.

In what follows, Section II discusses the CVS design-aspects. Section III describes the involved modules and Section IV details the supported use-cases. Section V highlights the CVS testing activities, while Section VI gathers interesting implementation experiences. Section VII concludes the paper.

II. TESTBED ARCHITECTURAL DESIGN

Figure 1 shows the main CVS actors: vehicle, road-side unit, cloud services; and their interconnections. The vehicle is

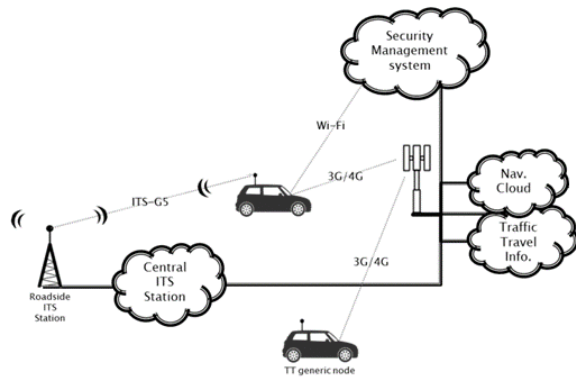


Fig. 1. Vehicle connected to roadside stations and cloud services

a mobile ITS station equipped with communication technologies: (i) ETSI ITS-G5, short-range (up to 1km) V2I wireless connectivity based on the IEEE 802.11p ETSI ITS-G5 [14]; (ii) 3G/4G/LTE cellular, the long-range mobile connectivity for the vehicle-to-cloud (V2C) link; and (iii) in-vehicle Ethernet/CAN/Wi-Fi, mainly for board-to-device link. The Roadside ITS Station (R-ITS-S) is fixed, installed close to the road and connected to the vehicle via ETSI ITS-G5 and to the cloud via wired connectivity. Cloud services provide enhanced functionality such as real-time traffic information, road-events notification, traffic management and user authentication.

The CVS is a dynamic system based on highly interconnected actors composed of third-party hardware and software modules. The high inter-connectivity enables the deployment of cooperative, lively updated and safety-related services that use information shared among vehicles and infrastructure. Security mechanisms of such an ecosystem mainly relate to the adopted communication technology.

- ETSI ITS-G5: message pseudonymity, integrity and authorization are guaranteed thanks to certificates and signature on the messages authorized by the Security Credential Management System that emits certificates as trusted authority (standard ETSI EN 103 097 121/131).
- 3G/4G/LTE cellular: conventional protection mechanisms such as the ones used in Internet are adopted for user authentication and protection of the cloud-service access; *e.g.*, (remote) authentication mechanisms (OAuth), network firewalls and encryption (HTTP over TLS/SSL).
- In-vehicle network: protection mechanisms, *e.g.*, Wi-Fi Protected Access II-WPA2, Wi-Fi/Ethernet/CAN network proxies and firewalls, authentication and access protection mechanisms, as well as encrypted communication protocols (HTTPS), are adopted.

III. HARDWARE AND SOFTWARE MODULES

Figure 2 shows the CVS logical architecture with the internal modules of each actor and relevant data flows, as well as the physical implementation (right-bottom corner).

A. The Vehicle Platform

The vehicle architecture is based on a Controller Area Network (CAN, ISO 11898-1/2) crossed by vehicle signals, such as speed, brake, and vehicle body signals. By means of a CAN Gateway, the network is separated into two segments: one hosts boards and devices having only in-vehicle connectivity (private); and the other one connects boards and devices having external connectivity. The CAN Gateway, hence, acts as network proxy and firewall.

The application on-board unit (APP OBU) runs applications that realize the considered use-cases. It is an embedded PC based on a Linux OS implementing security mechanisms as recommended by the Center for Internet Security¹. The Linux OS hosts a Docker² platform that executes third-party software modules. Docker is a virtualization platform allowing the isolation of software functionality from its surrounding *e.g.*, hosting OS. In the CVS, Docker separates different modules (images in the Docker terminology) that: (i) implement the use-cases logic (*e.g.*, triggering notifications towards the user-interface according to road events); (ii) provide API interfaces for both user authentication (*e.g.*, through OAuth) and user-interface; (iii) implement data fusion among different communication channels (ETSI ITS-G5 and cellular); and (iv) connect to cloud services via HTTPS.

The V2X on-board unit (V2X OBU) manages V2X messages according to the ETSI ITS-G5 standard. It, hence, de-/encapsulates messages and exposes their payload via APIs. According to the standards, the V2X OBU signs/verifies messages and manages vehicle security certificates, keys, rights to services and trust lists. Dedicated hardware such as V2X radio and a tamper-protected Hardware Security Module (HSM) is used. Finally, V2X OBU acquires CAN data (*e.g.*, speed) from the vehicle and the GNSS signal (*i.e.*, vehicle position).

The CVS has been implemented as a bench to work-on in a laboratory for testing purposes. To this aim, both vehicle CAN data and GNSS signal (both for vehicle and R-ITS-S) are provided by replaying previously recorded traces collected by running the use-cases, *i.e.*, driving the car in the road according to pre-defined scenarios. The CVS, hence, can reproduce continuously the interested use-cases, switch them on-the-fly, and notably, use real data. To this end: (i) a CAN replicator based on SocketCAN³ runs on the APP OBU and is connected to the in-vehicle network; and (ii) two GNSS signal replicators, based on Software Defined Radio for the vehicle and on GPSFAKE⁴ for the R-ITS-S, have been implemented to replay the GNSS signal recorded on the road.

The Network Gateway connects the vehicle components through a private Ethernet network and provides the mobile Internet access. Such a gateway protects the private network from unauthorized external access by acting as a firewall and checking the incoming traffic data. Finally, the vehicle user-

¹<https://www.cisecurity.org>

²<https://www.docker.com>

³<https://www.kernel.org/doc/Documentation/networking/can.txt>

⁴<https://gpsd.gitlab.io/gpsd/gpsfake.html>

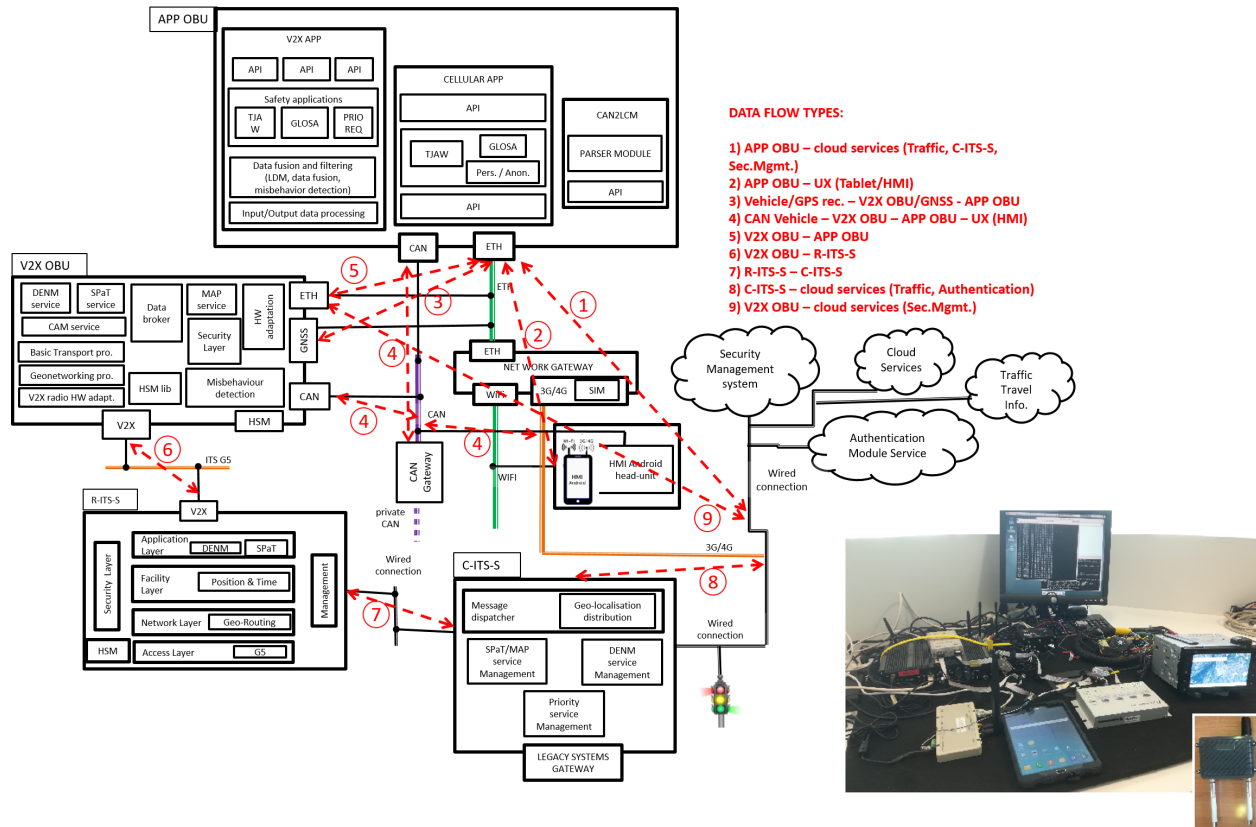


Fig. 2. CVS logical architecture and physical implementation (right-bottom corner)

interface consists of a tablet connected to the in-vehicle Wi-Fi provided by the Network Gateway and an after-marked vehicle user-interface. Both (Android-based) devices receive commands and data from the Docker images that implement the use-cases logic.

B. The Road-side ITS Station

The R-ITS-S is a gateway between the vehicle and the services provided by the Central ITS station (C-ITS-S). The R-ITS-S receives, transforms and sends data to/from vehicle and C-ITS-S. The R-ITS-S has to be installed in the vehicles' proximity, *i.e.*, the area covered by the IEEE 802.11p (ETSI ITS-G5) signal that the R-ITS-S sends-out.

The R-ITS-S implements the ETSI ITS-G5 stack, all ITS layers according to ETSI EN 302 665 as well as the security manager that signs and verifies messages following the ETSI TS 103 097. It, hence, encodes V2X messages (*e.g.*, SPaT and DENM about traffic events) to be sent to vehicles and decodes V2X messages (*e.g.*, CAM with speed and position) received from vehicles. Then, a private IP network is used to transfer and receive notifications and data to/from C-ITS-S.

C. The Cloud Services

The C-ITS-S is wiredly connected to: (i) other services, such as traffic management center (TMC), traffic light controllers (TLC), and traffic information providers; and (ii) R-ITS-S, thus reaching vehicles. The C-ITS-S is also directly connected,

via cellular network, to vehicles and user devices. C-ITS-S collects information about traffic signals, traffic events and the vehicle's position/dynamics and it is responsible for the accurate provision and delivery of relevant information, *e.g.*, traffic notification. C-ITS-S is deployed as a virtual server composed of three main services: (i) a TLA manager to generate V2X-compliant SPaT messages, based on TMC/TLC signal feedback, and MAP messages for cloud-based services; (ii) a priority manager to receive vehicles' priority requests and identify the authorized ones; and (iii) a DENM manager to generate V2X-compliant DENM messages, based on the raised traffic events.

The traffic-info service provides information about real-time traffic events for a given geographic area, *e.g.*, traffic jam information or presence of road-works. A real-world version of such a service is used rather than a testing mock-up version.

The Authentication module service manages user sensitive information, generates authentication tokens, and controls the user access to other services.

The Security Management System is a service (off-line) that manages the V2X Public Key Infrastructure (PKI) system, it provides certificates used by ITS stations to secure messaging. When security is enabled, V2X messages are signed with a key and a certificate is added to the message before sending it. Both vehicle and R-ITS-S store certificates that are used for secure messaging. They are downloaded in advance from the

PKI periodically (*i.e.*, a set of certificates valid for a week). The CVS relies on such certificates which are valid during the performed test (validity includes time, location and service specific permissions) and allow for a common trust between different entities (R-ITS-S verifies the vehicle messages and vice-versa). Common trust is achieved by being able to trace back each certificate to a group of trusted authorities (a common RootCA). While PKI systems are available for registered entities in Europe, certificates issued by them are subject to strict limitations and thus, they cannot be used for our purpose. A local service instead lets us generate customized certificates that are valid only for our testing purposes.

IV. THE CONSIDERED USE CASES

Out of a broad set of automotive use-cases that the testbed supports, we have selected to reproduce a limited yet challenging subset on the basis of their safe-criticality, maturity, and usefulness. Having them realized using dedicated hardware and a number of relevant applications, our aim is to create a realistic environment to validate the SAFERtec security assurance framework [12]. Towards that end, we need to have a fully functional testbed realizing the selected use-cases. Those involve the vehicle's communication with the road-side station and/or cloud services. An effort to identify use-cases that lend themselves to both communication types has been taken and initially almost 30 use-cases were identified (by considering Day 1/1.5 applications in line with the European Commission C-ITS platform). Without harming the generality of the testbed capabilities, we focused on the following use-cases:

- *ETSI ITS-G5*: Information originates from the infrastructure back-end (*e.g.*, TMC), it reaches the road-side station and becomes relevant in the use-cases of optimal driving speed advice (GLOSA), provision of real-time traffic information (*e.g.*, traffic notifications) and priority request in intersection crossing (the vehicle sends information to the infrastructure).
- *Cellular*: Information originates from cloud services and, through the cellular network, it becomes relevant for use cases like: GLOSA and traffic event provisioning. Moreover, personalized driving-advice from the cloud can help us study how to secure personal data.
- *ETSI ITS-G5 and Cellular*: Information originates from both the infrastructure back-end to reach the road-side station and from the cloud service, then it becomes relevant in the use-case of the provision of real-time traffic information.

V. CVS INTEGRATION AND TESTING

The CVS has been subject to a three-level testing, aiming at verifying the implementation accuracy, *i.e.*, the capability of reproducing the use-cases and respecting their (implementation and security) requirements. Unit Tests have been conducted on all CVS components for checking their main behavior and features. Integration Tests have been conducted by exercising all main component connections, thus checking the component interfaces and data-flows. Use-Case Tests have been conducted by replaying all main use-case scenarios, thus checking the CVS capability to realize the selected use-cases.

At each testing level, test cases have been defined by analyzing the requirements of the element under test (*i.e.*, component, component connection, and use-case), for identifying relevant execution flows, *i.e.*, scenarios. Then, each execution scenario has been concertized in test cases that exercise the CVS implementation, thus allowing us to verify it. All tests have been specified by providing: [*before test execution*] the Test Objective (*i.e.*, what we intend to test), the Test Object (*i.e.*, component, interface, behavior that we test), the Test Scenario (*i.e.*, how the test object is executed) and Execution Steps (*i.e.*, actions to run the test), Input Data and Execution Conditions (if any), Expected Output. [*after test execution*] the Test Output (*i.e.*, the output obtained when test is executed), and the Result (*i.e.*, "Passed" if the execution has led to the expected output, "Failed" otherwise). Figure 3 shows fragments of a test for the (ETSI ITS-G5) GLOSA use-case. We expect that by passing all test cases, the CVS demonstrate a reasonable capability of implementing the use-cases and their requirements.

Overall, we defined and executed 44 test cases for the Unit Tests, 21 for Integration Tests, and 15 for Use-Case Tests. Each test was executed multiple times in several iterations; particularly, those related to V2X have been executed with and without the V2X security enabled. Unit tests have been executed by component suppliers in their laboratory. Integration and Use-Case Tests have been executed in physical integration meetings by the involved component suppliers. During the test execution, information such as execution logs, user-interface screenshots as well as execution traces have been collected and post-analyzed to discover and fix issues. For instance, Figure 3 shows a fragment of test-plan and execution results collected for the ETSI ITS-G5 GLOSA use-case (Use-Case Test). The test specifications and test-execution information is shown: the top part shows the content of V2X messages (MAP/SPaT) sent by the R-ITS-S and collected on-the-air by a debug tool; the bottom part shows a screenshot of the user-interface application that warns the driver about traffic-light phases and provides speed advice.

VI. LESSONS LEARNED FOR VEHICULAR TECHNOLOGY

During the CVS implementation, we observed some incompatibility problems in the exchange of V2X messages (SPaT and MAP), due to the use of different V2X standard versions by R-ITS-S and the vehicle boards. The initial result was that SPaT and MAP messages sent by the R-ITS-S were discarded by the vehicle. The use of different (potentially old) V2X standard versions prevented the system from working as expected and exposed it to known vulnerabilities. As such, an alignment in the version of standards was needed for functional interoperability.

We encountered issues in testing the provision of real-time traffic information whereby a vehicle receives traffic event notifications via both ETSI ITS-G5 and cellular network. In case of concurrent notifications, a decision has to be made in the vehicle to properly provide the information to the driver avoiding wrong warnings. We showed all raised notifications

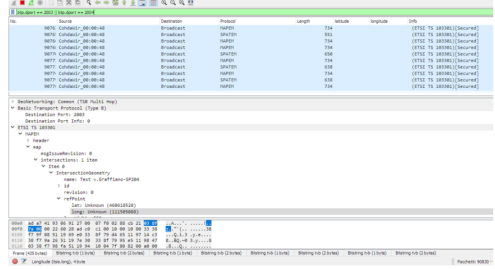
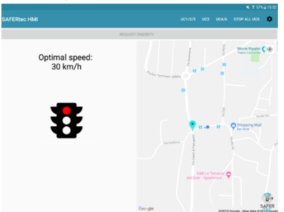
PHASE	TEST DETAILS
Objective	Test GLOSA information sent and received
Object	UC1: Optimal driving speed advice (V2X and cellular network) Test the flow in which a TL phase message is received by the vehicle
Input	*) A vehicle trace (vehicle GNSS and CAN data) recorded in Trento (IT) *) TL phases provided to the by C-ITS to the vehicle via R-ITS-S
Execution Conditions	*) Secured V2X communication *) TL phase are synchronized with a specific GPS position of vehicle
Execution Steps	BASIC FLOW_UC1_B1 1) A vehicle is approaching an intersection 2) The vehicle sends continuously CAM messages to the R-ITS-S 3) R-ITS-S forwards data about the vehicle to the C-ITS-S 4) C-ITS-S receives data from different sources: R-ITS, TMC, TLC 5) C-ITS-S composes and sends-SPaT messages to the R-ITS-S 6) R-ITS-S sends MAP and <u>SPaT</u> messages to the vehicle 7) The vehicle calculates the speed to approach the intersection in a safe and effective way 8) The vehicle indicates, via HMI, the calculated speed
Expected Output	When vehicle approaches the intersection, it receives the TL phases and it warns the driver through the user-interface
Achieved Output	*) Secured <u>SPaT</u> and MAP messages on the air, captured by a debug tool  *) HMI: Example of warning to the driver 
Result	Passed

Fig. 3. Example of testing specs and results for the ITS-G5 GLOSA use-case

but a more in-depth investigation (on this decision-making problem), lying outside of our scope, could be helpful.

While most of the implemented CVS cloud services were test/mock-up versions, we also used real services providing live notifications about traffic events in specific geographic areas. Real services required extra-effort but made the CVS realistic, allowing for more effective testing. For instance, geographic areas covered by such services typically correspond to big urban areas of potentially high-traffic; thus, we had to consider different testing sites (*i.e.*, three Italian cities), instead of one, for testing the CVS under diverse traffic profiles.

We also encountered issues in the use of the GNSS signal. Not always the testing laboratory were exposed to the GNSS signal, thus hampering the correct provision of V2X services. Furthermore, since we considered different test sites, the geographic area of tests (*e.g.*, Turin, IT) often was different from the real geographical location of the laboratory (*e.g.*, Trento, IT). To practically address this issue, we implemented two GNSS replayers for vehicle and R-ITS-S. The CVS, hence, enables the use/test of different configuration set-ups (*e.g.*, locations and GNSS signal power).

Finally, the enabling of V2X security (*e.g.*, ETSI TS 103-097) on CAM messages having ad-hoc field contents (*i.e.*, priority-request use-case) allowed us to implement/test out-of-standard services and features (*e.g.*, potentially part of future standard) but it was also a source of problems. The Service Specific Permissions configured for the V2X certificates generated with the local PKI tool did not allowed the CVS 'vehicle' to use privileged services. We, hence, initially failed in running test cases related to the priority-request use-case, with the V2X security enabled. Finally, the implemented CVS allows to enable/disable of the V2X security, thus evaluating the effectiveness and impact of V2X security features.

VII. CONCLUSIONS

In view of the limited literature on real-world testbeds in the V2I automotive scene, we have designed, implemented and tested a full V2I testbed. Despite the fact that our targeted usage is on security assurance evaluation, the testbed operation remains of generic purpose; it is capable of realizing any V2I communication instance and support exhaustive testing. While our SAFERtec work focuses on testing and validating the proposed assurance framework, the work on the CVS design and implementation is complemented with the provision of guidelines for assisting future experiments with V2I testbeds.

ACKNOWLEDGMENTS

This work is part of the SAFERtec project which is funded by the EU H2020 programme under grant agreement No 732319.

REFERENCES

- [1] N. Lu *et al.*, "Connected vehicles: Solutions and challenges", IEEE Internet of Things Journal, vol. 1, pp. 289–299, 2014
- [2] A. Tierno *et al.*, "Open issues for the automotive software testing", IEEE Conference on Industry Applications, Curitiba, pp. 1-8, 2016
- [3] Y. Barnard *et al.*, "Methodology for Field Operational Tests of Automated Vehicles", Transportation Research Procedia, Vol. 14, 2016
- [4] A.D. Kumar, K.N.R. Chebrolu, and V.R. Soman, "A Brief Survey on Autonomous Vehicle Possible Attacks, Exploits and Vulnerabilities", ArXiv, 2018
- [5] X. Duan, Y. Yang, D. Tian, Y. Wang, T. Li, "A V2X communication system and its performance evaluation test bed", IEEE International Symposium on Wireless Vehicular Communications, Canada, 2014
- [6] M. Ring, D. Frkat, M. Schmiedecker, "Cybersecurity Evaluation of Automotive E/E Architectures", ACM Computer Science in Cars Symposium, Germany, 2018
- [7] S. Bayer, T. Enderle, D.K. Oka, and M. Wolf, "Automotive Security Testing—The Digital Crash Test", Energy Consumption and Autonomous Driving. Lecture Notes in Mobility, Springer, 2015
- [8] P.N. Borzjani, C.E. Everett, and D. McCoy, "OCTANE: An Extensible Open Source Car Security Testbed", Embedded Security In Cars, USA, 2014
- [9] D.S. Fowler, M. Cheah, S.A. Shaikh and J. Bryans, "Towards a Testbed for Automotive Cybersecurity", IEEE Conference on Software Testing, Verification and Validation, Japan, 2017
- [10] M. Ahmed *et al.*, "Intra-vehicular wireless networks," IEEE Globecom Workshops, USA, pp. 1–9, 2007
- [11] CAN bus <https://www.kvaser.com/can-protocol-tutorial/>
- [12] P. Pantazopoulos *et al.*, "Towards a Security Assurance Framework for Connected Vehicles", IEEE WoWMoM Workshop on Smart Vehicles, Greece, pp. 01-06, 2018
- [13] SAFERtec <https://www.safertec-project.eu/> (2020)
- [14] European Telecommunications Standards Institute (ETSI) <http://www.etsi.org> (2020)