Fraunhofer

HIGH-PERFORMANCE CENTER CONNECTED SECURE SYSTEMS

A RAPID INNOVATION FRAMEWORK FOR CONNECTED MOBILITY APPLICATIONS

Fraunhofer AISEC, EMFT & ESK

High-Performance Center Connected Secure Systems

Fraunhofer Research Institution for Applied and Integrated Security
www.aisec.fraunhofer.de

Fraunhofer AISEC
Parkring 4
85748 Garching (near Munich)
Germany

Contact: Sascha Wessel
sascha.wessel@aisec.fraunhofer.de

Fraunhofer Research Institution for Microsystems and Solid State Technologies EMFT
www.emft.fraunhofer.de

Fraunhofer EMFT
Hansastraße 27d
80686 Munich
Germany

Contact: Franz Wenninger
franz.wenninger@emft.fraunhofer.de

Fraunhofer Institute for Embedded Systems and Communication Technologies ESK
www.esk.fraunhofer.de

Fraunhofer ESK
Hansastraße 32
80686 Munich
Germany

Contact: Dominique Seydel
dominique.seydel@esk.fraunhofer.de

CONTENTS

1 MANAGEMENT SUMMARY

Connected Mobility Applications help to continuously improve traffic safety and efficiency. Today, much time and effort have to be invested to bring an idea into a safe prototype and to finally launch a reliable product.

REQUIREMENTS ON DEVELOPMENT TOOLS

Software development tools have to adapt to these requirements. They have to support a rapid and continuous development process, that allows to test and validate the distributed application as one overall system. When developing cooperative applications, a higher design complexity has to be handled, as components are distributed over heterogeneous systems that interact with a varying timing behavior and less data confidence. Also, test and validation become more complex.

Our Innovation Framework is intended to rapidly bring an idea for a connected application into a prototype so the investment risk for innovative applications is reduced.

FIELDS OF EXPERTISE

Generally, we provide technical knowledge at the highest levels of science and technology, especially wide-ranging and vendor-neutral expertise in the area of safety and security covering hardware, embedded systems software as well as network technologies and connected applications. We provide state-of-the-art laboratories and analysis methods for verifiable product quality through security assessments.

RAPID APPLICATION DEVELOPMENT

In this whitepaper we describe the approach of a Rapid Innovation Tool Kit that is intended to speed up the development process for connected mobility applications. Thereby, a safe and secure prototype is available at an early development phase to gain experience within field tests that help to rapidly improve the intended application. Our software tool kit is able to find deviations from the specified behaviour and also it can instantly locate and identify erroneous functions within distributed systems. Extensive security tests can then be applied on the implemented application to ensure a secure operation.

COMMUNICATION TECHNOLOGY EVALUATION

Another use case for the described testbed is to evaluate communication technologies and to find the most suitable transmission technology for a certain application. For example, short range communication with the 802.11p WLAN technology or the upcoming LTE enhancement LTE-V2X are comparable within specific scenarios. This evaluation can help to reduce the investment risk for the deployment of connected applications.

CONTACT

Are you interested in collaborating with us? Please feel free to contact us for further information!

Dominique Seydel
+49 89 547088-363
dominique.seydel@esk.fraunhofer.de

2 INTRODUCTION

Connected Mobility Applications

For the deployment of future Intelligent Transport Systems (ITS) upcoming Vehicle-to-X (V2X) communication technologies and architectures, as ITS-G5 and 5G, enable the interconnection of vehicles, smart city infrastructure and backend platforms. The interaction of these **heterogeneous systems** to realize a certain application implicates new requirements on the development environment. Starting from early applications, where vehicles can automatically exchange awareness and event messages, up to smart lighting or cooperative enhancements of highly automated driving, a **secured development process** becomes inevitable.

New Application Architectures

System architectures for smart mobility applications will evolve from self contained systems to so called „Plastic Architectures", which are **highly dynamic** and **hardly predictable** at development stage. Thus, future applications will consist of fragments that are distributed over heterogeneous system components, e.g. other vehicles, edge components, transmission infrastructure, and backend services. **Complex interactions** that are necessary to realize the cooperative behavior have to be modelled and validated at an early development stage to detect and eliminate specification incompleteness or implementation failures. Current software methods and development tools are not applicable to design, simulate and validate those cross-systems applications. Moreover, during testing phase it is inevitable to have test data available for every involved component that form a **consistent and realistic test scenario** for the overall application.

V2X Communication Characteristics

The capability of connected applications is strongly affected by characteristics of the current **communication channel**, like Quality of Service (QoS) parameters, a varying reception area and the accuracy of the received information's position. To develop a reliable application, the realistic behavior of these characteristic has to be considered at an early development stage. On sensor level, the self-characterization of V2X communication as a sensor has to be determined. On application level, **state-of-the-art software methods**, e.g. graceful degradation strategies, have to be incorporated into the application model to adapt to the **varying reliability** during runtime. To evaluate the applications performance, much effort has to be spend on test hardware to reach a sufficiently high number of real road users. Moreover, V2X is an interface to other **potentially unreliable systems** and demands high security to ensure a **reliable and safe operation**. Next to security issues, privacy concerns emerge from interconnected V2X applications that constantly exchange personal information and consumer-specific meta data. An integrated simulation and testing environment, as depicted in Figure 2.1 and described in this paper, is an adequate tool to find answers to all of these questions described above.

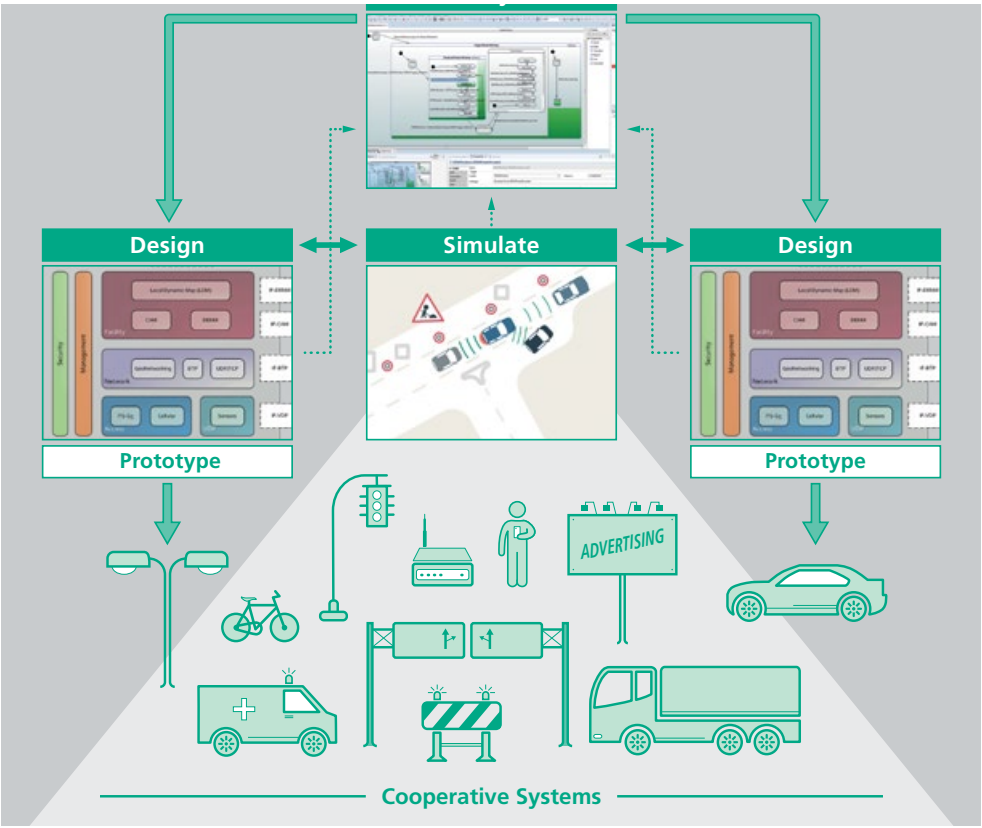


Figure 2.1: A Rapid Innovation Tool Kit for Cooperative Systems

A **secured development process** for distributed applications is required to provide validated functionality, security and privacy. In this white paper, we therefore present a **rapid innovation tool kit** as an enhanced testbed for supporting the secured development process of connected automotive systems in all stages of the development life cycle. With this testbed we want to enable our customers to **speed up** their **innovation process** by rapidly developing and refining innovative concepts and thereby reduce the time-to-market for their innovative features.

The white paper is structured as follows: In Chapter 2, we introduce the testbed by giving an overview first, explaining its individual components in detail subsequently. In Chapter 3, we present our fields of expertise and services within the V2X domain and the offerings to our customers.

3 RAPID INNOVATION TOOL KIT

In this chapter, we present an **extended testbed** which combines rapid application development, an integrated simulation environment for network simulation and also for microscopic and large scale traffic simulations as well as functional validation and security testing of the applications.

Testbed Features

One additional part of the testbed is an **infrastructure sensor kit** for the detection of road conditions and traffic monitoring that is integrated via generic interfaces and can be used for the simulation or field tests of innovative mobility applications.

Furthermore, a synchronization concept is implemented into the testbed to enable **hybrid testing**, where several testing participants can be simulated as virtual road users and other participants of the testing scenario can be implemented in real hardware.

The testbed is also fully integrated into a software development process that complies with **continuous integration** and provides secure deployment and **update** mechanisms. A summary of our services for application development using the testbed is depicted in Figure 3.1 and described within the following subchapters.

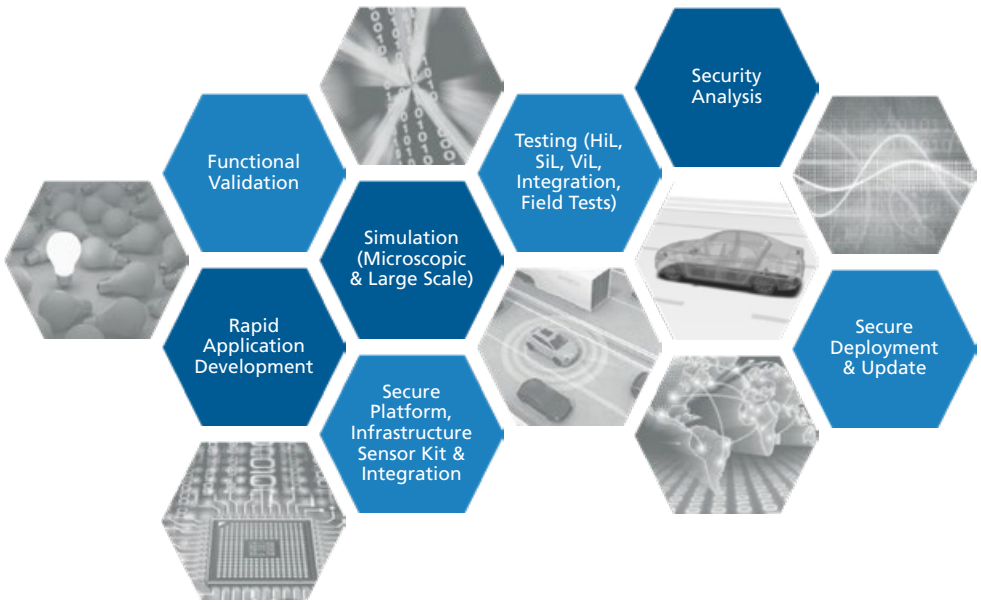


Figure 3.1: Our Services for Application Development

3.1 Application Development Cycle

The testbed combines several aspects of the V2X application development life cycle. It covers **development steps** beginning from application design, continuous integration into a simulation environment, testing environments, tools performing functional and security analyses up to a secure deployment and update process. The **application development flow** when using the innovation and testbed is shown in Figure 3.2. It is also designed to only use particular aspects when developing innovative applications.

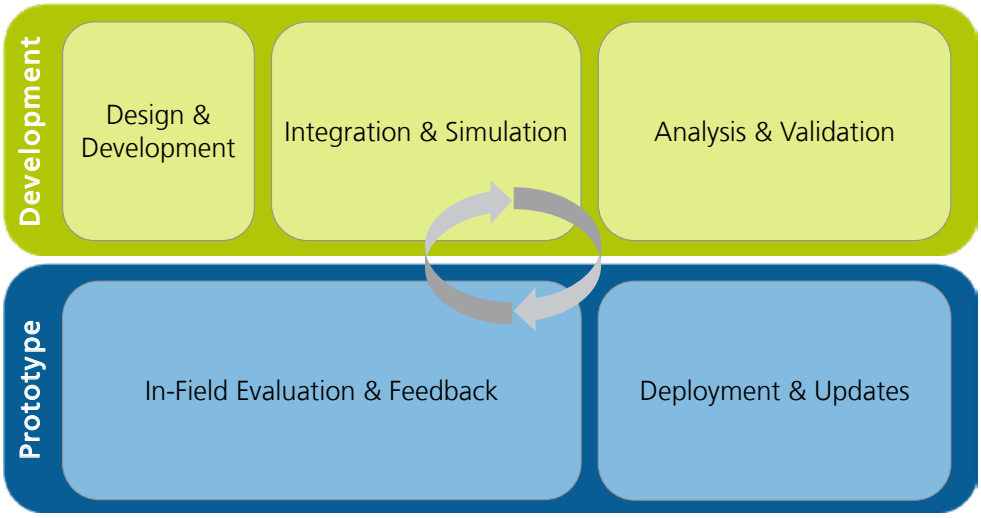


Figure 3.2: Application Development Cycle using the Testbed Services

Application Development Flow

An exemplary flow of the development process for an application is shown in Figure 3.3. The intended application can be realized as implemented source code or as a software model to be further developed and optimized within the testbed.

The testing layer consists of several testing methods that are specific for each step in the development process and include several simulation environments, integration testing and field tests. The testbed provides the ezCar2x® framework that allows testing connected applications within a simulation environment, using network and traffic simulation as well as integration of hardware-in-the-loop, e. g. roadside units (RSUs). Another main feature of the testbed is a **combined simulation and field test** approach, where virtual and real traffic participants can be tested in a synchronized environment.

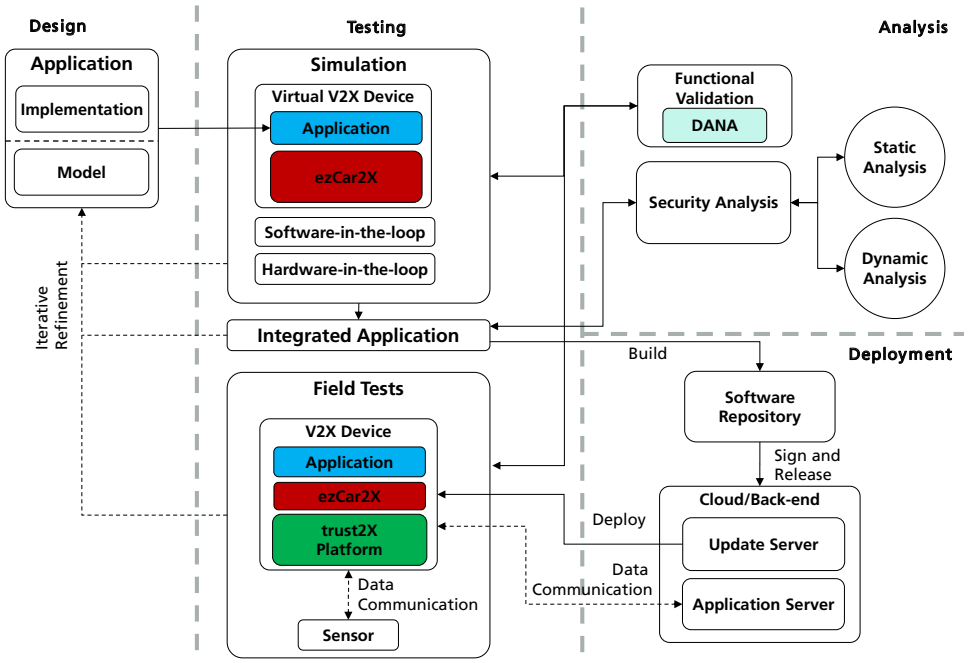
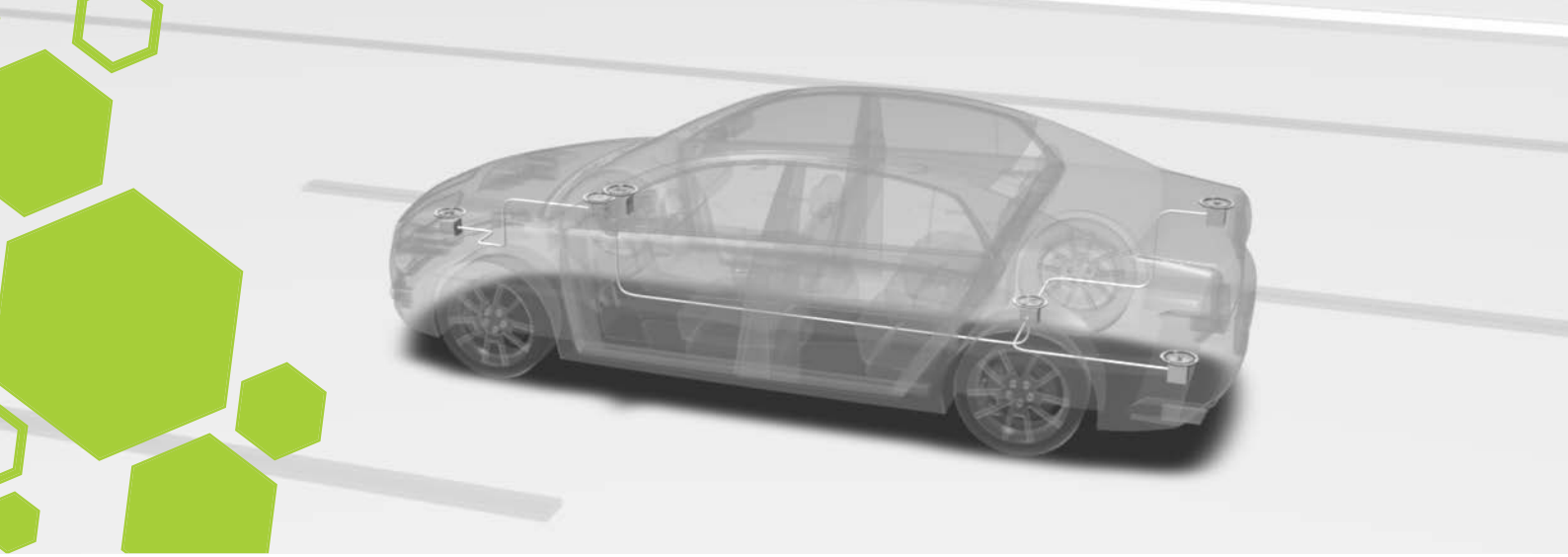


Figure 3.3: Exemplary Development Flow

Additional **analysis tools** can be applied on the developed application. On the one hand our Description and Analysis of Networked Applications (DANA) tool for **functional validation** can continuously be used in every testing step. For the integration testing of the application the analysis toolbox provides application security testing methods in order to detect **software vulnerabilities** in an early development stage. Using static application security testing (SAST) as well as dynamic application security testing (DAST) allows quick analysis during integration testing in order to detect potential software vulnerabilities.

Finally, the application is built and subsequently **signed** by the software repository and pushed to the update server which is part of the backend. The update server again signs the application and **deploys** it to V2X devices.

For V2X devices we provide trust2X, a hardened platform that includes hardware- and software-based security features in order to isolate and protect processes and data of ezCar2x® applications. In particular, this allows applications running within one container to be **protected against compromised or malfunctioning software** running in another container on the same system. Furthermore, the trust2X platform offers runtime integrity of containers and the kernel, container-restricted access to hardware resources, data confidentiality through



container-based RAM encryption, a secure update mechanism for the deployment of software and remote device management capabilities. Finally, the modularized software architecture of trust2X enables customization with respect to specific use cases such that a customer may integrate further software- and hardware-based security features using trust2X as a building block.

3.2 Application Design

Bring your Application

The application that should be further developed or improved within the testbed can exist in different ways:

- An idea or concept that should be modelled using established modelling tools, as Matlab Simulink or Automotive Data and Time-Triggered Framework (ADTF).
- A source code implementation, e.g. implemented in C++.
- An application model which is used to generate source code.

The implemented application is then **integrated into the testbed** by enhancing the interfaces for the use of the tool kit. An application model is continuously integrated into the testbed by using existing wrapper code for the ezCar2x® interfaces.

State-of-the-Art Safety Mechanisms

If safety mechanisms are required for the intended application, state-of-the-art software methods, e.g. **graceful degradation** strategies, can be incorporated into the application model within this development step. For example, a connected application with safety-critical functionality, as platooning, depends on QoS parameters of the communication link. Our safety function for resilient control uses these QoS parameters, such as the current Packet Loss Rate (PLR), to decide which degradation mode is sufficient, e.g. readjusting the distance to the vehicle ahead. The safety mechanisms for **resilient control** can be integrated into common automotive software architectures, as AUTOSAR, AUTOSAR Adaptive and further concepts. Also existing architectures, as developed by the **GENIVI** Alliance, can be integrated to handle the unreliability of the communication link.

Including New Architecture Concepts

Another aspect that is getting more relevant for application design and validation are **plastic architectures**. Parts of an application can be distributed over several participants, e.g. for a collision warning application the interaction of the originating vehicle, the warning notification and optionally edge or cloud components form the overall function. Also, parts of the appli-

cation can be **dynamically relocated during runtime**, e.g. from cloud to edge components. Thereby, the system boundaries dynamically change depending on the current communication relations and these runtime conditions have already to be covered within the design phase.

3.3 Simulation and Virtual Testing

Although, information received via V2X communication has a varying confidence, it can **complement the coverage area** of current automotive sensors by taking advantage of the different characteristics of electromagnetic signal propagation. When it comes to cooperative applications, as consensus decisions between traffic participants, V2X is a convenient sensor. Having this described applications in mind, V2X systems will perform various challenging and sometimes safety-critical tasks. These systems impose high requirements on safety and security, as its failures can endanger human life.

Simulation Purpose

One of the goals of simulation and testing is to achieve a **fail-operational behavior** of the application even when the information is **unconfident**. Therefore, the coverage of the used **testcases** within the simulation environment and during virtual testing should be as **realistic and as comprehensive** as possible. This is reached by (automatically) defining reference scenarios and generate variations out of them, e.g. by stochastic variations.

One of the parameter variations is the realistic behavior of the communication channel during a certain driving scenario. Therefore, a network simulation tool, e.g. ns-3 or OMNeT++, and a traffic simulation tool, e.g. SUMO, VTD or CarMaker, are integrated into the simulation environment. Our testbed could also be integrated with other microscopic and macroscopic traffic simulation tools, as each of them has advantages when testing a certain connected application.

3.3.1 Integrated Simulation Environment

The simulation environment, shown in Figure 3.4 helps developers debugging their applications during early development stages with **(semi-)automated testing procedures**. It contains interconnected tools for simulating wireless communication networks, e.g. network nodes using ITS-G5 or LTE, as well as traffic simulation.

As already described in Chapter 3.1, the application implementation is deployed on each virtual V2X node within the network simulation environment. Together with the ezCar2x® framework

each virtual node can be equipped with the developed application and also with V2X communication ability so that its interaction with other nodes can be **simulated as realistic as possible**. The simulation can be executed in **real time**.

Functionality of the Simulation Framework

The interaction between all virtual nodes is realized using a **virtual wireless channel** considering the specific characteristics of each communication technology by using channel models, e.g. dedicated models for ITS-G5, LTE or 5G. This virtual wireless channel is also used to **integrate real hardware** into the simulation by utilizing a channel proxy and creating a mirror node for each hardware component within the network simulation.

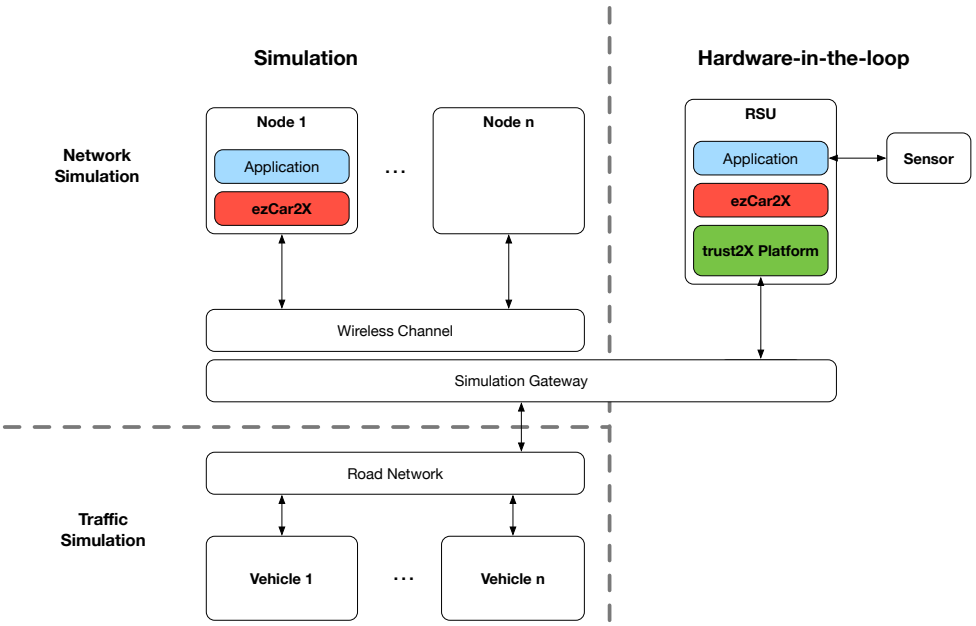


Figure 3.4: Simulation Environment

The **network simulation** is coupled with a macroscopic **traffic simulation** for large scale traffic scenarios, e.g. to test security mechanisms for V2X messages, and with a microscopic traffic simulation for scenarios with less participants, e.g. Cooperative Merging or Platooning. The coupling via control interface is needed to synchronize the behavior of communication nodes and traffic participants in each simulation for the given driving scenario.

Combined Simulation and Real Hardware

The environment can be extended to **hybrid simulation** by including hardware-in-the-loop. As

shown in Figure 3.4, a RSU comprising an application, e.g. Smart Lighting, can be integrated into the simulation loop, by connecting it to the wireless channel interface of the simulation environment. The RSU can again interact with further communication hardware, e.g. test vehicles that are in communication range. The RSU can also be connected with sensors, that are integrated into the testing environment and that can give status data to generate event messages, e.g. Decentralized Environmental Notification Messages (DENMs).

3.3.2 ezCar2x® V2X Communication Framework

The flexible ezCar2x® software framework provides the key components needed to rapidly create prototype applications for networking vehicles. This allows vehicle manufacturers, suppliers and road infrastructure providers the opportunity to rapidly implement new concepts and evaluate them in a real world setting.

The ezCar2x® framework features components for the following functions:

- ETSI ITS-compliant communication services
- GeoNetworking and IPv6-over-GeoNetworking via ITS-G5 and LTE
- Adaptive hybrid network layer with optimal network selection in real time
- CAN bus system connectivity, including quick vehicle integration
- Connectivity to external sensors such as GPS, radar or lidar
- Integration of digital map data
- Facilities as CAM and DENM management and Local Dynamic Map
- Integration of security services
- Connectivity to various HMI devices such as tablets or touch screens

Framework Characteristics

The framework is structured as a collection of C++ libraries, as depicted in Figure 3.5, that are available for Linux and Windows operating systems. Extensive API documentation and various examples help accelerate the learning curve for first-time users. With its flexible, **modular and portable design**, the framework can be deployed in a **variety of environments** such, e.g. in common onboard and roadside units, on security platforms like trust2X, as backend server components or directly in simulation environments.

The consistent utilization of abstraction concepts and known design patterns leads to a high degree of **modularity**. Single functions can be swapped out and if needed, replaced by other implementations without modifying the remaining systems. External platform-specific dependencies were minimized to allow the framework to be used on **different target platforms**.

The high degree of abstraction means that hardware-dependent functions, such as direct access to the wireless modules, can be easily adapted. Even with the achieved portability, performance is not impacted since the framework has access to native compilers and platform-specific optimizations. With the integration of ezCar2x® into the trust2X platform, services directly exchanging data with potentially unknown vehicles can be isolated and separated from sensitive business logic and cloud services.

With its modular design, the ezCar2x® framework can be **incorporated directly into a simulation** environment consisting of a road traffic and network simulator. As a result, it can be installed in an unlimited number of virtual vehicles. This allows developers to test and continually enhance V2X applications and protocols under realistic conditions with minimal effort. The same source code can then be used directly on the embedded target platform without modifying it, thus **eliminating additional development effort** and preventing new errors introduced by the reimplementation.

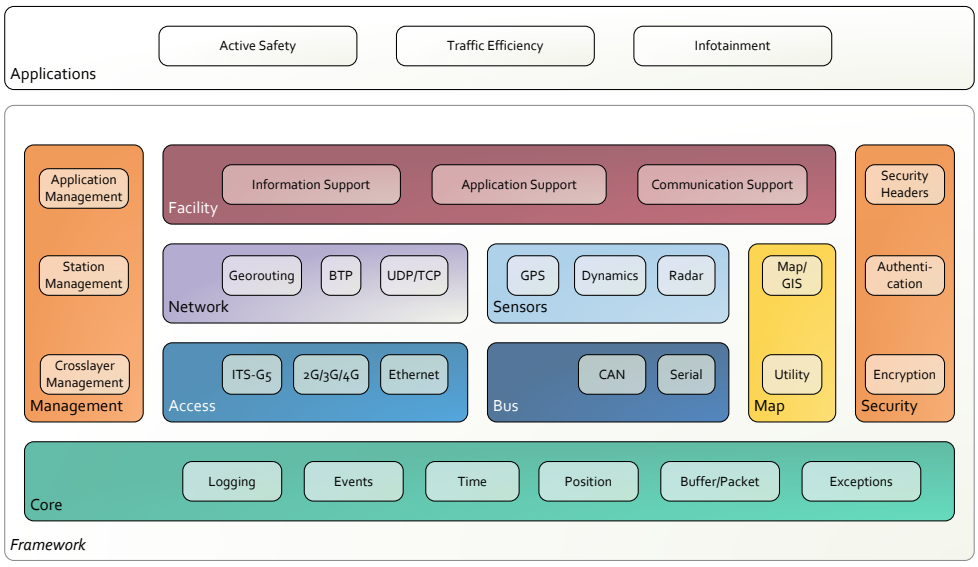


Figure 3.5: Modular Architecture of the ezCar2x® Framework

3.4 Software Analysis

In each step of the development process it is applicable to perform additional analyses to get detailed **knowledge of the overall system** and the applications behavior for debugging, monitoring, security, and validation purposes. In this chapter, we give an overview on our methods and tools for software analysis.

3.4.1 Monitoring and Functional Validation

For software validation and verification, model-based techniques are advantageous during the design and integration phase. Our DANA platform, an open and modular environment based on Eclipse, is a tool meant for specifying and analyzing networked applications. For this purpose, the **valid behavior** of the application is described as a layered reference model. This model provides a basis for further Model-based development steps. On the one hand, it can be used for various transformations of behavior models, e.g., for generating test cases or code for running simulations. On the other hand, it can be used for static analyses to check **conformance to modeling guidelines**, metrics for interfaces, and the compatibility of behavior models. The model-based approach also allows a quick integration of new sources for messages. Furthermore, the DANA tool can be used for the **verification and validation** of software interface behavior as messages between these interfaces can contain complex data.

In our proposed testbed we use DANA as a **central monitoring tool** to have all the status, debug and behavior information, the error messages and timing data centrally available from each component. This aggregation helps to **simplify and to speed up** the debugging process during development and during runtime. Further validation checks can also be applied on this collected data, as described in the previous paragraph.

3.4.2 Application Security Testing

Next to functional testing and simulation as described in 3.3, the extended V2X testbed includes application security testing. By integrating static as well as dynamic testing methods into the development process, applications can be tested against a **broad spectrum of software vulnerabilities**. Detecting vulnerabilities in early development stages is crucial, as it prevents the necessity of expensive software patching post-release. Therefore, we propose a **work flow**, where application security testing is part of continuous integration.

In the following Chapters, we describe the application security testing methods and how they are included in continuous integration.

3.4.3 Static Application Security Testing

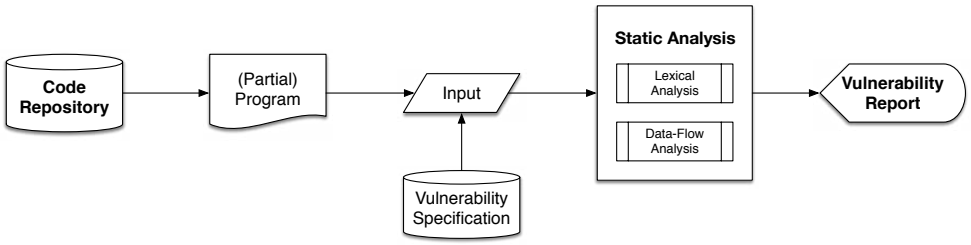


Figure 3.6: Static Analysis

Static code analysis as part of SAST allows inspecting program behavior without actually running the program. A large amount of tools for programming languages that are typically used within the V2X domain are available for detection of **software requirements being violated**, e. g. erroneous program behavior, reachability or dead code. At that, many tools are specifically designed to **detect program flaws** that lead to potential vulnerabilities, e. g. buffer overflows.

As shown in Figure 3.6, SAST depends on the application source code and vulnerability specification as input, where the latter defines what kind of vulnerabilities should be detected by the tool. The static analysis then runs fully automated and **outputs a report** of the detected potential vulnerabilities.

Applying SAST on Test Applications

Available tools can be applied directly on source code as well as on binaries and bytecode. The applied methods differ in efficiency and effectiveness depending on the underlying approach. Broadly, applied methods can be divided into lexical scanning and data or control flow analysis.

Lexical source code scanners are very efficient in terms of processing time and can also be applied on partial programs, e. g. single commits to the software repository. Therefore, these tools are helpful if real-time detection during development is required or for fast analysis before builds.

Optimized Detection Process

In order to achieve **higher effectiveness** in vulnerability detection data and control flow analysis is applied, e. g. static taint analysis or symbolic execution. These methods provide **higher sensitivity** and can therefore detect a larger scope of software vulnerabilities. However, these methods are applied on intermediate representations of the software product, which have a tree- or graph-like structure. Creating and analyzing these structures is mostly more costly in time than pure lexical scanning. However, these methods can still be part of continuous integration, e. g. within nightly builds.

3.4.4 Dynamic Application Security Testing

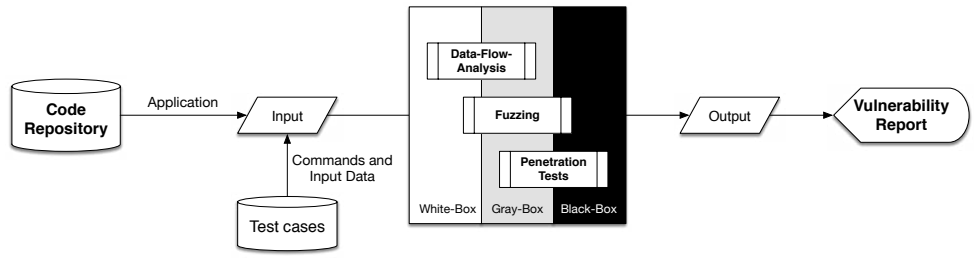


Figure 3.7: Dynamic Analysis

From SAST to DAST

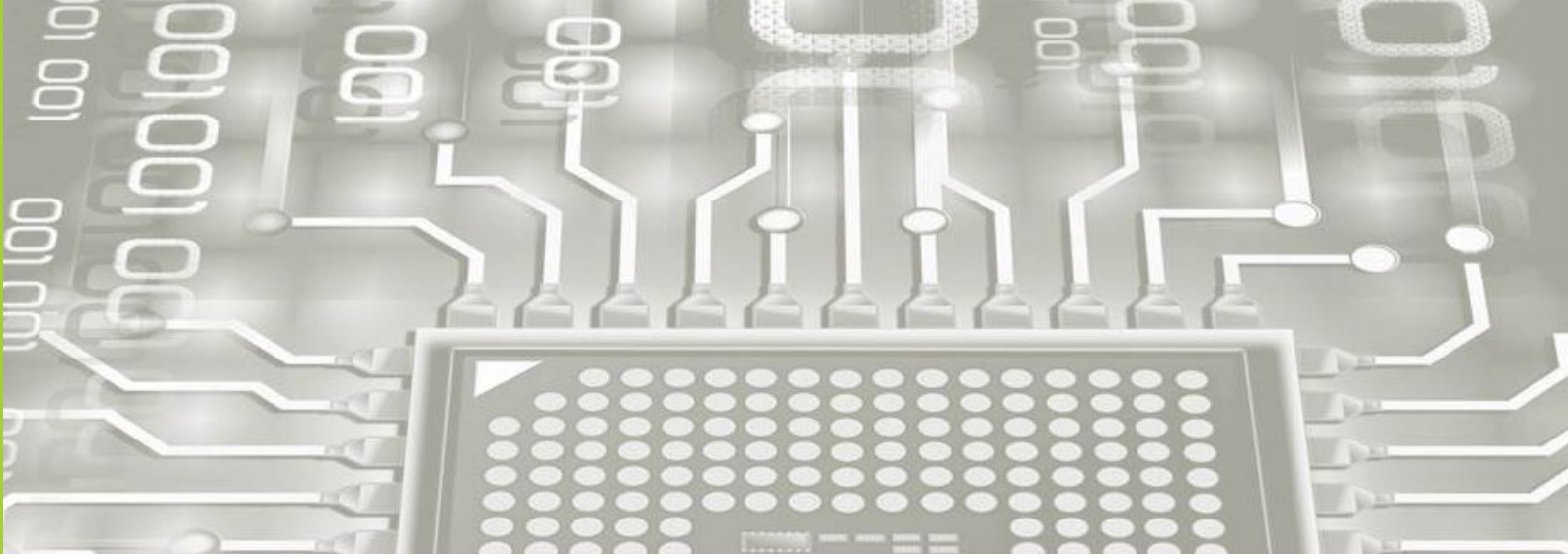
While SAST has shortcomings, e. g. detection of vulnerabilities in authentication, access control or cryptographic protocolls, as well as uncovering flaws in the security design, DAST can tackle these problems. Furthermore, it provides **deeper analysis** and can **imitate the attack scenarios**. Since applications are executed within this testing method, DAST depends on predefined input data. These inputs either represent specific test cases or are randomly chosen.

In this chapter, we present different methods of DAST, which can be applied in the V2X scenario.

Dynamic Code Analysis

Dynamic code analysis is a **white-box testing** approach. This means that the internal structure of an application, e. g. source code or intermediate representation, is available and can be leveraged for analyzing the application.

Monitoring of function calls by hooking into security critical functions, e. g. system calls, is one typical testing method. In addition, function **parameter analysis** is applied, inspecting the input-output relations of function calls. More sophisticated methods like **dynamic taint**



analysis are able to analyze execution paths an attacker may use to exploit an application. This method can be applied even if the source code of an application is not available.

All of these methods aim at detecting potential vulnerabilities that occur during runtime of the application. This allows verifying alarms that have been reported by SAST. Furthermore, it can be used as preparation for **penetration testing**, as it provides potential entry points for actual attacks.

Vulnerability Scanner

Vulnerability scanners provide more consolidated tools that allow automated detection of common or known vulnerabilities. Therefore, they have a restricted scope and are less flexible, but can be applied black-boxed, meaning without knowledge about the internal structure of the program. In general, scanners achieve this by using standardized input data for public interfaces of the application. By analyzing the output data, the tool reports potential vulnerabilities.

Penetration Testing

In penetration testing it is assumed that no information about the internal structure of the application is known. Although, mostly little information about the overall functionality of the application is given. In general, the behavior of the application is approximated by interacting only with public interfaces of the application. The goal is to detect and exploit vulnerabilities that lead to privilege escalation.

Fuzzing

Fuzzing aims at complete functional coverage of the given code in order to detect vulnerabilities. This is generally approximated by iteratively generating and analyzing input, state change and output data. Fuzzers can run in a white-, gray- and black-box scenario.

In the black-box scenario, the fuzzer generates random input data in order to **trigger new internal states** in the targeted application. This approach has the **highest efficiency** in terms of processing time but the lowest code coverage (effectiveness). This means that some possible internal states are not reached by the fuzzer and therefore potential vulnerabilities might not be detected.

In the white-box approach the internal structure of the application is known and can be leveraged to **detect reachable states**. For instance static code analysis can be applied to explore different execution paths.

Gray-box fuzzers, e. g. american fuzzy lop (AFL), apply instrumentation to **gather information about the code**. The instrumentation informs the fuzzer about the increased code coverage, such that the fuzzer knows what states already have been covered. This approach has high effectiveness in vulnerability detection and also provides reasonable efficiency in terms of processing time.

3.5 Deployment and Prototyping

To rapidly transfer the developed application into a prototype, as needed for field tests, an **integrated deployment process** is beneficial. As described in the following chapters the test-bed offers an integrated solution containing a secure V2X platform, a continuous integration workflow, secure deployment and update processes and a communication link to backend or cloud platforms.

3.5.1 trust2X Hardened V2X Platform

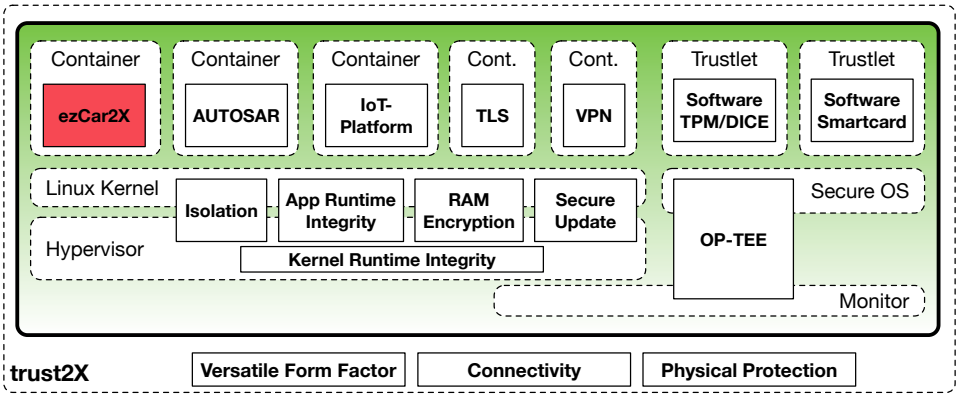


Figure 3.8: trust2X Hardened V2X Platform

The hardened V2X platform trust2X presented in Figure 3.8 is a hardened Linux-based operating system that provides **hardware- and software-based security features** for target devices of V2X applications. Its main goal is the secure isolation of different software entities that run concurrently on top of a single Linux kernel. As shown in Figure 3.8, ezCar2x® can run in a Linux container, isolated from other guest systems (e. g. AUTOSAR) and other functional modules or services (e. g. TLS, VPN). By isolating software entities, **each entity is protected** against compromised or malfunctioning software that runs in a different container. Therefore,

safety- and security-critical functions can run within an isolated environment, unaffected by failure or compromise of separate entities in the system.

Furthermore, trust2X ensures **integrity** of each container and the kernel at runtime. Next to container-restricted access of hardware resources, RAM encryption is applied to provide data confidentiality container-wise.

Its modularized software architecture allows customization with respect to specific use cases. The customer can integrate further software- or hardware-based security features using trust2X as a building block.

trust2X comes with own mechanisms for software deployment in the wild. This includes a **secure update mechanism** for the trust2X isolation engine and for the Linux kernel. Furthermore, it provides remote device management via a dedicated backend.

3.5.2 Continuous Integration

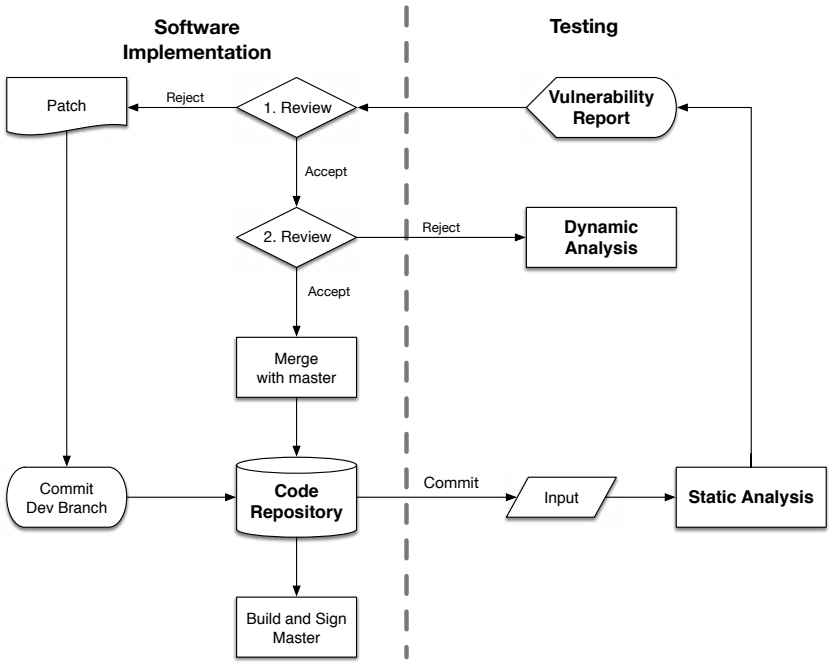


Figure 3.9: Continuous Integration: Static Analysis

Application security testing as part of the testing layer depicted in Figure 3.3 complies with continuous integration. Figure 3.9 and 3.10 propose a typical workflow for continuously integrating security testing in the development process.

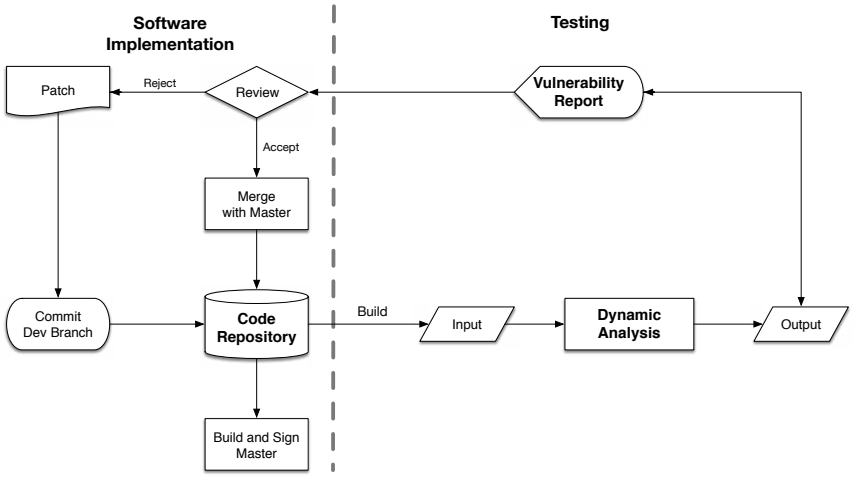


Figure 3.10: Continuous Integration: Dynamic Analysis

Continuously Integrated Security Tests

Each commit to the central software repository automatically triggers a build of the application under development. While each build is self-testing SAST and DAST is triggered automatically. Static security testing tools are applied either directly on source code, or on compilation products such as bytecode or binaries. Therefore, either **single commits can be tested** using light-weight analysis tools, as well as individual software modules or the complete application. Static application testing tools can run without user interaction and provide a report that lists each finding of potential vulnerabilities. DAST tools might require user interaction depending on the applied testing method, as depicted in Figure 3.7.

In case a potential vulnerability has been found the continuous integration server **alerts** the development team. Based on the review of the **testing reports** either further security tests can be applied, or the developer commits a patch in order to eradicate the vulnerability.

If the application has been successfully tested and no potential vulnerabilities were found, the continuous integration server triggers a merge request, or directly merges with master. Subsequently, the application is build and signed.

3.5.3 Software Deployment and Secure Update

The deployment and secure update mechanisms for both, trust2X and ezCar2x® are based upon a public key infrastructure (PKI), as depicted in Figure 3.11. The main purpose of the PKI is **trust establishment** between the update server and the V2X devices via authentication based on cryptographically signed certificates. At that, the PKI can **restrict authorization** for software deployment and update to specified roles within the supplier chain. Authorization for updating different targets on the same platform depends on the provided certificate chain that signed the update. For instance, updating a specific target requires the update to be signed by a certain subordinate certificate authority (CA).

Furthermore, cryptographic signing metadata, such as the binary hash and version, provides **integrity of the updated software** and protects against common man-in-the-middle (MITM) attacks on updates over-the-air (OTA), e. g. replaying or rolling back to older and insecure software updates.

By using hardware-based security, such as trusted platform modules (TPMs) or hardware security modules (HSMs), trust2X provides means for **tamper-resistant trust anchors** for secure update mechanisms.

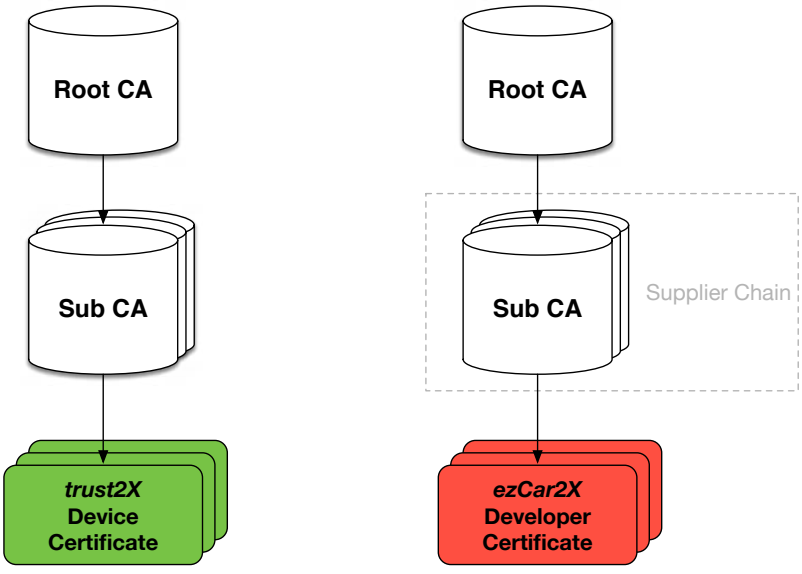


Figure 3.11: Software Deployment

The PKI also contains implicit and explicit **revocation mechanisms** to protect against compromised software signing keys. This means certificates are only valid within a defined time interval. At that, integration of an Online Certificate Status Protocol (OCSP) server allows explicit key revocation and means for verifying whether a signing key has been revoked. With the help of key exchange protocols revoked keys can securely be replaced.

3.5.4 Backend and Cloud

The deployment layer depicted in Figure 3.11 provides a backend that contains the update server and an application server. The update server is responsible for deploying and updating the software, as described in 3.5.3.

The application server provides services that are used by ezCar2x® applications. In order to provide a secure communication channel, Transport Layer Security (TLS)-based communication or a virtual private network (VPN) is applied. Furthermore, the application server is responsible for the trust2X device management.

3.5.5 Sensor Integration

The effectiveness of a connected applications simulation depends on how realistic input data can form a certain driving scenario. Input data from distributed sources, e.g. status data within Cooperative Awareness Messages (CAMs) from other vehicles or roadside sensors, have to be **synchronized during recording and replay phase**. Synchronization is required to provide the intended driving conditions for the application that are to be tested. To achieve realistic input data it is beneficial to have sensors integrated in an early development step as Hardware-in-the-Loop (HiL) or recorded input data stream.

The realistic sensor data can also be used to develop algorithms for sensor analysis, timing and improvement of machine learning processes. In addition, to develop tamper-proof algorithms it is advantageous to have a realistic behavior of sensor data available in order to avoid manipulation or attacks with sensor hardware or sensor data.

Within our simulation environment, vehicle sensors and a set of infrastructure sensors are integrated as components in ezCar2x® via **generic sensor interfaces**. When recording test data, the real sensors can easily be included into the synchronized recording process. The same setup can be used during field tests, where infrastructure sensors usually are integrated into RSUs and thereby provide their environment data.

3.5.6 Field Tests

As a last step of the development cycle the deployed application is **extensively tested** within field tests. For that purpose, our tests vehicles are equipped with the required test hardware, as communication units, antennas, a highly accurate positioning solution, additional radar sensors and automotive computers. The test vehicles are applicable for each application, that is to be tested within the testbed.

Additionally, there is communication hardware available to **equip further infrastructure** units, e.g. RSUs. Thereby, infrastructure based applications can, as Smart Lighting, can be implemented and validated with the testbed.

4 FIELDS OF EXPERTISE AND SERVICES

In this chapter, we present our fields of expertise, services and products for developing secure V2X systems.

Generally, we provide technical knowledge at the **highest levels of science and technology**, especially wide-ranging and vendor-neutral expertise in the area of safety and security covering hardware, embedded systems software as well as network technologies, services and applications. We provide **state-of-the-art laboratories and analysis methods** for verifiable product quality through security assessments.

With our own products trust2X, ezCar2x® and DANA we provide building blocks for a secure platform and application development, securing V2X systems as a whole.

In the following chapters, we describe our offerings for the safe development and secure deployment of V2X systems.

4.1 Rapid Innovation Development

All our services are intended to rapidly bring an idea for a connected application into a prototype so we can help our customers **decreasing their investment risk** for innovative applications. Therefore, we provide three possibilities for cooperation during the development process, that are outlined in Figure 4.1.

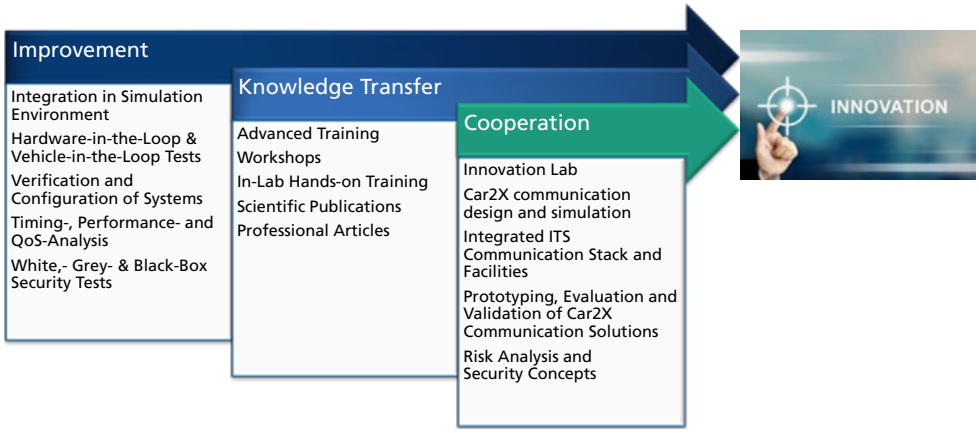


Figure 4.1: Our Services for Rapid Innovation Development

Collaboration in Application Implementation

First of all, we offer to collaborate with our customers in implementing the connected application together. For this purpose, we provide an **Innovation Lab**, where working stations with required development and simulation tools and also communication hardware are ready to use. With our long lasting expertise in V2X communication systems we can help to design and simulate European Telecommunications Standards Institute (ETSI) ITS and 3GPP compliant V2X systems. The developed application can use our ezCar2x® communication stack for access, network, transport and facilities layer functionality. With this package, it is possible to **prototype, evaluate and validate** the intended application, as described in Chapter 3. Beyond that, we offer to create security concepts and a risk analysis for the intended application.

We Transfer Our Konwledge to Qualify Your Employees

The second field of our services for rapid innovation development is aimed at transferring the required knowledge to our customers so that they can independently implement their idea. Considering that, you can register for one of our courses or workshops from the well-established **program for advanced training**. We also offer **hands-on training sessions** in our Innovation Lab, that can be individually scheduled. Numerous of our scientific publications concerning the field of safety and security for connected applications are available at open access. We can also assist you to publish scientific findings that result from our collaborative work.

We Help to Improve Your Application

If there is already an implementation or model of the intended application available we can assist with its improvement. Therefore, our simulation environment is intended to **easily integrate external applications**, as described in Chapter 3.3.1. The outlined approach of hybrid simulation allows to combine the virtual simulation with real system tests, like HiL or Vehicle-in-the-Loop (ViL) tests. On the one hand, the aim of the advanced testing is to find a convenient configuration of the system and to verify the application. On the other hand, the testbed can be used to perform timing-, performance- and QoS-analyses. In addition, it is beneficial to incorporate security tests during the integration testing phase to eliminate security vulnerabilities as early as possible.

Specific Research Building Blocks

With our research expertise in the area of V2X systems, we can additionally offer specific building blocks, that are outlined in Figure 4.2. An infrastructure sensor kit and corresponding Sensor Integration solutions are available as well as a secure V2X hardware platform, as indicated in Chapter 4.2. Our communication framework ezCar2x® implements current ETSI ITS Standards that we complement with algorithms resolving current research questions, like

Adaptive Communication Technology Selection and Unified GeoNetworking. There are further approaches and research results available that concentrate on Distributed Consensus algorithms, a trustworthy Virtual Sensor Integration, the Scalability of scenarios, adequate System Architectures, and also **Self-Awareness and Runtime Models** for an adaptive system behavior. These building blocks form an **application abstraction service**, as they should be similar for every application. Comprehensive services, as detailed in Chapter 3, complete our innovation and testbed.

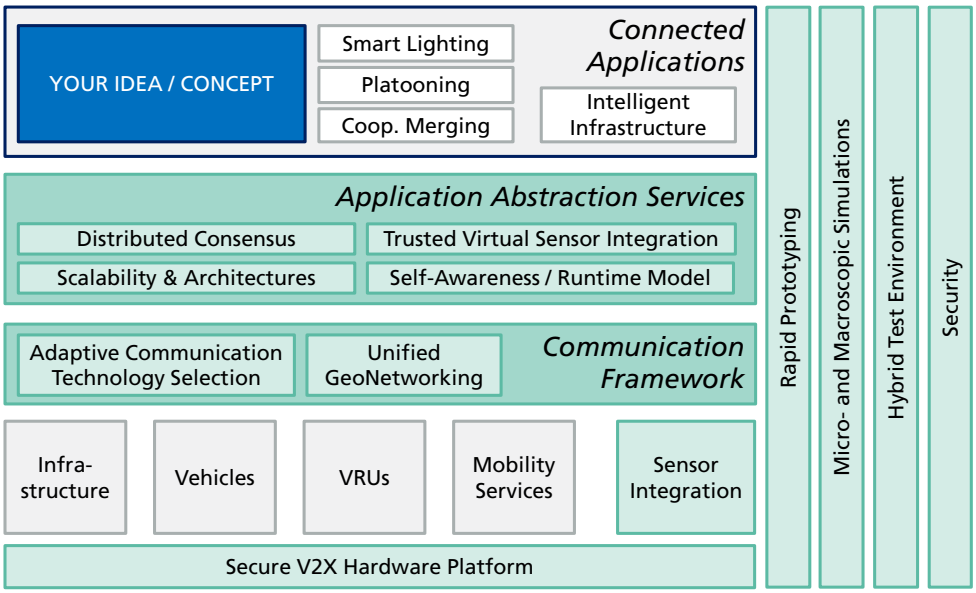


Figure 4.2: Building Blocks of our Research and Development Expertise

4.2 Secure V2X System as a Service

We offer secure solutions for V2X systems, including secure V2X applications that are deployed and run on a hardened V2X platform. By including multiple entities in the security design of the V2X system, e. g. platform, application, deployment and update mechanism, our solutions consider the V2X system as a whole and therefore promote **secure design principles** such as defense in depth and the principle of least privilege.

4.2.1 Hardened Platform

With trust2X we provide a hardened Linux-based platform for running V2X applications within a secure environment. Due to its portability feature, we offer porting our platform to other systems running a Linux kernel.

While trust2X has a comprehensive set of ready-to-use security features, it also provides building blocks for security enhancement and a modular system for customization. Therefore, the flexibility of trust2X allows adoption to specific functional or security requirements given by our customers.

Furthermore, we assist our customers by developing concepts which define **isolation and separation rules** for each entity within the system based on given security requirements and based on the underlying hardware of the system.

In case our customers want to use their own implementation of a hardened platform, we help with conception or security evaluation of given concepts.

trust2X comes with secure deployment and update functions which can be customized for specific use cases, requirements and supplier chains. We assist our customers in rolling out and testing PKIs for creating and **verifying trust boundaries** between backends and V2X devices.

4.2.2 Secure Application Development

Next to securing the platform, we help developing secure V2X applications that run on top of it, using the ezCar2x® application development framework. In addition, we offer consulting services and conception to support our customers in each development step. These include conceptual design, prototyping, security and privacy compliance, as well as application testing, with respect to functional and security properties of the application. We therefore provide solutions that combine the design, implementation and testing of functional and security requirements.

Conceptual Design

We support the development process of V2X applications from the early beginning. By developing conceptual designs, we help designing interconnected and secure V2X applications based on given requirements and use cases. Next to integrating required functional features, we also provide **security by design**. At that, we provide security and privacy evaluation of already developed concepts by the customer.

Prototyping

With ezCar2x® we can support testing of conceptual designs by implementing prototypes that can be tested within the ezCar2x® simulation environment. At that, we provide **analysis of vulnerabilities** by applying application security testing methods or developing test cases that can be performed by the customer within their own testing environment.

Compliance

Interconnected V2X applications often have to comply with certain **privacy regulations**. We help integrating these compliance requirements into the application design, such that privacy concerns are ruled out from the early beginning of the development process. At that, we can identify existing privacy issues within the application design or implementation given by the customer.

Application Security Testing

We provide consulting services for setting up a **continuous integration** environment that integrates V2X application security testing as described in Chapter 3.4.2. This includes evaluating potential tooling for performing security tests and proposing how these tools can be integrated into the development process.

Furthermore, we offer security testing on given software available as source code or binaries. By applying SASTs and DASTs, we provide a detailed **vulnerability report** including risk

assessment for each vulnerability. At that, we offer fuzzing of applications, which allows the detection of unknown vulnerability types. This supplements traditional **vulnerability scans** that search for predefined and therefore known types of vulnerabilities. With our expertise in static and dynamic program analysis and fuzzing, we offer comprehensive security testing of V2X applications.

4.3 Infrastructure Sensor Kit

The aim is to implement various relevant environmental sensors in an Infrastructure Sensor Kit. As the first implementation, a sensor was developed to measure the road conditions. The sensor is able to distinguish between different road conditions such as wet, dry or frozen. It could, for example, warn drivers of dangerous road conditions such as slippery roads or suggest alternative routes. For this purpose, the sensor would have to be equipped with its own, **independent evaluation algorithm**. It is also connected to a RSU, in order to be able to transmit the information to a cloud-based platform as well as to the surrounding vehicles.

In combination with the proposed testbed, the planned Infrastructure Sensor Kit is an excellent basis for the further development and expansion of **sensor applications** for smart traffic and smart city systems.

4.4 Simulation as an Online Service

All the testbed services described in Chapter 3 could also be made available as an online web-service. For this purpose, our customer can access **configurable or pre-configured simulations** via an online website. There, the user can remotely control the simulation parameters, define new scenarios and get qualified evaluation results. This ongoing development addresses the rapid development of connected applications with **abstracted technology know-how**. Thus, the developer can concentrate on the actual function of the intended application.

5 FURTHER READING

Car2MEC

Enhanced Traffic Safety with LTE and Mobile Edge Computing [Online]
Available at: https://www.esk.fraunhofer.de/en/research/projects/car2mec.html
[Access 02/20/2018].

Connected Mobility Expertise

Available at: https://www.esk.fraunhofer.de/en/business-areas/connected_mobility.html
[Access 02/22/2018].

DANA

Description and Analysis of Networked Applications Tool [Online]
Available at: https://www.dana.fraunhofer.de [Access 02/16/2018].

Embedded Security Summary

Available at: https://www.aisec.fraunhofer.de/en/fields-of-expertise/embedded-security.html
[Access 02/22/2018].

ezCar2x®

Website of the communication framework [Online]
Available at: https://www.ezcar2x.fraunhofer.de [Access 02/16/2018].

High-Performance Center Connected Secure Systems

Website of the High-Performance Center Connected Secure Systems [Online]
Available at: www.svs.bayern [Access 02/21/2018].

Sensor Integration Overview

Available at: https://www.emft.fraunhofer.de/en/research/services.html [Access 02/22/2018].

VICTOR

Demonstrator for Connected Driving [Online]
Available at: https://www.esk.fraunhofer.de/en/research/labs/victor.html [Access 02/20/2018].

6 ABBREVIATIONS

ADTF	Automotive Data and Time-Triggered Framework
AFL	American Fuzzy Lop
CA	Certificate Authority
CAM	Cooperative Awareness Message
DANA	Description and Analysis of Networked Applications
DAST	Dynamic Application Security Testing
DENM	Decentralized Environmental Notification Message
ETSI	European Telecommunications Standards Institute
HiL	Hardware-in-the-Loop
HSM	Hardware Security Module
ITS	Intelligent Transport Systems
MITM	Man-in-the-Middle
OCSP	Online Certificate Status Protocol
OTA	Over-the-Air
PKI	Public Key Infrastructure
PLR	Packet Loss Rate
QoS	Quality of Service
RSU	Roadside Unit
SAST	Static Application Security Testing
TLS	Transport Layer Security
TPM	Trusted Platform Module
V2X	Vehicle-to-X
ViL	Vehicle-in-the-Loop
VPN	Virtual Private Network

IMPRINT

Editor

Fraunhofer Institute for Embedded Systems and Communication Technologies ESK
Hansastr. 32
D-80686 Munich

Phone: +49 089 547088-0
Fax: +49 089 547088-220
info@esk.fraunhofer.de
www.esk.fraunhofer.de

Manuela Freese-Wagner
Phone: +49 89 547088-353
manuela.freese-wagner@esk.fraunhofer.de

Authors

Fraunhofer AISEC
Dr. Daniela Pöhn, Sascha Wessel, Felix Fischer,
Oliver Braunsdorf
{daniela.poehn, sascha.wessel, felix.fischer,
oliver.braunsdorf}@aisec.fraunhofer.de

Fraunhofer EMFT
Franz Wenninger
Franz.Wenninger@emft.fraunhofer.de

Fraunhofer ESK
Dominique Seydel, Dr. Gereon Weiß, Karsten Roscher
{dominique.seydel, gereon.weiss, karsten.roscher}
@esk.fraunhofer.de

Image Credits

Cover picture:
Fraunhofer ESK

Page 5:
panthermedia.net / amasterpics123

Page 8, Figure 3.1:
panthermedia.net / amasterpics123
panthermedia.net / Robert Crusitu
panthermedia.net / Sasa Ilic
panthermedia.net / chesky_w
panthermedia.net / Kheng Ho Toh
panthermedia.net / ktsdesign
Fraunhofer ESK

Page 11, 25 and 27:
Fraunhofer ESK

Page 16:
panthermedia.net / Sasa Ilic

Page 19:
panthermedia.net / Robert Crusitu

Page 26, Figure 4.1:
panthermedia.net / Pichet Wissawapipat

Page 30:
panthermedia.net / ktsdesign

Green honeycombed elements:
panthermedia.net / RAPEEPON BOONSONGSUWAN