Improved Security Performance for VANET Simulations

Raphael Riebl [*] **Markus Monz** [**] **Simon Varga** [**]
Leandros Maglaras [***] **Helge Janicke** [***] **Ali H. Al-Bayatti** [***]
Christian Facchi [*]

[*] *Technische Hochschule Ingolstadt, Research Centre, Germany (e-mail: raphael.riebl@thi.de)*
[**] *Technische Hochschule Ingolstadt, Faculty of Electrical Engineering and Computer Science, Germany*
[***] *De Montfort University, Software Technology Research Laboratory, Leicester, United Kingdom*

Abstract: Upcoming deployments of Vehicular Ad Hoc Networks (VANETs) in Europe are expected to sign and verify packets secured by cryptographic signatures by default. Thus, when VANET simulations are used for development and test of applications building upon vehicular communication, the overhead induced by security extensions to the ITS-G5 protocol stack shall not be neglected. This paper presents a standard compliant simulation model capable to handle secured messages. Beside its suitability for Hardware-in-the-Loop simulations employing secured communication, the model's major advantage is the minimisation of the simulation environment's performance penalty linked with cryptographic computations.

Keywords: Computer networks, Communication protocols, Device testing, Security

1. INTRODUCTION

As cars become more interconnected, one of the main challenges that manufacturers have to face is security. Especially for safety applications it is essential to ensure that life-critical information cannot be modified or dropped by an attacker's activity. Even informatory applications, which do not take control of the vehicle, are only useful when the driver is warned in time and not bothered by false warnings. Safe and reliable operation is even more of concern for Advanced Driver Assistance Systems (ADASs) intervening in actuators like steering or brakes. Hence, communication mechanisms shall constitute a safe environment for interconnected vehicles by establishing the three major pillars of security: Confidentiality, integrity and availability.

Future automated cars will rely on high quality inter-vehicle communication (IVC) that is incorporated with safety applications. With increasing levels of driving automation the reliable and far-sighted perception of surrounding traffic gains importance. A vehicle can perceive its environment either by mounted sensors or by receiving beacons sent by vehicles in the vicinity. These periodically emitted beacons may even convey additional information such as vehicle weights or planned manoeuvres, which are not measurable by sensors but can be propagated by IVC. In case of an emergency, event-driven messages can be generated and disseminated to endangered vehicles in the zone of relevance (ZOR). With safety concerns in mind, tackling of jamming threats is critical, especially in ZORs.

The area of security threats for VANETs has been widely investigated. Various attacks are possible, such as Denial of Service (DoS) attack, Fabrication, Alteration, Replay, Message Suppression, and Sybil attack (Douceur, 2002). One group of VANET attackers is formed by *selfish drivers*, who try to take advantage of received information for personal benefit. Apart from that, *malicious attackers* (Buttyán et al., 2007) aim for harming users or the network in general. In a black hole attack, a malicious node exploits the packet routing mechanism, advertising itself as providing the shortest path and attracting most of the traffic its way (Bibhu et al., 2012). When this route is established, the malicious node decides whether to drop all packets or forward them to an unknown address.

A substantial amount of research on defense mechanisms has been focused on intrusion detection systems (IDSs) for early detection of malicious nodes (Maglaras, 2015; Larson et al., 2008; Müter et al., 2010). In this regard, both specification-based (Larson et al., 2008) and anomaly-based treatments (Müter et al., 2010) have been investigated. Moreover, an attempt to deflect attacks using honeypots has been described in Verendel et al. (2008), while other novel techniques for filtering out tweaked data have been recently developed (Basaras et al., 2015).

Cryptography is one of the mechanisms that can solve a lot of VANET systems' security issues (Mejri et al., 2014). Modern cryptography offers several security features such as confidentiality, authentication, integrity, non-repudiation and secret sharing, which provide a secure environment for communications among vehicles (Pathan, 2010). Unfortunately, cryptography is linked with computational cost which can grow to an impediment, especially when all costs accumulate at one computer as it usually arises with VANET simulations.

2. MOTIVATION & CONTRIBUTIONS

Simulations are a great tool for examining VANETs since no special hardware is required and even complex scenarios can be investigated without the difficulties of conducting field tests involving an immense number of network nodes. For determining the feasibility of novel vehicle features such as cooperative manoeuvre planning, simulations are an adequate and inexpensive tool. At some point on the path towards system testing a real piece of hardware, e.g. an on-board unit (OBU), shall be coupled with the simulation environment forming an IVC testbed. In most cases it is impossible to slow down hardware execution so the simulation environment needs to run in real-time for this setup. The demand for fast execution times imposes a compromise between realism and computational complexity of the simulation model. Still, the model needs to be accurate enough, i.e. it has to serve standard compliant packets when a device under test (DUT) is connected to the testbed through its radio interface. The value of such test facilities incorporating real devices is underlined by repeatedly organised Intelligent Transportation System (ITS) plugtest events by the European Telecommunications and Standard Institute (ETSI).

Applications and systems based on IVC possess an intrinsic level of complexity due to the nature of its decentralised algorithms. Because of this complexity, subtle errors slip easily into simulation models or their parameterisation and deteriorate simulation results in the end. Model validation and elimination of faulty behaviour can be achieved by testing the simulation model against third-party implementations, such as OBUs for IVC. Hence, support for testbed execution can increase the trust in a simulation model, even when the model is used in a solely virtual environment. Depending on the testing objective, either the simulation model can be evaluated against a mature OBU or vice versa.

Testbeds for VANETs in a laboratory environment have been proposed before as by Vandenberghe et al. (2011). While their testbed architecture is composed of a multitude of devices, our approach relies more heavily on simulation technology which is described initially in Riebl and Facchi (2014). The focus of our architecture is the environment emulation for one single DUT. For this purpose, the DUT's communication partners are simulated and the simulation is coupled with the DUT over the radio interface, which adheres to the IEEE 802.11p standard (see Fig. 1). Within the simulation's virtual world the physical DUT is represented by a proxy vehicle acting on behalf of the DUT and passing information back and forth. Both, proxy and DUT, can be looked upon as counterparts.

Previously, our simulation framework *Artery*[1] (Riebl et al., 2015) was capable to take time delays and packet length overhead into account for simulations without involvement of a DUT. With this paper we enhance the simulation by secured messages as specified in ETSI (2015). This enables the communication between DUT and virtual environment via secured messages, i.e. the over-the-air packets are cryptographically signed and follow the standardised binary format. Contrary to other simulation

[1] `https://github.com/riebl/artery`

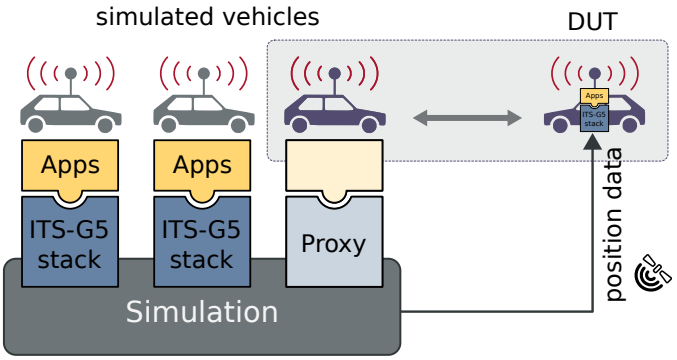

Fig. 1. Simulation and DUT testbed

frameworks as in DOT (2011, Appendix G) these extensions are not primarily intended for evaluating various security mechanisms. Instead, the presented testbed enables DUT tests with low hardware requirements (at the minimum a simulation computer, IEEE 802.11p capable network card and a DUT) based on standard compliant packets.

Contributions of this paper comprise in a nutshell:

- enhanced simulation model regarding secured messages compliant to standards (Section 3.1)
- efficient DUT integration into security-aware testbed through over-the-air packets (Section 3.2)
- evaluation in terms of runtime costs for various settings on two independent systems (Section 4)

The discussion of current limitations and ideas for further improvements concludes the paper (Section 5).

3. MODEL DESIGN

Conceptually, security is considered a cross-layer entity in the ETSI ITS reference architecture (ETSI, 2010), i.e. it is a collaborative effort with several involved communication layers. The lowest layer concerned about security in the ITS-G5 protocol stack is the GeoNetworking layer (ETSI, 2014a), just above the Medium Access Control (MAC) layer. The primary purpose of the GeoNetworking layer is the routing of packets from source to one or more destination nodes, possibly with help of forwarding nodes in between. The routing layer, however, is also capable to encapsulate the payload data from upper layers into so-called secured messages. As part of this encapsulation process all payload and some of parts of the GeoNetworking headers are cryptographically secured employing elliptic curve cryptography (ECC). Secured messages can employ encryption but for car safety applications confidentiality is not an issue because all vehicles should be able to interpret the disseminated data. Integrity and authenticity, however, are of importance so faulty or manipulated packets can be detected and in the end rejected at receiver side. Thus, the focus lies on cryptographic signatures in the following, employing the Elliptic Curve Digital Signature Algorithm (ECDSA) scheme.

3.1 Security extension

Analysis of the packet structure reveals that only data fields of the GeoNetworking layer are aware of security. Indeed, secured messages are transparent for layers above

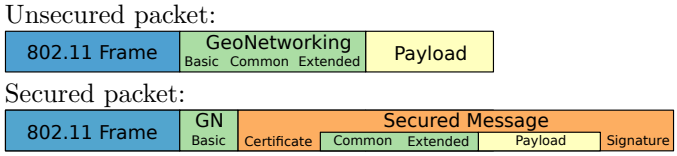

Fig. 2. Structure of unsecured and secured packets

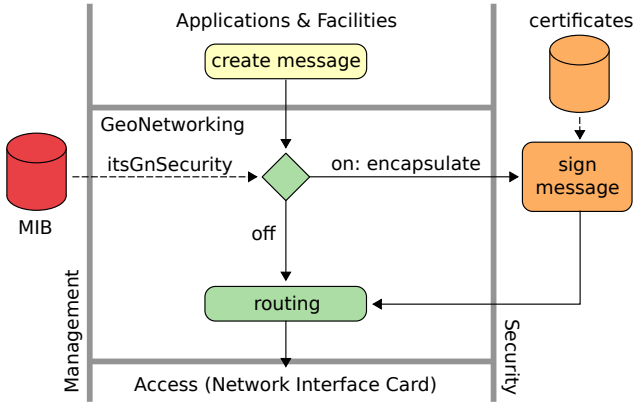

Fig. 3. Security related packet flow (outgoing)

the routing layer. VANET applications providing services such as Cooperative Awareness (CA) or Decentralized Environmental Notification (DEN) can influence the encapsulation of their messages by a control parameter called "security profile". Such a profile is an internal parameter and is not encoded in the transmitted packets at all but it is used to choose an appropriate certificate during the encapsulation and signing process. After decapsulation at the receiver side, the GeoNetworking layer passes the message along with control parameters to its upper layer. One of these control parameters reports the security level of the received packet, so the receiving application can then decide how much confidence it puts in this particular message.

Fortunately, the envisaged testbed for DUTs can build upon several already existing building blocks. On the one hand side, *Vanetza*[2] is an open-source implementation of the ITS-G5 protocols including GeoNetworking capabilities. On the other hand, the simulation framework *Artery* integrates *Vanetza* and combines it with virtual IEEE 802.11p compliant network interface cards. Each simulated vehicle within *Artery* can be equipped with a middleware, which models the Facilities and Application layer according to the ITS reference architecture (ETSI, 2010). This middleware is thus the simulation's runtime environment for VANET applications such as the CA and DEN service.

When an application generates a message as depicted in Figure 3, this message is yet unsecured. Just on arrival at the GeoNetworking layer security operations are applied to messages depending on the *itsGnSecurity* flag stored in the Management Information Base (MIB). If *itsGnSecurity* is enabled, plaintext messages are encapsulated in a security envelope and become secured messages as a result. For this purpose *Vanetza* has been extended to read and write the secured message format, i.e. raw bits and bytes can be converted to appropriate data structure and vice versa. Furthermore, this basic encoding and decoding feature is supported by the security entity, which handles these secured messages functionally. First and foremost the security entity is responsible for calculating signatures of outgoing packets and verify signatures of incoming packets. These calculations are by far the computationally most expensive steps executed in the GeoNetworking layer, especially compared to the actual routing.

[2] `https://github.com/riebl/vanetza`

3.2 Deferred cryptography

The idea for reducing the negative performance impact due to cryptographic signature calculations stems from the fact of having only a single DUT linked with many simulated vehicles in the proposed testbed. There is no need to calculate signatures for packets exchanged only between simulated vehicles since those vehicles are completely under control of the testbed and the absence of attackers can be guaranteed. Hence, only for packets deemed visible for the DUT the costly signing needs to be actually executed. Which packets are ultimately visible for the DUT is determined by the radio propagation model used in the simulation through signal loss induced by distance, interference and shadowing effects. Since the nodes in a VANET are usually vehicles and thus highly mobile, the signal loss occurring for a packet can only be calculated when it is actually transmitted, i.e. leaving the sender at the (virtual) antenna. Those time-varying environmental parameters cannot be determined in advance. Even the exact transmission time is unfeasible to determine because of various queues located in the MAC and GeoNetworking layer. These queues can delay packet transmissions diversely. Thus, at the time of encapsulating the payload in a secured message it is very hard to predict if the packet will be received by the DUT's simulated counterpart later on.

In ordinary ITS stack implementations the security entity calculates the signature at this encapsulation time point, no matter if anybody is listening to its transmissions. Contrary to specialised OBUs, where cryptographic calculations are usually accelerated by a dedicated Hardware Security Module (HSM), the simulation shall be executable on off-the-shelf computer hardware and has to calculate signatures of all simulated nodes. Because of the increased amount of calculations and the lack of specialised hardware there is a need for strategies reducing this computational burden. For our simulation-centric stack the already implemented deferred (or lazy) serialization concept is extended. Deferred serialization postpones the actual conversion of packet header data structures to their binary form until this format is actually needed, e.g. when it has to be transmitted over a real network card. Packets circling within the simulation keep their original data structures per layer, so unnecessary serialisation and deserialisation procedures are omitted. One has to keep in mind, though, that all data structures are already complete when they are added to a packet, i.e. all required computations except for the straightforward conversion to its binary representation are finished.

For signature creation this concept has to be taken one step further by postponing the computation itself and thus the completion of the data structure. This is conceptually related to lazy evaluation (Henderson and Morris, 1976) known from some programming languages like Haskell. The term "lazy cryptography" is avoided in favour of "deferred

cryptography" because cryptography itself is not weakened. This might be confused with laziness otherwise. While the signature length is known a-priori, the actual signature's bytes are comparatively expensive to calculate. Thus, a placeholder is put in the secured message's data structure instead of the signature. This placeholder can later be converted to the real signature on-demand, e.g. when the secured message needs to get serialised or a receiver wants to verify the incoming packet. For this purpose the placeholder contains a *Future*, which can be understood as asynchronous return value. *Futures* are a feature of several programming languages for concurrent code execution (Baker and Hewitt, 1977). Signature futures in our model contain the private key, the data to be signed and the associated function for the actual signature calculation.

This endeavour can be considered complementary to other approaches reducing the security processing overhead as described by Jin and Papadimitratos (2015). Their approach reduces the average computational costs for packet verification by a single network participant. From a global perspective, however, every signature still needs to be verified by at least one receiver. The approach of this paper then again aims at accelerating the signing process for testbeds. Computational load is concentrated at one machine instead of being spread over a multitude of OBUs.

4. EVALUATION

The usefulness of the proposed model is evaluated by measuring the execution times of *Artery* running a map scenario with various traffic densities and parameters influencing the security handling. All measurements are based on the LuST scenario, a SUMO road network modelling the traffic of Luxembourg city (Codeca et al., 2015). Four traffic variants (T1-4) have been investigated as summarised in Table 1. These variants differ in the starting times when observation of network communication starts respectively. Later starting times are tied with more vehicles on the roads. While the total number of transmitted (Tx) packets increases linearly with the number of vehicles, the total number of received (Rx) packets rises overproportionally due to denser traffic. All transmissions in these scenarios carry CA messages, which are generated accordingly to the rules specified in ETSI (2014b, Section 6.1.3). Since these rules evaluate a vehicle's dynamics – changes in position, speed and direction – the vehicle's route in combination with surrounding traffic affects the number of transmissions per vehicle. For example, the DUT's transmission numbers are of same magnitude but unequal for all four 30 s long traffic scenarios. More overall traffic, however, does not automatically imply more packet receptions as the comparison of receptions by DUT for T3 and T4 shows.

Since the execution times of the simulation are highly dependent on the performance of the used computer, a baseline measurement is given for which no secured messages are used at all. This baseline measurement rests upon the *off* configuration with switched off *itsGnSecurity* option in the GeoNetworking layer (ETSI, 2014a, Annex G), so no messages are encapsulated in a security envelope. Thus, these packets are faster to generate and occupy less time on the wireless channel due to shorter packet length.

Table 1. Summary of simulated scenarios

	traffic volumes			
	T1	T2	T3	T4
starting time t_0	180 s	460 s	90 min	210 min
simulation time window	$[t_0, t_0 + 30\,\mathrm{s}]$			
number of vehicles at t_0	49	101	204	341
Tx packets (DUT)	153	106	133	105
Rx packets (DUT)	189	1023	2448	2080
Tx packets (total)	6146	13786	26051	44142
Rx packets (total)	16394	88565	395716	956284

Table 2. Overview of security configurations

configuration	sign	verify
off	disabled	disabled
full	by each sender	by each receiver
full-trust	by each sender	skipped
defer	prepared by sender,	by each receiver
defer-trust	calculation on demand (once for all)	skipped

One vehicle is selected to represent the DUT in each traffic scenario T1-4, i.e. this vehicle acts as DUT's proxy in the simulated environment. The proxy's behaviour is different from other simulated vehicles: Packets received by this proxy are going to be forwarded to the DUT and packets emitted by the DUT are injected through this proxy into the simulated VANET. Because the measurements in this paper shall not be influenced by a particular DUT's behaviour, no DUT hardware is used in this evaluation setup. The proxy's special behaviour is still considered by serialising all its received packets into binary form as if they were going to be transmitted over-the-air. Vice versa, injected packets are only present in binary form as well, whereas packets originating from other nodes are available as data structures, so no parsing is required for these. Consequently, the induced computational load for this setup is mostly the same as with a DUT attached to the testbed.

Contrary to *off*, the *full* configuration computes signatures for every generated packet and hence communication between all network nodes is cryptographically secured. Configuration *defer* implements on-demand packet signing as presented in Section 3, i.e. only signatures of packets eventually received by the proxy are actually calculated. Nevertheless, the overhead in terms of packet length is still respected for all packets. Those two configurations with a *-trust* suffix exhibit a modified packet verification behaviour: The signature of every received packet is verified with configuration *full*, whereas with *full-trust* no verification takes place, i.e. any packet is blindly trusted. This approach eliminates a lot of expensive calculations and can be tolerated for a testbed where no real harm can be done. Even more, trusting all packets circulating within the simulation is a prerequisite for efficient operation with on-demand signatures. Otherwise, the signature of every packet would need to be calculated albeit being deferred, except when a packet is not received by any network node at all. Table 2 summarises all studied configurations along with their main implications.

Measured execution times of two generic computer systems are shown in Table 3 and 4, respectively. Depending on the traffic volume, *defer-trust* is capable to truncate

Table 3. Execution times in seconds (System A)

configuration	T1	T2	T3	T4
	each simulation with 30 s duration			
off	8.7	23.8	77.8	179.2
full	93.7	463.7	2015.0	4849.6
full-trust	16.2	47.6	154.0	349.3
defer	94.3	465.2	2018.7	4863.2
defer-trust	12.4	39.3	140.1	324.2

Table 4. Execution times in seconds (System B)

configuration	T1	T2	T3	T4
	each simulation with 30 s duration			
off	64.9	179.4	607.2	1408.2
full	228.1	1043.7	4436.6	10678.5
full-trust	70.5	194.5	644.9	1483.7
defer	227.3	1043.2	4437.4	10681.0
defer-trust	68.0	186.6	622.3	1449.4

execution times compared to *full-trust*: On system A, *defer-trust* shrinks to 77–93 % of the time required for *full-trust*. These reductions are less distinctive on system B, which is also considerably slower than system A in general. Figure 4 depicts the performance penalty induced by security relative to the configuration *off* without security features. According to this, a faster system (such as A) benefits more from the presented simulation model design than a slower system (such as B).

Because of the outnumbering packet receptions compared to transmissions, omitting verifications in *-trust* configurations has a staggering effect on the execution performance. Furthermore, a slight performance penalty is observable when all signatures need to be calculated but their calculation is only deferred. Since creating signatures at a later point in time is in fact not more computationally complex than immediate signing, it should be possible to minimise this observed penalty without too much effort, though.

So far, all cryptographic operations were realised with the help of the Crypto++ library (Dai et al., 2010). Since a large share of processing time is allotted to these operations, the selected cryptographic library can influence the overall simulation performance significantly. While this paper focusses on the outlined simulation design rather than mere performance numbers, preliminary tests were conducted using OpenSSL (Young et al., 2016) as alternative cryptographic library. For these tests, pre-built versions of Crypto++ and OpenSSL from the system's package repository were used for compiling the simulation binary. As expected, the *off* configuration results in equal run times for Crypto++ and OpenSSL. Latter, however, takes only about 60 % of Crypto++'s run time for the *full* configuration while the processed data were identical.

The employed cryptography library can be selected through a runtime parameter for each vehicle, i.e. simulated vehicles can be equipped with differing implementations. These could also make use of specialised hardware such as HSMs. We refer to the published source code of *Artery* and *Vanetza* for implementation details.

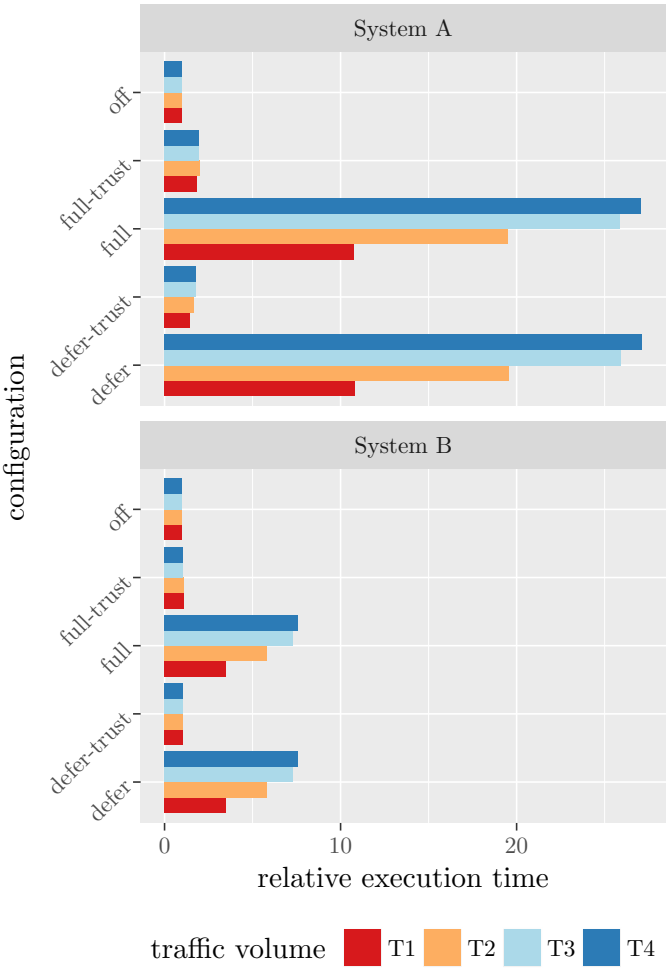

Fig. 4. Execution times relative to configuration *off*

5. CONCLUSION

Preserving real-time requirements for VANET testbeds is challenging, especially with cryptographic computations for many vehicles. However, the presented results show that computational load and execution time of VANET simulations can be reduced considerably when signature calculations are deferred. Only for packets leaving the simulation boundary these calculations are inevitable, i.e. when a DUT is attached to the simulation-based testbed and communication takes place between them.

While the created packets have a valid binary format according to ETSI standards, the current implementation is in several regards still simplistic. Identity management and pseudonym changes are currently not implemented, i.e. used identifiers such as GeoNetworking addresses and station identifiers in CA messages are static. This is not much of a problem for a testbed where no real driver's privacy is put at risk because of this limitation.

In the future, the remaining packet signing and verifying operations could be sped-up by a more optimised implementation of the ECDSA algorithm, possibly utilising the power of a Graphics Processing Unit (GPU). Another acceleration approach might be offloading of cryptographic calculations onto a HSM if available. This requires, however, that this

chip has a sufficient number of certificates at disposal and can switch between them fast enough for mimicking all of a DUT's simulated neighbours.

For demonstration purposes we issued all certificates ourselves, i.e. a DUT has to be configured in a way accepting our certificate and vice versa. The testbed can be easily extended to use other certificates as well, though. The computational costs are the same for certificate chains of same length, so conclusions about the proposed testbed design are still valid.

Future revisions of the standards might change security interfaces: The aforementioned security profile is going to be replaced by a more sophisticated certificate selection method based on ITS Application Identifiers (AIDs) and required Service Specific Permissions (SSPs). These data fields are also included in ITS certificates and thus allow plausibility checks whether a vehicle is actually authorised to set certain application data fields, e.g. special options reserved for emergency vehicles. While the encoding of these features is already implemented, the security entity itself does neither check nor use them for now.

We welcome contributions of other researchers to both open-source projects – *Artery* and *Vanetza* – which include the implementation of the ideas outlined in this paper.

ACKNOWLEDGEMENTS

Remarkable implementation efforts concerning various security features of *Vanetza* have to be attributed to following students: Stefan Kopitzki, Robert Lamprecht, Christina Obermaier, Aaron Seidler, and Sebastian Urich.

REFERENCES

Baker, Jr., H.C. and Hewitt, C. (1977). The incremental garbage collection of processes. In *Proceedings of the 1977 Symposium of Artificial Intelligence and Programming Languages*, 55–59. ACM, New York, NY, USA.

Basaras, P., Maglaras, L.A., Katsaros, D., and Janicke, H. (2015). A robust eco-routing protocol against malicious data in vehicular network. In *8th IFIP Wireless and Mobile Networking Conference (WMNC 2015)*. IFIP.

Bibhu, V., Roshan, K., Singh, K.B., and Singh, D.K. (2012). Performance analysis of black hole attack in VANET. *International Journal of Computer Network and Information Security (IJCNIS)*, 4(11), 47.

Buttyán, L., Holczer, T., and Vajda, I. (2007). On the effectiveness of changing pseudonyms to provide location privacy in VANETs. In *Security and Privacy in Ad-hoc and Sensor Networks*, 129–141. Springer.

Codeca, L., Frank, R., and Engel, T. (2015). LuST: a 24-hour scenario of Luxembourg city for SUMO traffic simulations. In *SUMO2015 – Intermodal Simulation for Intermodal Transport*. Deutsches Zentrum für Luft- und Raumfahrt e.V. Institut für Verkehrssystemtechnik.

Dai, W. et al. (2010). Crypto++. URL `http://www.cryptopp.com`. Version 5.6.1.

DOT (2011). *Vehicle Safety Communications - Applications (VSC-A) Final Report: Appendix Volume 3 Security*. U.S. Department of Transportation, National Highway Traffic Safety Administration.

Douceur, J.R. (2002). The sybil attack. In *Peer-to-peer Systems*, 251–260. Springer.

ETSI (2010). *EN 302 665 - Intelligent Transport Systems (ITS); Communications Architecture*. European Telecommunications Standards Institute, v1.1.1 edition.

ETSI (2014a). *EN 302 636-4-1 - Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality*. European Telecommunications Standards Institute, v1.2.1 edition.

ETSI (2014b). *EN 302 637-2 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*. European Telecommunications Standards Institute, v1.3.2 edition.

ETSI (2015). *TS 103 097 - Intelligent Transport Systems (ITS); Security; Security header and certificate formats*. European Telecommunications Standards Institute, v1.2.1 edition.

Henderson, P. and Morris, Jr., J.H. (1976). A lazy evaluator. In *Proceedings of the 3rd ACM SIGACT-SIGPLAN Symposium on Principles on Programming Languages*, POPL '76, 95–103. ACM, New York, NY, USA.

Jin, H. and Papadimitratos, P. (2015). Scaling VANET security through cooperative message verification. In *Vehicular Networking Conference (VNC), 2015 IEEE*, 275–278.

Larson, U.E., Nilsson, D.K., and Jonsson, E. (2008). An approach to specification-based attack detection for in-vehicle networks. In *Intelligent Vehicles Symposium, 2008 IEEE*, 220–225. IEEE.

Maglaras, L.A. (2015). A novel distributed intrusion detection system for vehicular ad hoc networks. *International Journal of Advanced Computer Science and Applications(IJACSA)*, 6(4), 101–106.

Mejri, M.N., Ben-Othman, J., and Hamdi, M. (2014). Survey on VANET security challenges and possible cryptographic solutions. *Vehicular Communications*, 1(2), 53–66.

Müter, M., Groll, A., and Freiling, F.C. (2010). A structured approach to anomaly detection for in-vehicle networks. In *Information Assurance and Security (IAS), 2010 Sixth International Conference on*, 92–98. IEEE.

Pathan, A.S.K. (2010). *Security of self-organizing networks: MANET, WSN, WMN, VANET*. CRC press.

Riebl, R. and Facchi, C. (2014). Implementation of day one ITS-G5 systems for testing purposes. In *Proceedings of the 2nd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication (FG-IVC 2014)*, 33–36.

Riebl, R., Günther, H.J., Facchi, C., and Wolf, L. (2015). Artery – extending Veins for VANET applications. In *Models and Technologies for Intelligent Transportation Systems (MT-ITS)*.

Vandenberghe, W., Moerman, I., Demeester, P., and Cappelle, H. (2011). Suitability of the wireless testbed w-iLab.t for VANET research. In *18th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*.

Verendel, V., Nilsson, D.K., Larson, U.E., and Jonsson, E. (2008). An approach to using honeypots in in-vehicle networks. In *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th*, 1–5. IEEE.

Young, E., Hudson, T., et al. (2016). OpenSSL. URL `http://www.openssl.org`. Version 1.0.2g.