Generic Framework Enabling Secure and Efficient Automotive Wireless SW Updates

Marco Steger*, Michael Karner*, Joachim Hillebrand*, Werner Rom*, Carlo Boano†, and Kay Römer†

*Virtual Vehicle Research Center, Graz, Austria
†Institute for Technical Informatics, Graz University of Technology, Graz, Austria
Email: marco.steger@v2c2.at

***Abstract*—Future vehicles will be wirelessly connected to nearby vehicles, to the road infrastructure and to the Internet in order to enable new comfort features, safety functions and a number of new vehicle-specific services. The latter will include a fast, secure, and reliable way to remotely diagnose and reconfigure a vehicle as well as to install new software on the electronic control units integrated in a vehicle. Such wireless software updates are beneficial for both automotive OEMs and customers, as they allow to enable new features of the vehicle remotely and to fix software bugs by installing a new software version over the air. Wireless diagnostics and software updates are required in several stages of a vehicle's lifetime: from the manufacturing stage on the assembly line and the maintenance in a workshop to the remote download of up-to-date software directly by the car owner. To support this process over a whole vehicle's lifetime, a generic framework is needed. In this paper we propose a generic framework enabling secure and efficient wireless automotive SW updates and hence supporting a vehicle's whole lifetime. We describe the IEEE 802.11s network used as wireless medium to interconnect vehicles and diagnostic devices in a reliable, trustworthy and fast way and propose a dedicated cross-layer security concept applying strong authentication as well as encryption mechanisms.**

I. Introduction

A modern vehicle includes a growing number of electronic control units (ECU) allowing to incorporate new features and services. Enabling these features and services, however, requires elaborate and complex software (SW) implementations installed on these ECUs, potentially introducing a growing number of bugs in the automotive software.

New concepts allowing efficient automotive SW updates are required to fix such SW bugs, as well as to enable the remote installation of new features and to support the development and maintenance of modern vehicles. Efficient and secure SW updates can be beneficial over the whole life-cycle of a modern vehicle and can significantly reduce the time needed for vehicle maintenance.

In this paper we propose a generic framework enabling wireless SW updates as well as wireless diagnostics that is meant to support the whole life-cycle of a modern vehicle. Thereby, the proposed framework considers the requirements coming from different application scenarios (i.e., vehicle development, vehicle assembly line, maintenance in workshops, and wireless remote updates and diagnostics). In all these scenarios, the wireless SW updates have to be fast, efficient, reliable, and secure. To enable efficient and fast wireless SW updates, our framework will support parallel SW updates, where the same SW binary will be installed on different vehicles and ECUs simultaneously.

978-1-5090-1314-2/16/$31.00

Additionally, we present a security concept that uses strong authentication and encryption mechanisms and also ensures the integrity of the transferred data as well as of the entire vehicle protecting the diagnostic devices, the transferred data, the vehicles and the OEM backbone from unauthorized access.

The vehicles and the diagnostic devices are interconnected using an IEEE 802.11s mesh network [1]. In [2] we already proved the applicability of IEEE 802.11s for automotive applications. The mesh characteristics of an IEEE 802.11s network increases the reliability as well as the flexibility of the network thanks to its multi-hop capability and the resulting redundant paths. The IEEE 802.11s standard also supports multicast data streams, which can be used to realize parallel and therefore efficient SW updates by sending a SW binary through the wireless network to several vehicles simultaneously.

After reviewing related work in the next section, we describe our generic framework and highlight the role of involved devices as well as users in the considered application scenarios in Section III. Thereafter, in Section IV, we illustrate the properties and advantages of the employed IEEE 802.11s [1] network, explain how it enables the construction of a large mesh network and illustrate how to exploit its characteristics to perform reliable and efficient wireless SW updates. Section V provides an overview of the performed security analysis as well as the resulting security concept. The results of the performed framework evaluation can be found in Section VI.

The key contribution of this paper is the introduction of a generic framework for wireless automotive SW updates that is applicable for different application scenarios, namely: wireless SW updates in the vehicle development, in the assembly line, during maintenance in workshops, as well as wireless remote updates. Our framework addresses the *efficiency* aspect by enabling parallel SW updates using IEEE 802.11s multicast data streams, and *security* by presenting a novel robust security concept for wireless SW updates.

II. Related work

Wireless automotive SW updates were already addressed in previous work. In [3] the advantages of wireless SW updates compared to traditional wired SW updates are highlighted and a high-level architecture for over-the-air updates is presented. Technical details such as a description of the wireless medium or required security mechanisms, however, are not covered.

Idrees et al. [4] proposed a system for wireless over-the-air updates, where a HW Security Module (HSM) is used for data encryption, data integrity, as well as key management on both the wireless interface and all ECUs of a vehicle. Data encryption and verification is handled directly on the concerned ECU. This approach, however, requires an HSM on every ECU, which leads to significant extra costs. Additionally, the proposed system does not describe the specific type and properties of the wireless link.

Nilsson et al. [5], [6] have proposed a system, where a vehicle is connected to a portal server via an Internet link to install new automotive SW over the air. The authors highlight the importance of the security aspects data integrity and data confidentiality for wireless SW updates. However, neither the wireless network nor the used wireless protocol and the data flow in the network have been addressed.

Mahmud et al. [7] have proposed an architecture for secure SW updates in which the integrity of data is ensured by sending multiple copies of the SW. The proposed solution, however, is dedicated to unicast links between one vehicle and an OEM sever and relies on a number of prerequisites and assumptions (e.g., ensuring secure SW updates by sending multiple copies).

Although the authors are addressing several security aspects, the solutions described above are only applicable for wireless point-to-point links and only the remote SW update scenario is covered. Multicast data streams and hence parallel SW updates cannot be realized.

Tesla is currently the only OEM providing a solution for wireless remote updates over the air. To this end, Tesla is using a dedicated wireless link (mainly 3G, but a vehicle can also be connected to the user's Wi-Fi at home) to securely transfer new SW from the manufacturer servers to a vehicle [8]. This point-to-point connection (i.e., between server and vehicle), however, cannot be used to simultaneously install SW in different vehicles and on several ECUs in parallel.

The authors of [4]–[7] have specifically focused on secure wireless SW updates and highlighted a number of security threats and aspects, namely:

- Vehicle integrity and authentication
- Data integrity
- Data confidentiality
- Key management and exchange

Different from previous work, our security concept presented in Section V-A addresses all these aspects at once.

A. IEEE 802.11s Mesh Network

Our generic framework for wireless SW updates is based on an IEEE 802.11s mesh network. The IEEE 802.11s standard provides multicast data streams and therefore we can realize parallel SW updates in our framework.

To the best of our knowledge there are no other wireless SW update solutions based on IEEE 802.11s available, however, IEEE 802.11s is used in other automotive applications.

For example, in [9] and [10], IEEE 802.11s is used as a backbone network for V2X (vehicle to vehicle and vehicle to infrastructure communication) networks. The main idea is to replace the wired connections between the RSUs (road side units) and the V2X servers by wireless ones. In [2] we proved the applicability of IEEE 802.11s for an automotive application by performing different experiments in an automotive environment. These experiments, however, were not dedicated to wireless SW updates but focusing on reliable wireless data transfer in an automotive context.

In Section IV we describe the mesh characteristics of the used IEEE 802.11s network in more detail and highlight the advantages of the latter w.r.t. wireless automotive SW updates.

B. IEEE 802.11s Security

Several contributions have focused on the security aspects of IEEE 802.11s and proposed possible improvements and extensions. These aspects, however, have only been discussed outside of the automotive domain. Tan et al. [11] have described internal as well as external attacks on IEEE 802.11s networks. Security requirements for systems based on IEEE 802.11s were extracted and these requirements were used to evaluate different approaches. The results of this evaluation have shown that none of the evaluated approaches (e.g., [12] and [13]) was able to satisfy the desired security requirements.

Sbeiti et al. [14] have proposed to use GPS positioning to mitigate a wide range of potential attacks. The system uses a combination of digital signatures, lightweight authentication trees and symmetric block ciphers called PASER. In [15] the same authors also evaluate and compare other approaches for secure IEEE 802.11s networks to PASER. They highlight the inefficiency of these approaches w.r.t. time (delay of about 70ms per-hop) and power as well as their sensitivity to blackhole and wormhole attacks. However, the use of GPS, as proposed in [14] and [15], is not feasible inside a workshop building or the assembly line and therefore PASER is not applicable for our wireless SW update system.

Although several approaches to create secure IEEE 802.11s networks exist [11]–[15], none of these approaches is fulfilling both the efficiency requirements of our framework for wireless SW updates and the immunity against possible attacks. Because of that, our security concept described in Section V is using a lightweight security mechanism based on pre-shared keys on the network layer and additional strong authentication as well as encryption mechanisms on the application layer.

In particular, our security concept is based on a structured security analysis. In [16] we have described the security analysis in detail and showed how it can be used to analyze an automotive application w.r.t. the new automotive security standard SAE J3016 [17]. A security concept enabling wireless SW updates, however, was not part of this work.

III. Framework Description and System Overview

In this section we first describe the considered application scenarios for wireless SW updates and diagnostics in more detail (Section III-A). Thereafter, in Section III-B, we list the involved nodes, devices, and tools and sketch the data flow between these entities in our framework.

A. Considered Application Scenarios

In this section we use the following descriptions of the considered application scenarios to identify the scenario-specific requirements as well as its peculiarities and describe the corresponding environment w.r.t. the user education, available infrastructure, and security concerns. In Table I these aspects are summarized.

In the **vehicle development scenario**, development engineers have to update the SW of one or more ECUs several times to evaluate and test newly developed features. A flexible and efficient framework for SW updates and vehicle diagnostics is required to support the development engineers in their work. Vehicle development activities will primarily take place in a restricted environment and performed by expert users.

Vehicle assembly is performed in a highly automated and secure environment where many working steps are performed by machines and robots. Before a vehicle can leave the assembly line, the latest SW version shall be installed on all integrated ECUs. Therefore, the SW of many vehicles must be updated – ideally in parallel – to install the latest SW on the ECUs of these vehicles. Because of the high number of vehicles as well as the high degree of automation, the scalability, reliability and efficiency of the SW update system are very important.

In a typical **workshop scenario** mechanics will diagnose, repair and maintain several vehicles. Therefore, the mechanics (trained users) will connect to a vehicle, run some diagnostic functions, look for Diagnostic Trouble Codes (DTC) on the ECUs and perform necessary repairs. If new SW is available for a vehicle, the mechanics additionally install the new SW. Parallel SW updates would be very beneficial especially when large vehicle recalls (e.g., due to a critical SW bug) are necessary: a mechanic can connect to several vehicles in parallel and install the new SW simultaneously.

Remote SW updates are mainly relevant for future vehicles with an integrated wireless interface to the vehicle. To remotely install new SW on a vehicle, the vehicle owner will first be informed about the possible update. In the next step, the user will choose a suitable time slot for the SW update (the vehicle cannot be used while a SW update is performed due to safety reasons). The data will then be transferred to the vehicle either via a dedicated 3G/4G connection or when the vehicle owner connects the vehicle to his home WLAN.

In Table I we collected the application-scenario-specific information w.r.t. wireless SW updates. Thereby we focus on the user and his education, important requirements w.r.t. reliability, efficiency, etc., as well as the required security level.

SW updates in the assembly line as well as in the vehicle development phase are performed in secured areas by expert users. Therefore, security is still an important issue (e.g., industrial espionage) but not as critical as in the remote SW update case, were a SW update is performed in public or at the users home, potentially using an insecure network by an untrained user. Especially the assembly line as well as the workshop scenario require a very efficient and fast way to install SW updates. High flexibility by easily extending the transmission range of the wireless network is especially required during vehicle development due to the big variety of system evaluations, function tests and diagnostics performed in this phase.

TABLE I
Involved users, aspects and required security level w.r.t. wireless SW updates in the considered application scenarios.

Scenario	User	Aspects	Security Level
Development	Development engineer; expert	Flexible, efficient	Medium
Assembly line	Operator; expert	Scalable, reliable, efficient	Medium
Workshop	Mechanic; trained user	Efficient, backward compatibility	High
Remote	Vehicle driver or owner; untrained user	Easy to use	Very high

B. System Overview

To interconnect the vehicle with the wireless network, a reliable and secure interface is required. This **Wireless Vehicle Interface (WVI)** is the core component of our framework for wireless SW updates. Independent from the application scenario, the WVI interconnects the in-vehicle communication system, consisting of different automotive bus systems (e.g., CAN) and ECUs, with the wireless network and therefore with the outside world.

We see two different types of WVIs: either a fully integrated device or a plug-in solution. The plug-in WVI can be temporarily connected to the vehicle via the OBD interface. This is mainly relevant for the workshop scenario, where the plug-in property and the use of standardized protocols for the in-vehicle communication are ensuring backward compatibility. The fully integrated WVI is either realized as a dedicated ECU or as a component of a smart gateway interconnecting different bus systems and therefore part of the in-vehicle communication system.

The **Diagnostic Tester (DT)** is either a dedicated device (e.g., tool in a workshop) or more likely a SW application running on a laptop, PC or server. The DT provides information about the vehicle configuration (e.g., type and CAN IDs of the ECUs), availability of new SW versions, as well as keys to authorize a SW update. The information is either stored locally on the device or can be acquired using a dedicated and trusted connection to an OEM server. Additionally, a DT supports various diagnostic functions. In the remote SW update scenario the DT will be located at a facility of the OEM and a secure Internet link (e.g., a VPN tunnel) between the WVI and the DT will be established.

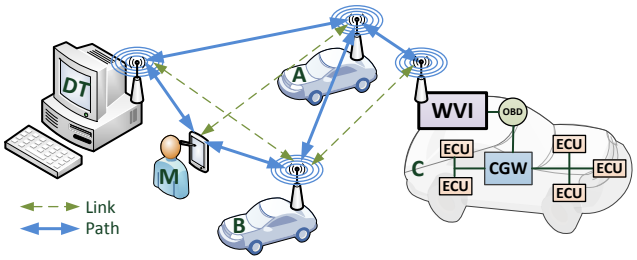

Fig. 1. IEEE 802.11s wireless mesh network applied in a typical workshop scenario. A mechanic M can connect to a vehicle either directly (e.g., vehicle B) or using a multi-hop path (e.g., vehicle C via vehicle B and vehicle A).

Handheld devices can be used to trigger, monitor, and validate the SW update process. Additionally, they can be used by mechanics in a workshop or by a development engineer in the vehicle development phase to run wireless diagnostics and to monitor the vehicle bus.

C. Setup Phase and Data Flow

In Figure 1 a typical workshop scenario is sketched, where mechanics maintain vehicles by using handhelds to run wireless diagnostics and trigger wireless SW updates. We will use the illustrated scenario to explain the setup phase and the data flow in our framework: similar procedures are used in the other scenarios. To simplify the description of the processes below, we will not cover the performed security steps. However, in Section V-A we will refer to this example again to illustrate our security concept.

A mechanic first connects a WVI to the vehicle to be maintained using the OBD interface. The WVI powers up and joins the wireless IEEE 802.11s network. The WVI now waits for beacons advertising the presence of a DT. These beacons are broadcasted periodically and once the WVI receives those beacons, it will establish a connection to the DT. Also handhelds can connect to the WVI by sending beacons. The mechanic can now use the connected handheld device to run wireless diagnostic and, if a new SW version is available, will trigger a wireless SW update. Therefore, the SW will be transferred from the DT to the WVI. Note that a new SW version is not stored on the handheld devices due to security reasons.

Our generic framework supports different SW update modes: most likely the SW binary will first be fully transferred from the DT to the WVI. The latter will then verify the received binary, and install the binary on the ECU using the Unified Diagnostic Service (UDS) protocol [18] (*two-stage approach*). When using the *direct-programming mode*, the DT will send packets containing UDS-complaint data chunks to the WVI and the latter unpacks the data and forwards it to the ECU. Therefore, the WVI only acts as a gateway forwarding the data and the DT has to take care about authorization with the ECU, data transfer, and verification of the SW update. Our framework can also support *delta downloads*, where only the difference between the current and the new SW version is sent to the ECU. This mode, however, requires that also the ECU itself supports this delta download mode (which is not yet fulfilled by current ECUs).

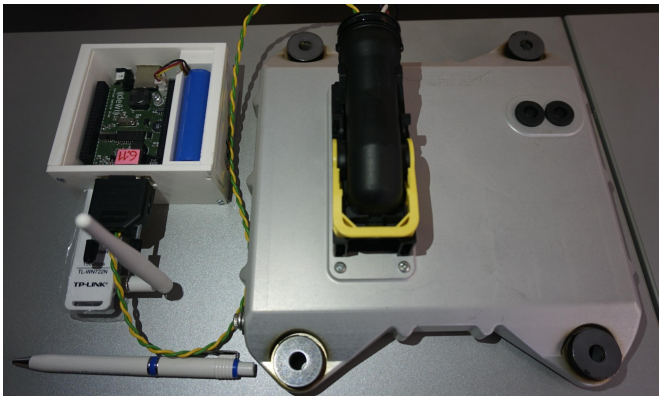

Fig. 2. WVI prototype based on a BeagleBone Black (left) and the Volvo ECU used to test the SW update (right).

To perform a SW update on several vehicles in parallel, a mechanic will register a vehicle for a certain SW update and the DT will then automatically start the update process when all vehicles are registered and ready to receive the new SW.

D. Prototype Implementation

We developed prototypes of WVI, DT, and handheld device to evaluate the framework, the data flow, and the individual components.

The WVI prototype is shown in Figure 2 and consists of a BeagleBone Black (BBB) board and an additional, self-designed PCB, the DEWI[1] cape. This cape includes the required HW interfaces (i.e., CAN and OBD) and the corresponding transceivers to connect the WVI with a vehicle. Additionally, the cape is handling the battery management of the WVI and provides some means to measure the power consumption of the board as well as of the individual radio. The WVI SW is mainly developed in Java and the Java Native Interface (JNI) is used to control the HW-related parts.

The DT is implemented in Java and the implementation was tested on a dual-core laptop running Win7 as well as on a PC running Debian Linux. We developed a GUI providing different modes such as ECU programming, OBD diagnostics, and CAN bus monitoring.

A Nexus 7 tablet running Android is used as handheld device. The current version of the developed Android application allows to connect to a WVI and a DT simultaneously, to monitor the vehicle bus, and to exchange status information with the DT. Currently, we are extending the application to fully support the defined security mechanisms. The new version of the application can then be used to trigger, monitor and validate the SW update process.

[1]The ARTEMIS project DEWI (Dependable Embedded Wireless Infrastructure) focuses on the area of wireless sensor / actuator networks and wireless communication. With its four industrial domains (Aeronautics, Automotive, Rail, and Building) and 21 industry-driven use cases, DEWI will provide and demonstrate solutions for wireless seamless connectivity and interoperability in smart infrastructures [19]. (For further details see www.dewi-project.eu)

IV. Wireless IEEE 802.11s Network

Our generic framework supporting efficient and secure wireless SW updates as well as wireless diagnostics uses an IEEE 802.11s network to interconnect the devices described in Section III. Contrary to other standards out of the IEEE 802.11 protocol family, where an access point is used to interconnect the nodes of a network, the IEEE 802.11s protocol is based on a mesh network, and each node can directly communicate with other nodes in its transmission range or use other nodes in between to forward a data packet to its final destination [20]. Because of the mesh characteristics of IEEE 802.11s, a data packet can use different paths when sent through the network and this redundancy increases the reliability of IEEE 802.11s networks.

Additionally, the transmission range of the network can be increased by adding relay nodes at the edge of the network. Thanks to the multihop capability of mesh networks, an IEEE 802.11s network can even be used in difficult environments (for wireless communications) and thus can fulfill the requirements of the considered application scenarios.

For example, in a typical workshop scenario, wireless links will be affected by the shielding of the vehicles and other (metal) objects: if the direct link between a DT and a vehicle is too weak to send a packet directly from the DT to the WVI, other vehicles parked in between may be used to forward the data packet to the target vehicle (also illustrated in Figure 1).

A. Links and Paths in IEEE 802.11s

Figure 1 illustrates a mesh network with five nodes in a typical workshop scenario. The Hybrid Wireless Mesh Protocol (HWMP), used as the default routing algorithm in IEEE 802.11s networks, automatically finds the best (reliable and fast) route between all nodes in the network [20]. In an IEEE 802.11s network we have to distinguish between links and paths. An established link between two nodes (e.g., vehicles B and C) means that the nodes are in transmission range of each other. However, this doesn't always mean that data between these nodes is exchanged directly. If a link is weak (e.g., several packets are lost when sent directly from vehicle B to vehicle C) the HWMP tries to find a better way to route the packets through the network. In IEEE 802.11s these routes are called paths (e.g., between vehicle A and B or mechanic M and DT).

B. Multicast in IEEE 802.11s

In our framework we use open11s [21], an open source implementation of the IEEE 802.11s standard available for Linux. open11s is still under development and doesn't include all features described in the IEEE 802.11s standard. However, as open11s is open-source, missing features can be added using (self-implemented) patches.

The IEEE 802.11s standard also describes multicast data streams in the mesh network. Such a multicast can be used to send data packets from one node to several other nodes in a network. In our framework we use multicasts to transfer and install a new SW version on several vehicles simultaneously (i.e., parallel SW updates). Although the IEEE 802.11s standard defines multicast data streams, the current open11s implementation does not support multicasts yet. We are currently evaluating the reliability of experimental patches enabling multicast data exchange for open11s in our IEEE 802.11s testbed consisting of twelve nodes. In parallel we plan to develop our own reliable multicast and to compare the performance of the different approaches using our wireless testbed.

The framework for wireless SW updates and diagnostics as well as the corresponding security concept presented in Section V, however, is already designed and implemented in a way that it will fully support parallel SW updates.

V. Security and Trust

Security is a critical aspect of wireless SW updates. Depending on the application scenario, different levels of security are required (as described in Section III-A). In this section we briefly describe how security is addressed in our framework and illustrate the required steps towards a security concept for wireless SW updates. These steps can be summarized as follows:

1) ***Security analysis*** of the framework resulting in a secure system configuration.
2) ***Extraction of the security requirements*** from the secure system configuration.
3) ***Definition of a security concept*** based on the security requirements and the peculiarities of the application scenario.
4) ***Evaluation of the defined security concept*** using the STRIDE threat model [22].

We used the DEWI security metric [16] to perform a structured security analysis. The DEWI security metric is based on the SHIELD multi-metrics approach [23], [24] and can be used to analyze the different application scenarios described in Section III at once. The analysis results in a secure system configuration for every application scenario and the security-related requirements can be directly extracted from these secure system configurations. In [16] we described the DEWI security metric in more detail and presented a case study illustrating its usage.

A. Security Concept

The security concept is based on the results of the structured security analysis using the DEWI security metric: the secure system configurations and the corresponding security requirements.

As described in Section II, our security concept applies security mechanisms on the network as well as on the application layer. We utilize wpa_supplicant, a generic security framework for different types of wireless networks, to secure the wireless IEEE 802.11s network in a lightweight way based on existing mechanisms provided by the current open11s implementation. SAE [25], developed especially for 802.11s-based, multi-hop capable mesh networks, is fully integrated

in the latest wpa_supplicant version and thus can be used to secure our wireless IEEE 802.11s mesh network.

On the application layer different keys and security mechanisms are used. Thereby our security concept addresses the four security aspects stated in Section II: vehicle integrity and authentication, data integrity, data confidentiality, key management and exchange.

Vehicle integrity and authentication require strong authentication mechanisms. Additionally we have to avoid that an attacker can endanger a whole fleet of vehicles by breaking one vehicle and extracting the shared authentication key. Thus, a unique key shall be used on every vehicle. In our concept we are using a key pair consisting of a private and a public key (asymmetric keys) on the WVI as well as the DT ensuring an unambiguous authentication between all nodes without using the same (symmetric) key on all vehicles. This *master key pair* is used to handle the **authentication between a WVI and a DT** and to **encrypt as well as sign unicast data**.

Secure multicasts needed to enable parallel SW updates, however, require symmetric keys for data encryption and verification. A data packet sent from the DT to several vehicles must be encrypted using a shared key. This shared *session key* is created by the DT and then sent from the DT to each WVI using a unicast data packet encrypted with the public master key of the WVI and signed with the private master key of the DT. The session key can then be used to encrypt and sign the multicast data packets and hence ensuring **data confidentiality** as well as **data integrity**.

All keys used in our security concept must be kept secret and therefore stored securely on the devices to ensure that an attacker cannot extract the keys (e.g., by stealing a WVI). In our concept, keys are either stored in dedicated secure memory or a trusted platform module is used to securely hold the keys.

We illustrate a typical workshop scenario in Section III-B and Figure 1. We now refer to the same scenario and discuss the related steps w.r.t. security:

1) ***User authentication***: the mechanic authenticates with the system using a smartcard and a PIN code. Different user profiles can be used to authorize different features: a normal mechanic can use the system to run wireless diagnostics only. A privileged user can additionally perform wireless SW updates.
2) ***WVI power up***: after connecting the WVI to the vehicle, the WVI powers up and tries to connect to the IEEE 802.11s network. Therefore, a shared network key is used.
3) ***Authentication with the DT***: the master keys of the DT and the WVI are used to authenticate with each other.
4) ***Parallel SW update***: to enable parallel SW updates, the DT first creates a session key and sends it to every concerned vehicle's WVI (unicast). Now the DT can send the SW binary to the WVIs. Thereby every transferred packet is encrypted and signed using the session key (multicast).
5) ***Data verification***: after transferring the whole binary to the WVIs, the DT computes the hash value of the entire binary, signs and encrypts the hash value using the master key and sends it to the WVIs (unicast). The WVIs can verify the received SW binary using the received hash before installing it on the concerned ECUs.

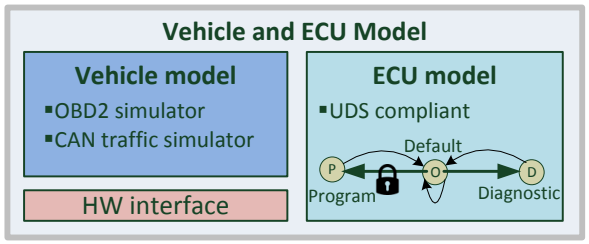

Fig. 3. Vehicle and ECU model consisting of a vehicle model, simulating traffic on a CAN bus and supporting some OBD requests, and an ECU model, implemented as state machine and supporting UDS (required for SW updates)

A more detailed description as well as a STRIDE based evaluation of our security concept can be found in [26].

VI. Framework Evaluation

In this section, we present an evaluation of our framework for secure and efficient wireless SW updates and diagnostics. The evaluation is performed in three steps: first we use our developed Vehicle and ECU Model (VEM) to perform fundamental experiments and communication tests, as well as to test the behavior in case of errors (Section VI-B). In the second step, we wirelessly connect to a real vehicle and run wireless diagnostics (Section VI-C). Finally, we evaluate our framework by installing a new SW binary on an automotive ECU provided by our project partners Volvo Trucks (Section VI-D).

A. Vehicle and ECU Model

We use the VEM to perform fundamental system evaluations, analysis of new features, and communication tests. Furthermore, the VEM can also be used to evaluate the behavior of the developed system in case of errors (e.g., sending unexpected frames) and communication problems (e.g., lost, delayed or duplicated CAN frames).

As illustrated in Figure 3, the developed VEM consists of a fundamental communication model of a vehicle as well as a simplified model of an ECU and is implemented in C/C++. A developed HW interface library can be used to communicate with a vehicle via the OBD or CAN interface.

The vehicle model can be used (i) to simulate the traffic on a CAN bus and (ii) to create a response for certain OBD requests. Bus traffic simulation is done by using log files taken from different vehicles by connecting our WVI to the OBD interface of the vehicles to collect the timestamps, CAN-IDs, and payloads of all received CAN frames. The vehicle model parses such a log file and writes the collected messages on the CAN accordingly.

The ECU model is implemented as state machine covering different diagnostic sessions described in the UDS standard [18]. This state machine is also sketched in Figure 3. By default, the ECU will power up and stay in operational mode,

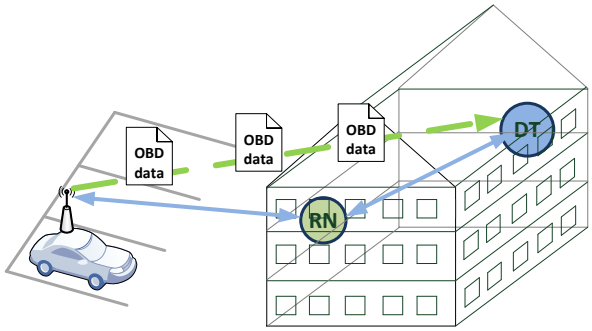

Fig. 4. Evaluation of the wireless diagnostic feature: connecting a real vehicle with the DT located in the second floor via a relay node (RN) located in the first floor.

where the ECU is performing its dedicated task. A UDS *diagnostic session control* command can be used to switch the ECU in the *programming session* or in the *extended diagnostic session* [18]. Depending on the state of the ECU (i.e., on the current UDS session) different diagnostic commands are available. An authorization step, often based on a seed & key mechanism, must be performed to empower the programming session. If the ECU is in the extended diagnostic or programming session and the session is inactive (i.e., no UDS messages received for a predefined time), the ECU falls back to the default session.

Our current ECU model supports all programming-related commands, as well as the *diagnostic session control*, *security access*, *ECU reset*, and *tester present* commands as defined in the UDS standard [18].

B. Framework development and evaluations using the VEM

The first experiments and evaluations were performed using the developed VEM. Therefore, the VEM is connected with the WVI using (i) a virtual CAN bus, where the VEM and the WVI implementation is running on the same BBB board or (ii) using a CAN connection between the WVI prototype and the VEM, running on a second BBB.

In both cases (i.e., WVI and VEM on the same BBB or VEM and WVI on their own BBB) we can test the wireless diagnostic as well as the ECU programming part of the developed system using the VEM. Therefore, the VEM can be used to continuously evaluate and test the currently developed versions of the DT and WVI prototypes.

C. Wireless diagnostics in a real vehicle

We also evaluate our framework by running wireless diagnostics on a real vehicle using our WVI prototype. As illustrated in Figure 4, the DT, running on a PC located on the second floor of our company building, is connected to the WVI via another WVI prototype, located on the first floor and acting as Relay Node (RN).

Using this experimental setup, we are able to monitor the CAN bus that is available at the OBD connector of the vehicle and use different OBD requests to read vehicle-specific parameters such as the current engine temperature or the current engine speed (RPM). However, some particular OBD requests/responses were heavily delayed or lost during transmission. We performed additional measurements using the described scenario to get accurate numbers for the Packet Error Rate (PER). Therefore we sent 10000 UDP packets with 8 bytes payload and a period of 100ms from the WVI, connected to the vehicle, to the DT, located in the office. An acknowledgment packet with the same size is then sent back to the WVI. This experiment was performed without any retransmission mechanisms on application layer and results in a PER of 0.43% for messages sent from the WVI to the DT. 0.83% of all packets sent back from the DT to the WVI were also lost. In total we lost 0.65% of all transferred packets.

Further investigations using our IEEE 802.11s testbed are planned to evaluate and increase the reliability of the network.

D. Wireless SW update using a real ECU

The SW update process was evaluated using a demo ECU provided by our project partner Volvo T rucks. Due to organizational and especially safety reasons, a SW update test in a real vehicle is not possible yet in the current stage of the project.

As shown in Figure 2, the WVI is directly connected to the demo ECU using a twisted-pair cable. The demo ECU is UDS-compliant and can be programmed in two steps: first a secondary bootloader is installed and executed. Afterwards the application binary can be installed. A reboot of the ECU is used to complete the SW update process.

Volvo Trucks provided different SW versions periodically writing CAN frames on the bus when in operational (default) mode. Every version uses its own CAN-ID and therefore we can easily verify if a new version is installed on the demo ECU or not.

The first version of the WVI HW causes some problems when connecting to the Volvo demo ECU using the CAN bus. Seconds after connecting our WVI to the demo ECU, the latter got caught in a reset loop due to a bad CAN bus state. After investigating and fixing the problem by replacing the CAN transceiver chips, we are now able to connect our WVI (revision 2) to the demo ECU and install new SW versions on it without any problems.

The SW update process for the Volvo ECU is done in two steps. First the Secondary Bootloader (SBL) is transferred to the ECU and then a UDS command is used to launch this SBL on the latter. In the second step, the actual application binary is transferred and installed on the ECU. Our framework support both the use of an SBL and the application binary as well as an approach, where the application binary is directly installed without using an SBL.

In Table II the times needed to transfer the SBL and the application from the DT to the WVI and then from the WVI to the ECU are shown. The table shows that the wireless data transfer (including data transfer, integrity check, and acknowledgments) is 12 times faster then the wired one using the CAN bus. Thus, it make sense that the WVI autonomously control the data transfer to the ECU once the binary was received from the DT (i.e., no permanent wireless connection

TABLE II
TIME NEEDED TO TRANSFER THE BINARIES FROM THE DT TO THE WVI VIA THE IEEE 802.11S LINK AND THEN SEND IT TO THE ECU VIA CAN

Binary type	Binary size	Time on 11s link	Time on CAN
SBL	67KB	0.503s	6.268s
Application	375 KB	2.527s	30.664s

to the DT needed). The WVI then informs the DT when the SW is installed on the ECU successfully or any problems occur.

VII. CONCLUSION

In this paper we proposed a generic framework enabling secure and efficient wireless SW updates for vehicles. The framework is designed such that it can fulfill the needs of several application scenarios: vehicle development, vehicle assembly line, vehicle maintenance, and wireless remote updates. The proposed framework enables parallel SW updates, where the SW of several vehicles is updated simultaneously. We also describe the properties and characteristics of the IEEE 802.11s standard and its applicability for performing wireless SW updates. Furthermore, we describe our approach to enable secure automotive SW updates and especially focus on the resulting security concept.

We are currently working on the extension and improvement of the open11s implementation with specific focus on reliable and secure multicasts and the stability of the paths in the network. The revised version of open11s will then be used to evaluate the performance of our system w.r.t. parallel SW updates.

ACKNOWLEDGMENT

We are particularly grateful to our colleagues from Volvo Trucks for their great support regarding the demo ECU.

The research from DEWI project (www.dewi-project.eu) leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreement n° 621353. The authors acknowledge the financial support of the COMET K2 - Competence Centres for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT), the Austrian Federal Ministry of Science, Research and Economy (BMWFW), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG).

REFERENCES

[1] IEEE, "Local and metropolitan area networks-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY):Amendment 10: Mesh Networking," IEEE, Tech. Rep., 2011.

[2] M. Steger, M. Karner, J. Hillebrand, W. Rom, E. Armengaud, M. Hansson, C. A. Boano, and K. Romer, "Applicability of ieee 802.11s for automotive wireless software updates," in *Telecommunications (ConTEL), 2015 13th International Conference on*, July 2015, pp. 1–8.

[3] Redbend Software, "Updating Car ECUs Over-The-Air (FOTA)," *White Paper*, pp. 1–14, 2011.

[4] M. S. Idrees, H. Schweppe, Y. Roudier, M. Wolf, D. Scheuermann, and O. Henniger, "Secure automotive on-board protocols: A case of over-the-air firmware updates," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6596 LNCS, pp. 224–238, 2011.

[5] D. K. Nilsson and U. E. Larson, "Secure firmware updates over the air in intelligent vehicles," *IEEE International Conference on Communications*, pp. 380–384, 2008.

[6] D. Nilsson, P. Phung, and U. E. Larson, "Vehicle ECU classification based on safety-security characteristics," *Road Transport Information and Control - RTIC and ITS United Kingdom Members' Conference, IET*, pp. 1–7, 2008.

[7] S. M. Mahmud, S. Shanker, and I. Hossain, "Secure software upload in an intelligent vehicle via wireless communication links," in *Intelligent Vehicles Symposium. Proceedings. IEEE*, June 2005, pp. 588–593.

[8] N. Gabe, "Over-the-air updates on varied paths," *Automotive News*, 2016-01-25.

[9] D. T. Tuan, S. Sakata, and N. Komuro, "Priority and admission control for assuring quality of I2V emergency services in VANETs integrated with Wireless LAN Mesh Networks," *ICCE 2012*, pp. 91–96, 2012.

[10] S. Chakraborty and S. Nandi, "IEEE 802.11s mesh backbone for vehicular communication: Fairness and throughput," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 5, pp. 2193–2203, 2013.

[11] W. K. Tan, S.-G. Lee, J. H. Lam, and S.-M. Yoo, "A security analysis of the 802.11s wireless mesh network routing protocol and its secure routing protocols," *Sensors*, vol. 13, no. 9, p. 11553, 2013.

[12] M. S. Islam, M. A. Hamid, and C. S. Hong, "SHWMP: A Secure Hybrid Wireless Mesh Protocol for IEEE 802.11s Wireless Mesh Networks," in *Transactions on Computational Science VI*. Springer Berlin Heidelberg, 2009, pp. 95–114.

[13] J. Ben-Othman and Y. I. Saavedra Benitez, "IBC-HWMP: a novel secure identity-based cryptography-based scheme for Hybrid Wireless Mesh Protocol for IEEE 802.11s," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 5, pp. 686–700, 2013.

[14] M. Sbeiti, A. Wolff, C. Wietfeld, and I. Technology, "PASER: Position Aware Secure and Efficient Route Discovery Protocol for Wireless Mesh Networks," in *International Conference on Emerging Security Information, Systems and Technologies - SECURWARE*, 2011.

[15] M. Sbeiti and C. Wietfeld, "One stone two birds: On the security and routing in wireless mesh networks," in *Wireless Communications and Networking Conference (WCNC), 2014 IEEE*, April 2014.

[16] M. Steger, M. Karner, J. Hillebrand, W. Rom, and K. Romer, "A Security Metric for Structured Security Analysis of Cyber-Physical Systems Supporting SAE J3061," in *CPSData – Second International Workshop on modeling, analysis and control of complex Cyber-Physical Systems*, 2016, pp. 1–8.

[17] SAE, "SAE J3061: Surface Vehicle Recommended Practive - Cybersecurity Guidebook for Cyber-Physical Vehicle Systems," SAE International, Tech. Rep., 2016.

[18] ISO, "Road vehicles Unified diagnostic services (UDS) Specification and requirements," ISO 2006, Tech. Rep., 2006.

[19] W. Rom, P. Priller, J. Koivusaari, M. Komi, R. Robles, L. Dominguez, J. Rivilla, and W. Van Driel, "DEWI – Wirelessly into the Future," in *2015 Euromicro Conference on Digital System Design (DSD)*, Aug 2015, pp. 730–739.

[20] G. R. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke, "IEEE 802.11s: The WLAN mesh standard," *IEEE Wireless Communications*, pp. 154–160, 2010.

[21] "open80211s – An open-source implementation of the recently ratified IEEE 802.11s wireless mesh standard," http://open80211s.org/open80211s/.

[22] B. Potter, "Microsoft SDL threat modelling tool," *Network Security*, pp. 15–18, 2009.

[23] I. Garitano, S. Fayyad, and J. Noll, "Multi-metrics approach for security, privacy and dependability in embedded systems," *Wirel. Pers. Commun.*, vol. 81, no. 4, pp. 1359–1376, Apr. 2015.

[24] J. Noll, I. Garitano, S. Fayyad, E. Asberg, and H. Abie&, "Measurable security, privacy and dependability in smart grids," *Journal of Cyber Security*, vol. 3, pp. 371–398, 2015.

[25] D. Harkins, "Simultaneous authentication of equals: A secure, password-based key exchange for mesh networks," in *Second International Conference on Sensor Technologies and Applications. SENSORCOMM '08.*, Aug 2008, pp. 839–844.

[26] M. Steger, C. Boano, M. Karner, J. Hillebrand, W. Rom, and K. Romer, "SecUp: Secure and Efficient Wireless Software Updates for Vehicles," in *Under Submission*, 2016, pp. 1–8.