Project acronym:	EVITA
Project title:	E-safety vehicle intrusion protected applications
Project reference:	224275
Programme:	Seventh Research Framework Programme (2007–2013) of the European Community
Objective:	ICT-2007.6.2: ICT for cooperative systems
Contract type:	Collaborative project
Start date of project:	1 July 2008
Duration:	42 months

Deliverable D1.2.5.1: Presentation slides from the EVITA project workshop

Editor:	Olaf Henniger (Fraunhofer Institute SIT)

Dissemination level:	Public
Deliverable type:	Other
Date:	1 July 2010

Abstract

The objective of the EVITA project is to design, verify, and prototype building blocks for automotive on-board networks where security-relevant components are protected against tampering and sensitive data are protected against compromise. Thus, the EVITA project will provide a basis for the secure deployment of electronic safety aids based on vehicle-to-vehicle and vehicle-to-infrastructure communication. In order to support a broad utilisation of the project results, a public dissemination workshop has been held on 1 July 2010 after the project has reached a sufficiently mature stage. The objective of this workshop has been to present project results such as the secure on-board architecture and protocol specifications to the public and to instigate a wider review. The target audience has included, beside the interested public, also potential users of the EVITA results such as car manufacturers and automotive electronics suppliers. The workshop has been organized in cooperation with CAST (Competence Center for Applied Security Technology) in Darmstadt, Germany, see http://www.cast-forum.de/en/workshops/infos/129.

Contents

Implementation of the simTD security architecture

Hagen Stübing, Norbert Bißmeyer

Zusammenfassung

Sim^{TD} is the worldwide first field operational trial for Car-to-X technology that applies several hundred vehicles in a real-life environment in order to evaluate an entire spectrum of applications with regard to effects on traffic safety and traffic efficiency.

For a comprehensive integration of security into the sim^{TD} architecture several challenges have to be met. It has to be examined which security standards can be deployed with the given architecture. Adaptations and further extensions of common standards are necessary in order to fit the security and privacy mechanisms into the entire C2X architecture. Furthermore the security mechanisms have to deal with hardware restrictions due to automotive requirements and funding restrictions. Finally novel concepts have to be developed with regard to the scale factor of the large fleet consisting of vehicles and infrastructure.

In this work we give a first glance on a security architecture for C2X communications. We present the different concepts, protocols and cryptographic procedures used in sim^{TD}. Furthermore the chosen strategies to protect the driver´s privacy based on pseudonyms are proposed.

CV

Hagen Stübing studied Electrical Engineering at the Technical University of Darmstadt with emphasis on embedded system design. In 2004 he joined a double degree program with the Universitat Politècnica de Catalunya in Barcelona, Spain from where he received his Master's degree in Information and Communication Technologies in 2006. He completed his Diploma Degree in Electrical Engineering (Dipl.-Ing.) in 2008.
Since July 2008 he is working towards his PhD at Adam Opel GmbH in the field of vehicular ad hoc networks. In particular his research interests are physical protection techniques for security and privacy issues as well as security architectures in general.

Norbert Bißmeyer studied Applied Computer Science at the FH Münster and received his Bachelor's degree in 2006. Afterwards he studied Advanced Security Engineering at the FH Joanneum in Austria and Ireland and received his Master`s degree in 2008. Since November 2008 he is working at the Fraunhofer Institute for Secure Information Technology in Darmstadt in the department Secure Mobile Systems. He is working in the field of vehicular ad hoc networks with focus on security and privacy concepts.

Kontakt

Hagen Stübing
GME Advanced Active Safety
Adam Opel GmbH, Rüsselsheim
Tel. +49 6142 / 7-53888

Norbert Bißmeyer
Fraunhofer SIT
Rheinstaße 75, 64295 Darmstadt
Tel. +49 6151 869-324

eMail: Hagen.Stuebing@de.opel.com eMail: norbert.bissmeyer@sit.fraunhofer.de

Literatur

[1] N. Bißmeyer, H. Stübing, M. Mattheß, J. P. Stotz, J. Schütte, M. Gerlach, and F. Friederici, sim^{TD} Security Architecture, Embedded Security in Cars Conference (escar), 2009

Implementation of the sim^{TD} Security Architecture

Norbert Bißmeyer
Fraunhofer SIT, Secure Mobile Systems
Norbert.Bissmeyer@sit.fraunhofer.de

Hagen Stübing
Adam Opel GmbH, GME GTE E&E Advanced Engineering
Hagen.Stuebing@de.opel.com

Sichere Intelligente Mobilität
Testfeld Deutschland

Content

Introduction | sim^{TD} Project Overview

- Objectives:

  Safe and efficient mobility using Car-to-X
- Issues:

  To test and validate technologies and functions for Car-to-X .

  To evaluate the effectivness and benefits gained by Car-to-X

  To gather sufficient information to support a country-wide deployment decision
- Duration: 48 month
- Test Fleet: 100 controlled and 300 free flow test vehicles, over 100 roadside stations

- Partner
- Sponsor

Federal Ministry of Education and Research

Federal Ministry of Transport, Building and Urban Affairs

Federal Ministry of Economics and Technology

Introduction | Application Examples

Driving and Safety

- End of Congestion Warning
- Road Weather Warning
- Emergency Vehicle Warning
- Electronic Brake Light

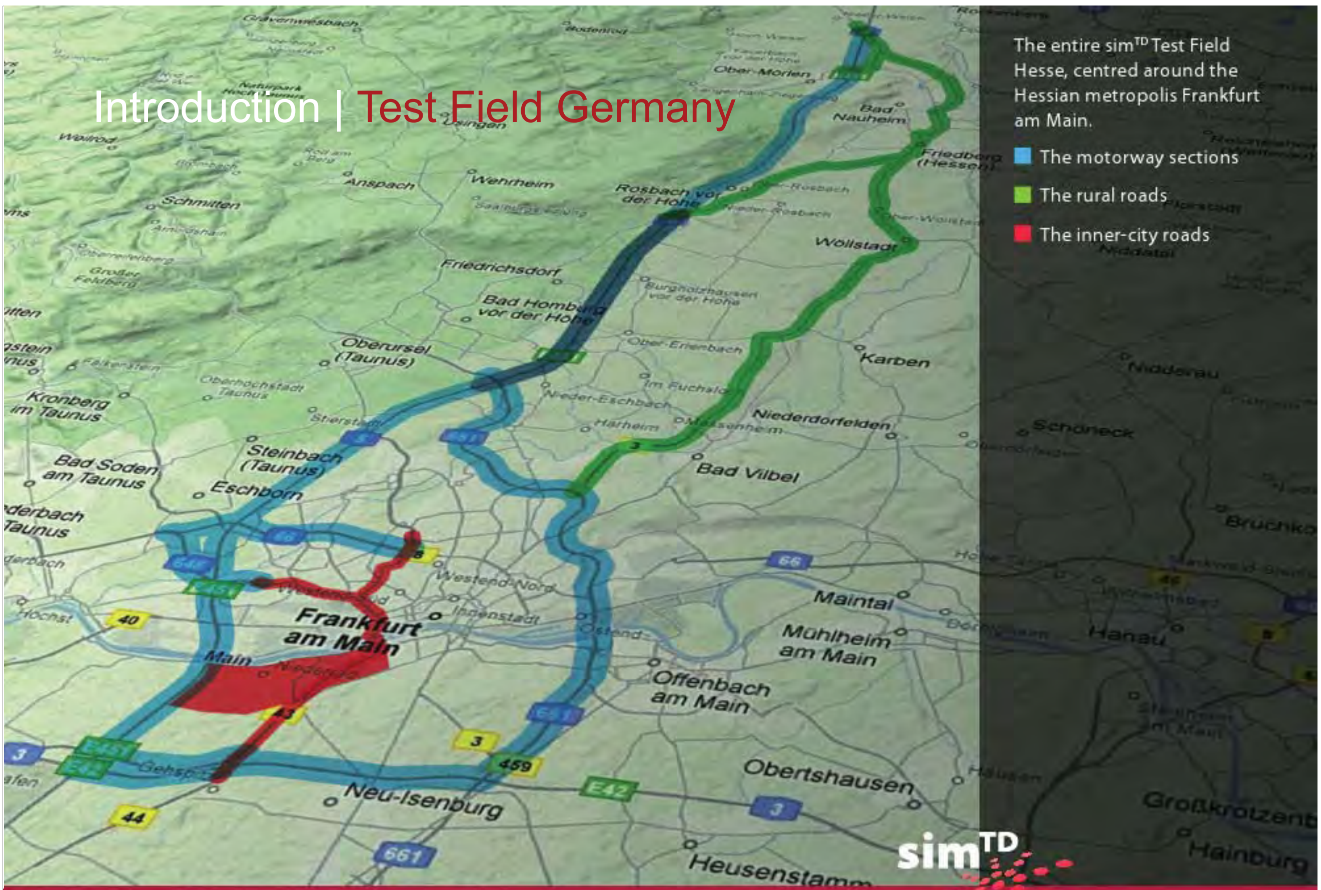
Traffic Efficiency

- Road Works Information
- Traffic Information
- Traffic Light Phase Assistent
- Advanced Route Guidance

Additional Services

- Connectivity
- Location Based Service
- sim^{TD} HW / SW Remotediagnosis

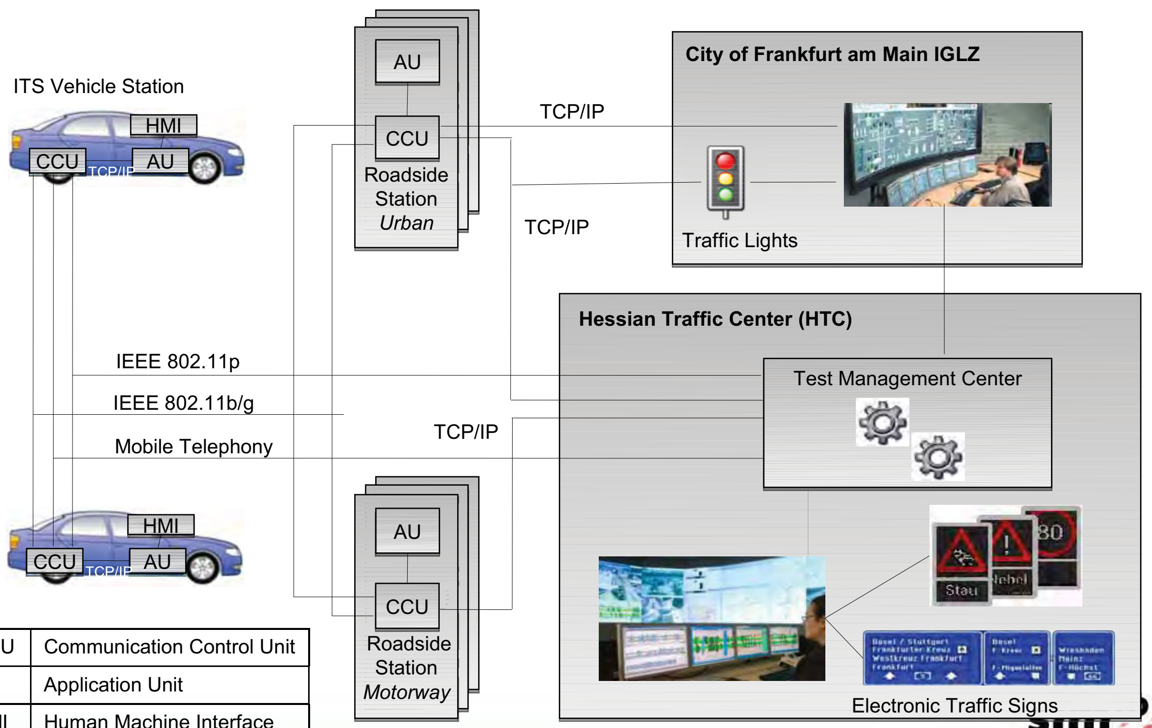
Introduction | Test Field Germany



The entire sim^{TD} Test Field Hesse, centred around the Hessian metropolis Frankfurt am Main.

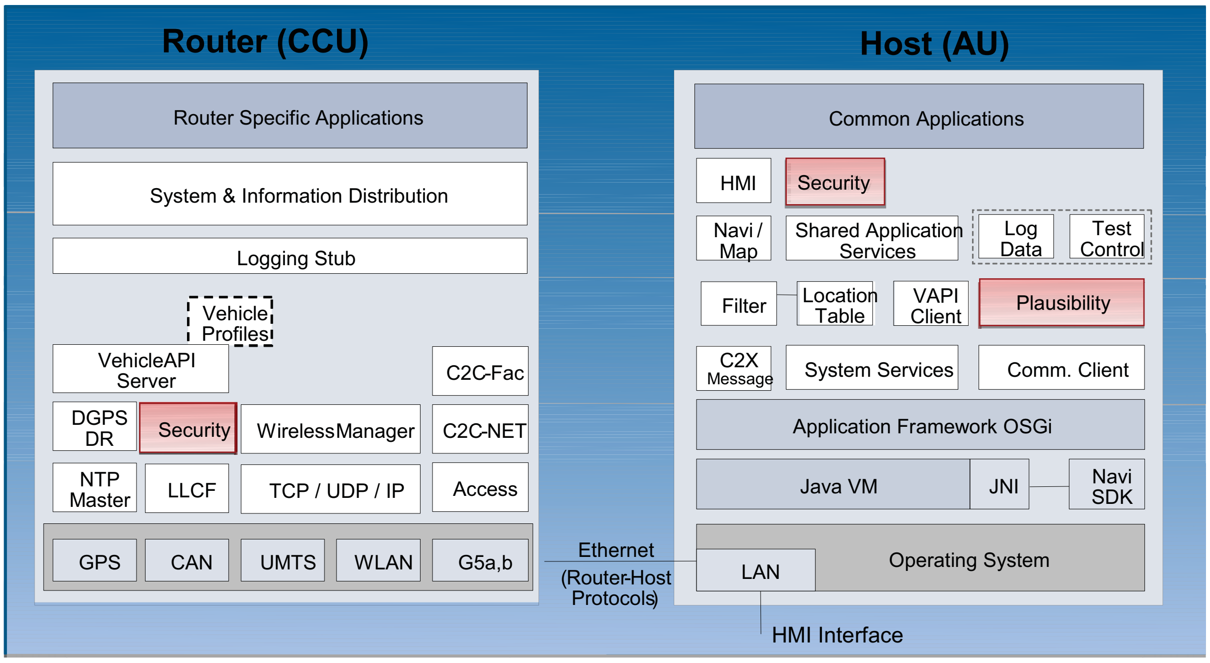
- 🟦 The motorway sections
- 🟩 The rural roads
- 🟥 The inner-city roads

System Architecture



CCU	Communication Control Unit
AU	Application Unit
HMI	Human Machine Interface

System Architecture| Vehicle CCU and AU



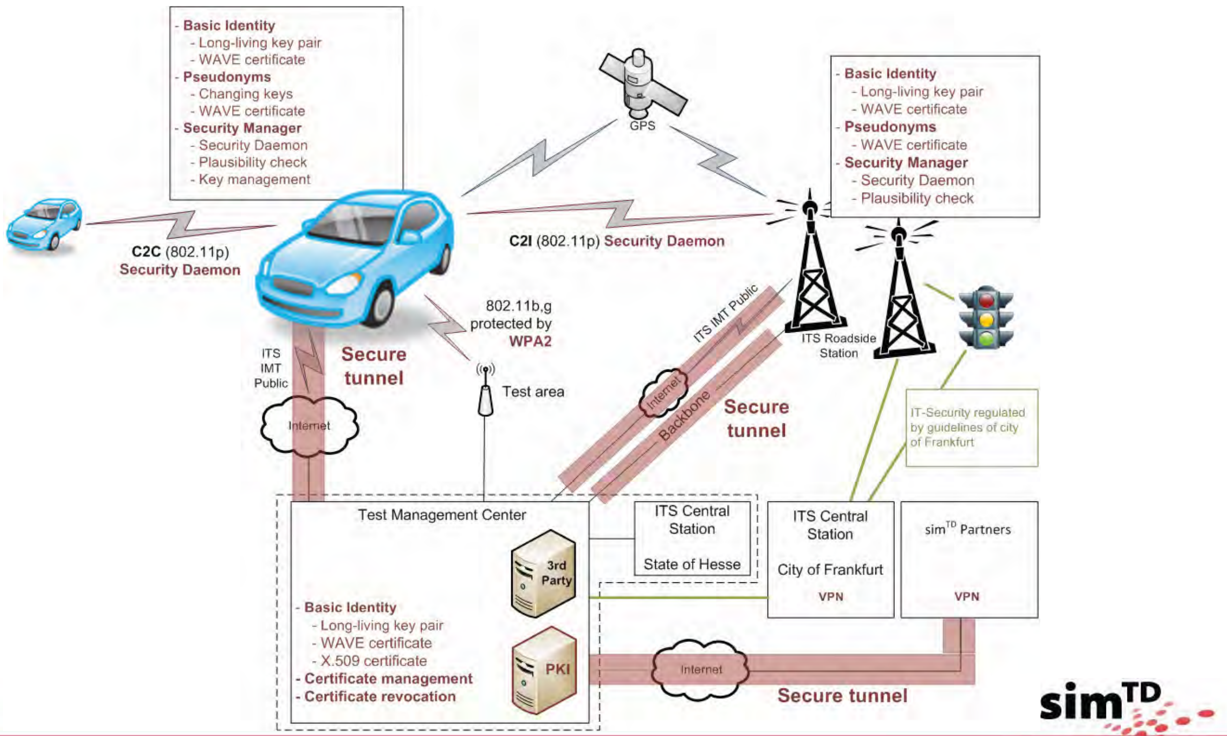
Data Security

- Objectives
  - Message Integrity
  - Authenticity
  - Confidentiality
- Standards
  - IEEE 1609.2 with adaption of the cryptographic algorithms
  - RSA with PKCS #1 v2.0 and SHA-1 according OpenSSL v.1.0.0
- Hardware
  - CCU
  - AU
- Basic Load
  - 20 incoming messages per second
  - 2 outgoing messages per second
- Plausibility check:
  - Range of values
  - Transmission frequency
  - Movement plausibility of vehicles

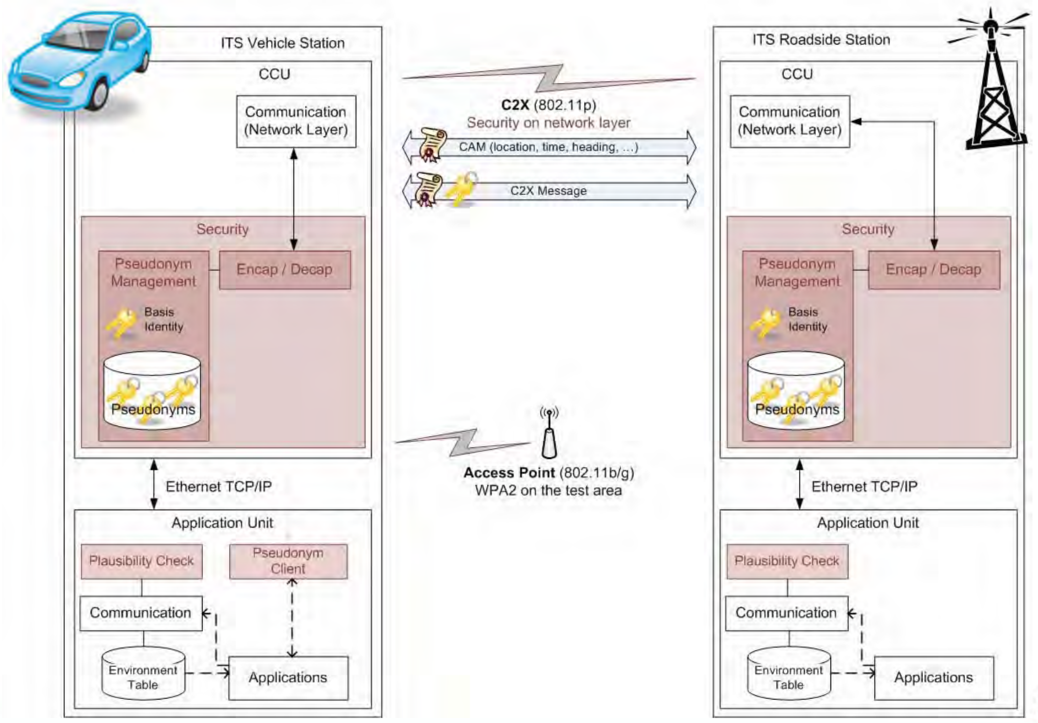
Data Security | Decision Basis for Cryptographic Algorithms

Criterion	ECDSA 256	RSA 512 / 1024	Symmetric Keys (HMAC)
PKI necessary	Yes	Yes	No
Key distribution	Yes	Yes	Yes
Revocation possible	Yes	Yes	No
Additional HW	Yes (Crypto HW , PKI)	Yes (PKI)	No
Verification time	> 54 ms	~ 1.9 ms	< 1 ms
Security overhead per message	~ 200 Byte	~ 250 Byte	~ 60 Byte
Authentication	Yes	Yes	No
Active Revocation necessary	No	No	Yes
Auditabilty	Yes	Yes	No
Security Risk (RFC 3766)	136 Bit	50 Bit	128 Bit
Privacy	Yes	Yes	No
Experience for Future ITS	Yes	Yes	No
Standards	IEEE 1609.2	Adapted IEEE 1609.2	No C2X

Data Security | Security Architecture

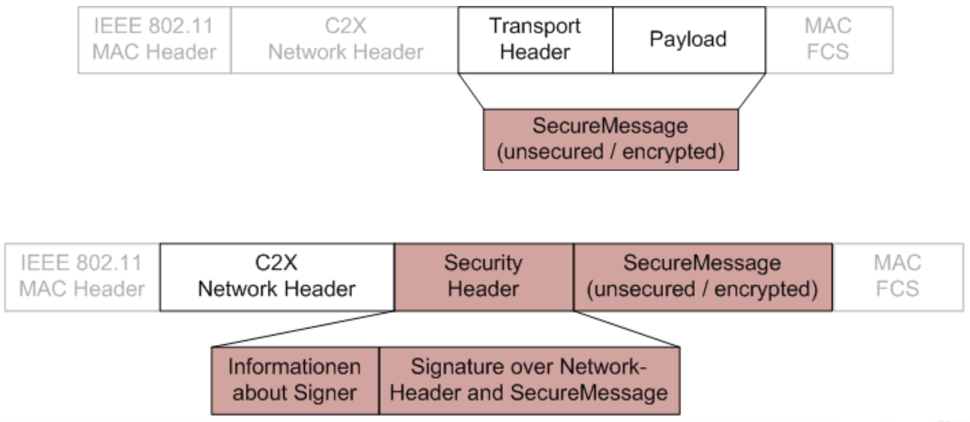


Data Security | Car-to-Infrastructure Communication

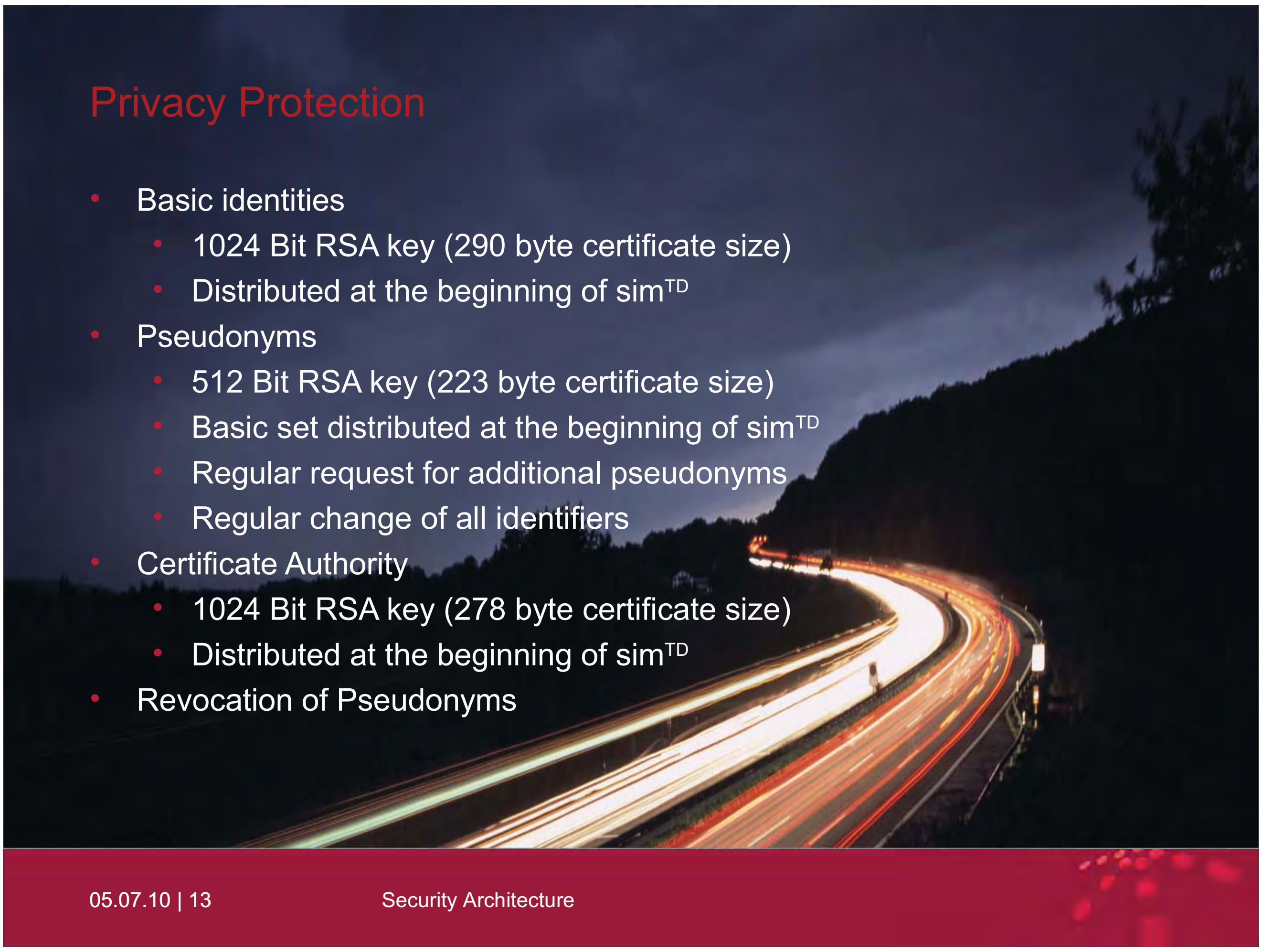


Data Security | C2X Message Format

- Application of IEEE 1609.2 on network layer for securing C2X messages
- Adaption of WAVE message formats
- Implementation of IEEE 1609.2 with RSA
- Application of WAVE Certificates



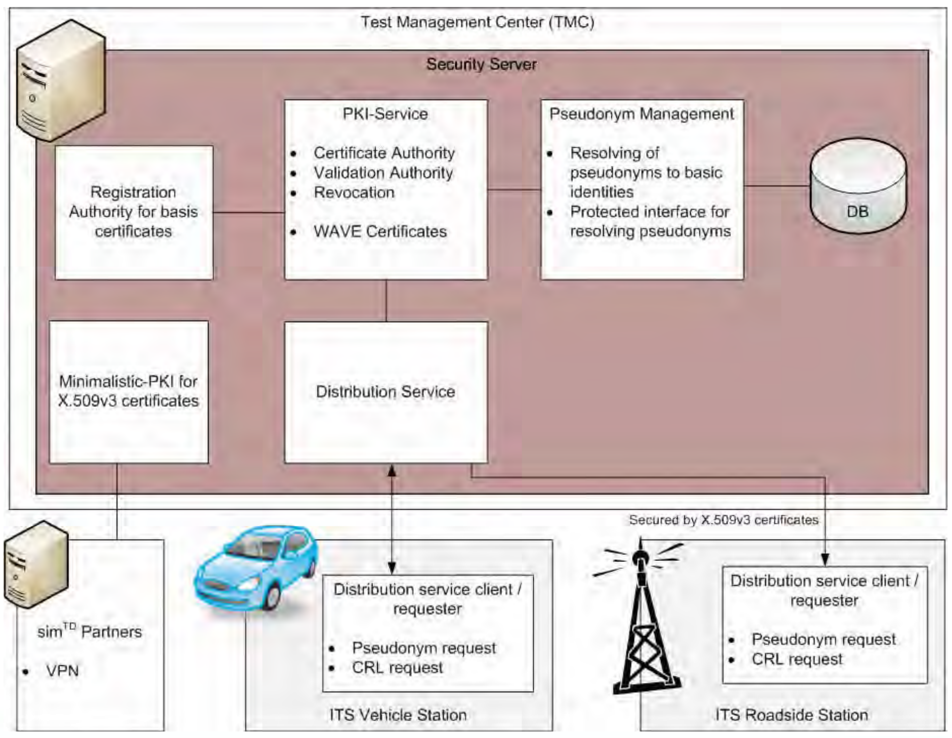
IEEE 802.11 MAC Header
C2X Network Header
Transport Header
Payload
MAC FCS
SecureMessage (unsecured / encrypted)
IEEE 802.11 MAC Header
C2X Network Header
Security Header
SecureMessage (unsecured / encrypted)
MAC FCS
Informationen about Signer
Signature over Network-Header and SecureMessage



Privacy Protection

- Basic identities
  - 1024 Bit RSA key (290 byte certificate size)
  - Distributed at the beginning of sim^{TD}
- Pseudonyms
  - 512 Bit RSA key (223 byte certificate size)
  - Basic set distributed at the beginning of sim^{TD}
  - Regular request for additional pseudonyms
  - Regular change of all identifiers
- Certificate Authority
  - 1024 Bit RSA key (278 byte certificate size)
  - Distributed at the beginning of sim^{TD}
- Revocation of Pseudonyms

Privacy Protection | Pseudonym Distribution



Test Management Center (TMC)
Security Server
Registration Authority for basis certificates
PKI-Service
• Certificate Authority
• Validation Authority
• Revocation
• WAVE Certificates
Pseudonym Management
• Resolving of pseudonyms to basic identities
• Protected interface for resolving pseudonyms
DB
Minimalistic-PKI for X.509v3 certificates
Distribution Service
Secured by X.509v3 certificates
sim^TD Partners
• VPN
Distribution service client / requester
• Pseudonym request
• CRL request
ITS Vehicle Station
Distribution service client / requester
• Pseudonym request
• CRL request
ITS Roadside Station

simTD

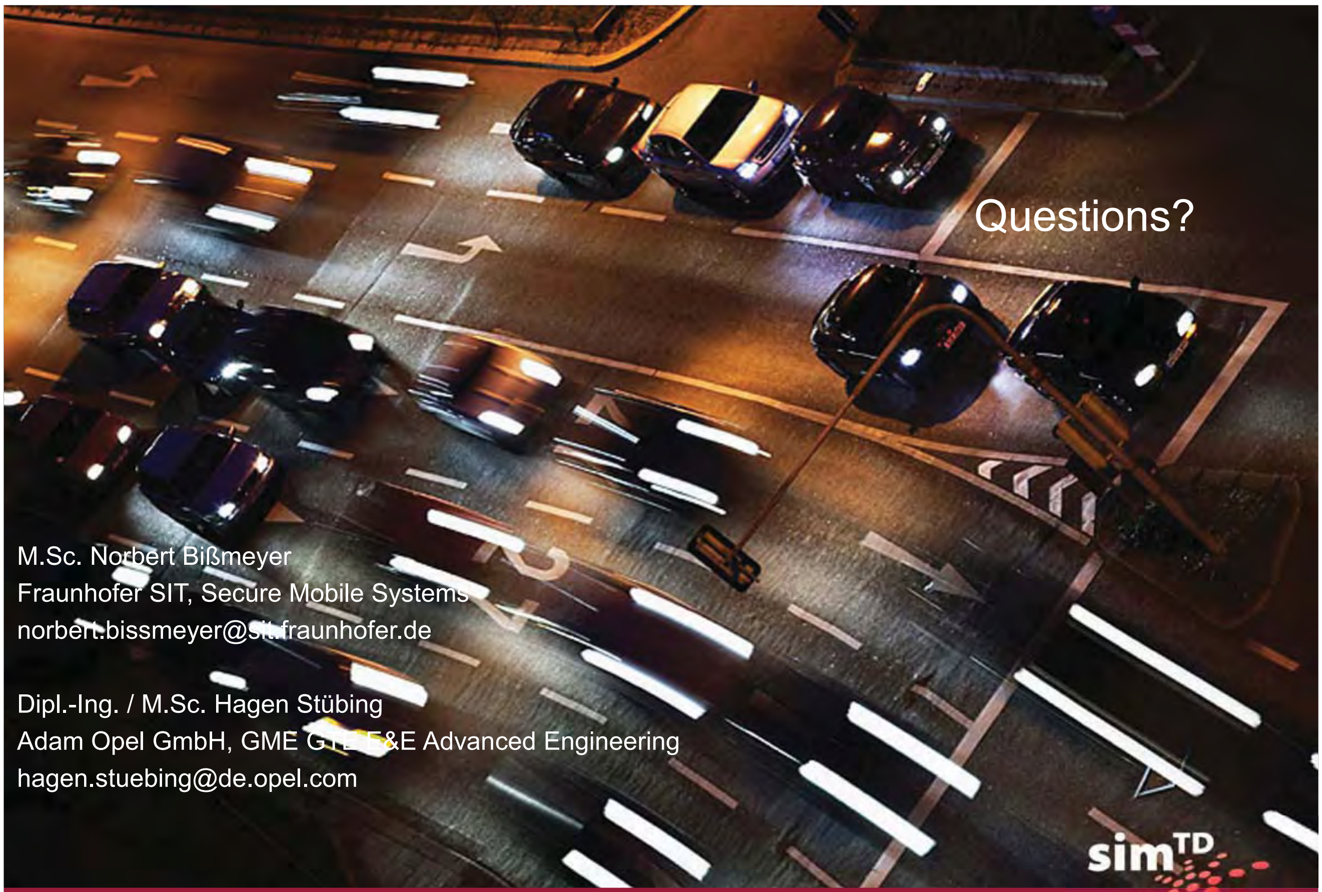
Summary and Outlook

sim^{TD} is worldwide the first field operational test that is large enough to

- test and validate technologies and systems for C2X communication in a real-life environment that exceeds the demonstrator status,
- examine the entire spectrum of applications with regard to the effects on traffic safety and efficiency, and
- learn a lot about integration of security and privacy protection mechanisms into a C2X communication system, and
- gain knowledge for further development and enhancements of security and privacy protocols for C2X communication.

but…

- The sim^{TD} security architecture has been developed under difficult constraints regarding time, performance and costs
- A deeper scientific discussion is needed and still ongoing in the context of a future standardization



Questions?

M.Sc. Norbert Bißmeyer
Fraunhofer SIT, Secure Mobile Systems
norbert.bissmeyer@sit.fraunhofer.de

Dipl.-Ing. / M.Sc. Hagen Stübing
Adam Opel GmbH, GME GTE E&E Advanced Engineering
hagen.stuebing@de.opel.com

Identification of Security Requirements for Vehicular Communication Systems

Roland Rieke

Fraunhofer Institute for Secure Information Technology SIT

Abstract

In vehicular communication systems vehicles and roadside units communicate in ad hoc manner to exchange information such as safety warnings and traffic information. As a cooperative approach, vehicular communication systems can be more effective in avoiding accidents and traffic congestion than current technologies where each vehicle tries to solve these problems individually. However, introducing dependence of possibly safety-critical decisions in a vehicle on information from other systems, such as other vehicles or roadside units, raises severe concerns to security issues. Security is an enabling technology in this emerging field because without security some applications within those cooperating systems would not be possible at all.

This talk addresses the security requirements elicitation step in the security engineering process for such vehicular communication systems. The method comprises the tracing down of functional dependencies over system component boundaries right onto the origin of information as a functional flow graph. Based on this graph, we systematically deduce comprehensive sets of formally defined authenticity requirements for the given security and dependability objectives. The proposed method thereby avoids premature assumptions on the security architecture's structure as well as the means by which it is realised.

CV

Roland Rieke works since 1982 as a senior researcher at the Fraunhofer Institute for Secure Information Technology SIT. His research interests are focused on the development of methods and tools for formal security models and application of these techniques for architecting secure and dependable systems. In the project EVITA (E-safety Vehicle Intrusion proTected Applications), for instance, he worked on a method for security requirements elicitation in systems of systems applied in the context of vehicular communication systems. He is currently working on predictive security analysis for event-driven processes in the context of the Internet of things within the project ADiWa (Alliance Digital Product Flow). His recent papers furthermore comprise work on attack graph analysis and on proving security and dependability properties in parameterised systems based on self-similarity. Roland will be the research director of the project MASSIF (MAnagement of Security information and events in Service InFrastructures), a large-scale integrating project co-funded by the European Commission starting in October 2010. He is member of the ERCIM working group on Security and Trust Management.

Contact

Roland Rieke
Security Modeling and Model Validation
Fraunhofer Institute for Secure Information Technology
Rheinstrasse 75
64295 Darmstadt, Germany
Phone +49 6151 869-284
eMail: roland.rieke@sit.fraunhofer.de

Literatur

[1] Andreas Fuchs and Roland Rieke. Identification of Authenticity Requirements in Systems of Systems by Functional Security Analysis. In *Workshop on Architecting Dependable Systems (WADS 2009), in Proceedings of the 2009 IEEE/IFIP Conference on Dependable Systems and Networks, Supplementary Volume*, 2009. http://sit.sit.fraunhofer.de/smv/publications/.

[2] Andreas Fuchs and Roland Rieke. Identification of Security Requirements in Systems of Systems by Functional Security Analysis. In C. Gacek A. Casimiro, R. de Lemos, editor, *Architecting Dependable Systems 7*. Springer, to appear.

[3] Alastair Ruddle, David Ward, Benjamin Weyl, Sabir Idrees, Yves Roudier, Michael Friedewald, Timo Leimbach, Andreas Fuchs, Sigrid Gürgens, Olaf Henniger, Roland Rieke, Matthias Ritscher, Henrik Broberg, Ludovic Apvrille, Renaud Pacalet, and Gabriel Pedroza. Security requirements for automotive on-board networks based on dark-side scenarios. EVITA Deliverable D2.3, EVITA project, 2009. http://evita-project.org/deliverables.html.

Fraunhofer SIT

Identification of Security Requirements for Vehicular Communication Systems

Andreas Fuchs and Roland Rieke

roland.rieke@sit.fraunhofer.de

Fraunhofer Institute for Secure Information Technology SIT, Darmstadt, Germany

CAST-Forum Workshop Mobile Security for Intelligent Cars, July 2010

Overview

1. Motivation
   - Scenario - cooperative reasoning in vehicular ad hoc communication
   - Dependence of safety critical decisions raises security concerns
2. Objectives
   - Systematic security requirements elicitation for novel architectures
   - Avoid premature architecture constraints
3. Functional Security Analysis
4. Results and Outlook

Rationale for New Vehicular Architecture Using Cooperative Reasoning

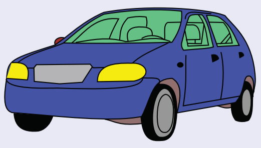
overall goal

reduce number and impact of accidents in Europe

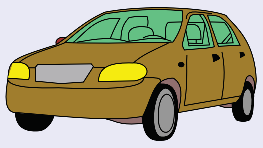
difficulties

to improve safety measures in vehicles $\rightsquigarrow$ improve infrastructure

cooperative approach



$\Rightarrow$ warning $\Rightarrow$



vehicular communication systems can be more effective in avoiding accidents and traffic congestion than current technologies where each vehicle tries to solve these problems individually

- the work presented here was developed within the project EVITA being co-funded by the European Commission within the Seventh Framework Programme
- EVITA develops internal on-board security such as trust anchor and secure storage of secret keys which is the basis for secure external vehicular communication.

Related European Projects

SeVeCom (2006-2009) - protection of external vehicular communication

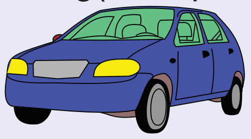
PRECIOSA (2008-2010) - protection of privacy in vehicular communication

EVITA (2008-2011) - protection of on-board networks

http://www.evita-project.org

Use Case: Send Danger Warning

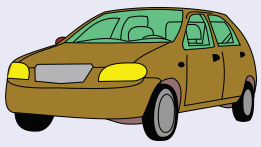
sense(ESP,SlipperyWheels)
positioning(GPS,position)



send(CU,danger(position,type))

$\longrightarrow$

receive(CU,danger(position,type))
positioning(GPS,position)

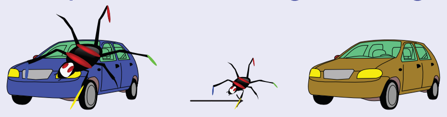


show(HMI,D,warn(relative-position))

ESP - Electronic Stability Protection	HMI - Human Machine Interface
GPS - Global Positioning System	D - Driver
CU - CommunicationUnit	

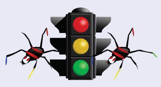
Security Enables Novel Vehicular Communication Systems
$\hookrightarrow$ Exposing Vehicles to the Internet makes them Vulnerable

- Attacks on safety
  - Unauthorized brake
  - Attack active brake function
  - Tamper with warning message



  - Attacking E-Call
  - On-Board Diagnostics (OBD) flashing attack

- Attacks on privacy
  - Trace vehicle movement
  - Compromise driver privacy
- Manipulate traffic flow
  - Simulate traffic jam for target vehicle
  - Force green lights ahead of attacker



  - Manipulate speed limits
  - Prevent driver from passing toll gate
  - Engine refuses to start
- Increase/Reduce driver's toll bill

Security Requirements Engineering Process

- the identification of the target of evaluation and
  the principal security goals and
  the elicitation of artifacts (e.g. use case and threat scenarios)
  as well as risk assessment
- the actual security requirements elicitation process
- a requirements categorisation and prioritisation,
  followed by requirements inspection

Further Steps in Security Engineering

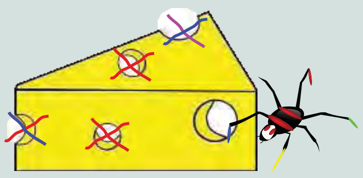
- security requirements (structural) refinement
- mapping of security requirements to security mechanisms

Methods to Elicit Security Requirements

- misuse cases (attack analysis),
- anti-goals derived from negated security goals,
- use Jackson‘s problem diagrams,
- actor dependency analysis (i^* approach)

Why yet another Approach ?

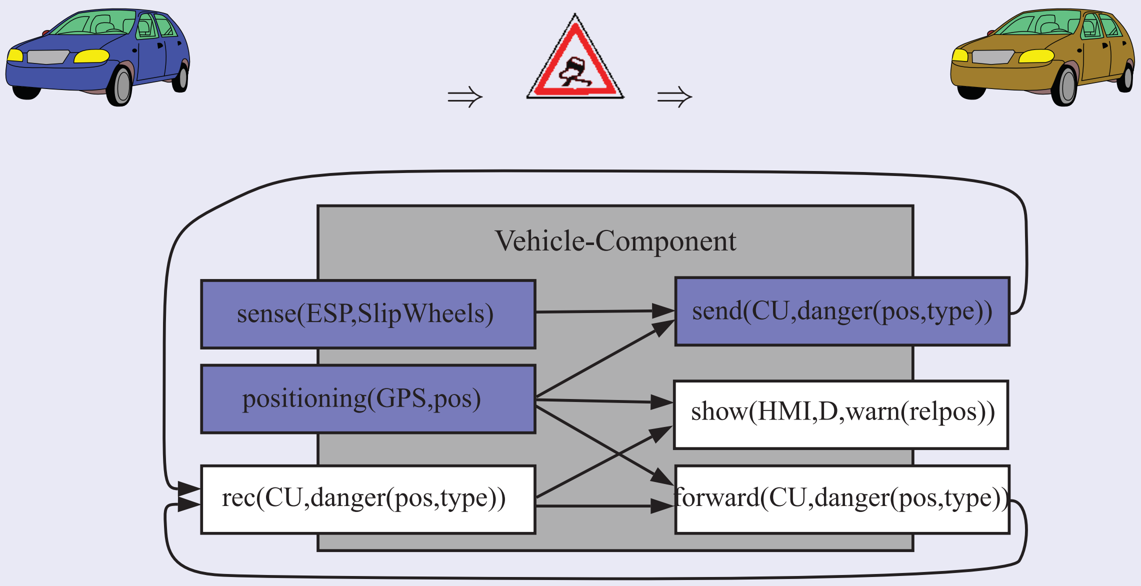
Completeness



Avoid Premature Architecture Constraints

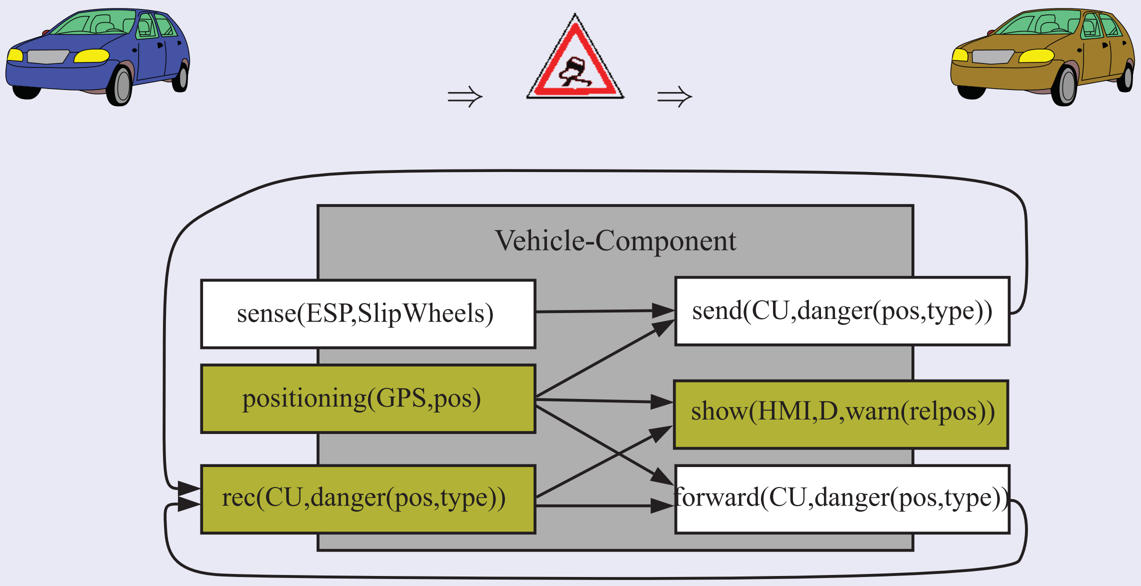
- protocols SSL/TLS/VPN/IPv6
- trust anchor TPM
- infrastructure PKI, PDP/PEP
- end-to-end/hop-by-hop

Functional Component Model



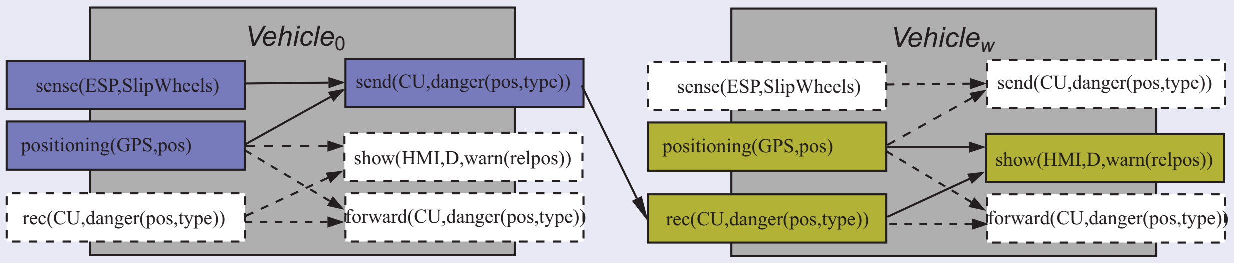
Security goal of the system at stake:
Whenever a certain output action happens, the input action that presumably led to it must actually have happened.

Functional Component Model



Security goal of the system at stake:
Whenever a certain output action happens, the input action that presumably led to it must actually have happened.

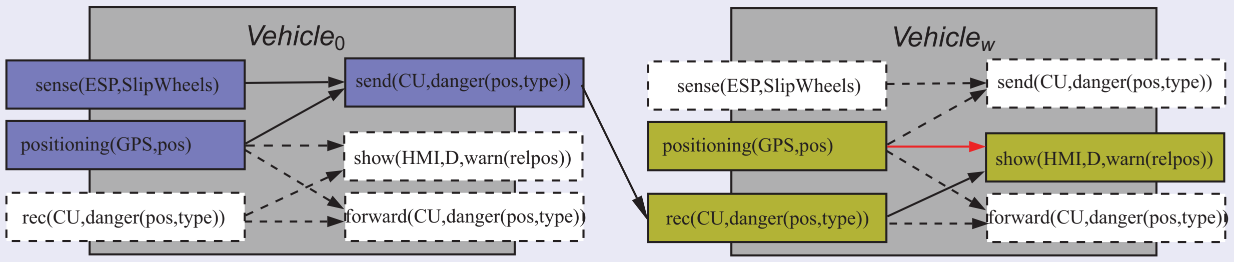
Functional Security Requirement Identification



Formally, the functional flow among actions can be interpreted as an ordering relation ζ_i on the set of actions Σ_i in a certain system instance i.

$$\begin{aligned}\zeta_1 = \{ \; & (positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos))),\\ & (rec(CU_w, danger(pos, type)), show(HMI_w, D_w, warn(relpos))),\\ & (send(CU_0, danger(pos, type)), rec(CU_w, danger(pos, type))),\\ & (sense(ESP_0, SlipWheels), send(CU_0, danger(pos, type))),\\ & (positioning(GPS_0, pos), send(CU_0, danger(pos, type)))\}\end{aligned}$$

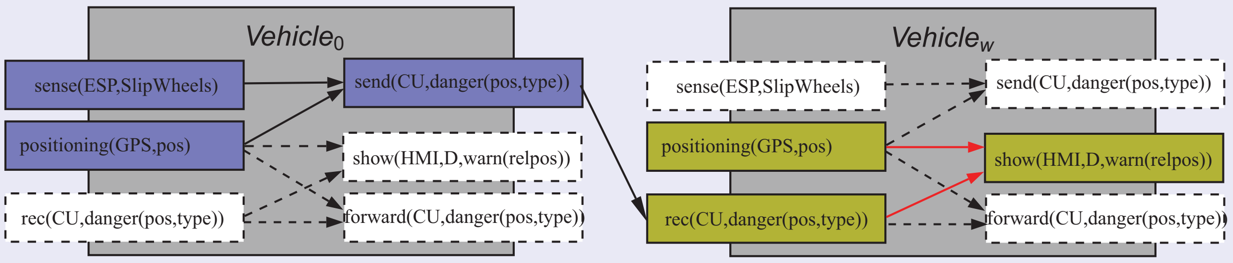
Functional Security Requirement Identification



Formally, the functional flow among actions can be interpreted as an ordering relation ζ_i on the set of actions Σ_i in a certain system instance i.

$$\begin{aligned}\zeta_1 = \{ \; & (positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos))),\\ & (rec(CU_w, danger(pos, type)), show(HMI_w, D_w, warn(relpos))),\\ & (send(CU_0, danger(pos, type)), rec(CU_w, danger(pos, type))),\\ & (sense(ESP_0, SlipWheels), send(CU_0, danger(pos, type))),\\ & (positioning(GPS_0, pos), send(CU_0, danger(pos, type)))\}\end{aligned}$$

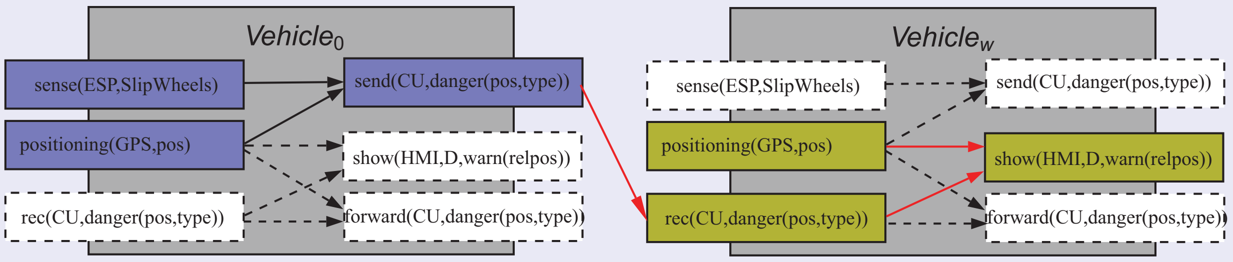
Functional Security Requirement Identification



Formally, the functional flow among actions can be interpreted as an ordering relation ζ_i on the set of actions Σ_i in a certain system instance i.

$$\begin{aligned}\zeta_1 = \{ \; & (positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos))),\\ & (rec(CU_w, danger(pos, type)), show(HMI_w, D_w, warn(relpos))),\\ & (send(CU_0, danger(pos, type)), rec(CU_w, danger(pos, type))),\\ & (sense(ESP_0, SlipWheels), send(CU_0, danger(pos, type))),\\ & (positioning(GPS_0, pos), send(CU_0, danger(pos, type)))\}\end{aligned}$$

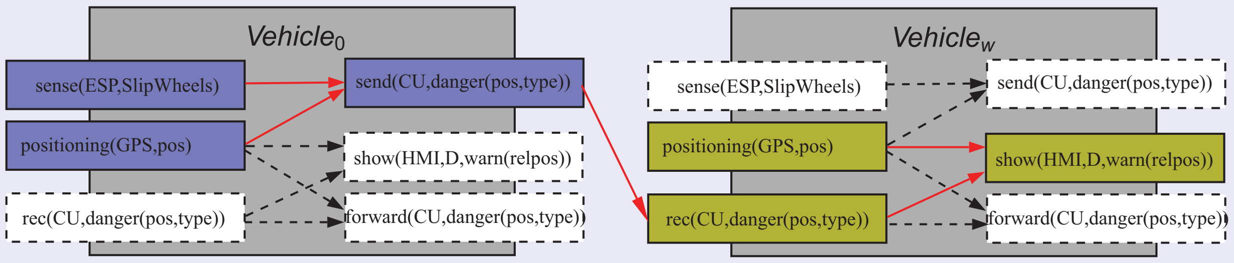
Functional Security Requirement Identification



Formally, the functional flow among actions can be interpreted as an ordering relation ζ_i on the set of actions Σ_i in a certain system instance i.

$$\begin{aligned}\zeta_1 = \{ \; & (positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos))),\\ & (rec(CU_w, danger(pos, type)), show(HMI_w, D_w, warn(relpos))),\\ & (send(CU_0, danger(pos, type)), rec(CU_w, danger(pos, type))),\\ & (sense(ESP_0, SlipWheels), send(CU_0, danger(pos, type))),\\ & (positioning(GPS_0, pos), send(CU_0, danger(pos, type)))\}\end{aligned}$$

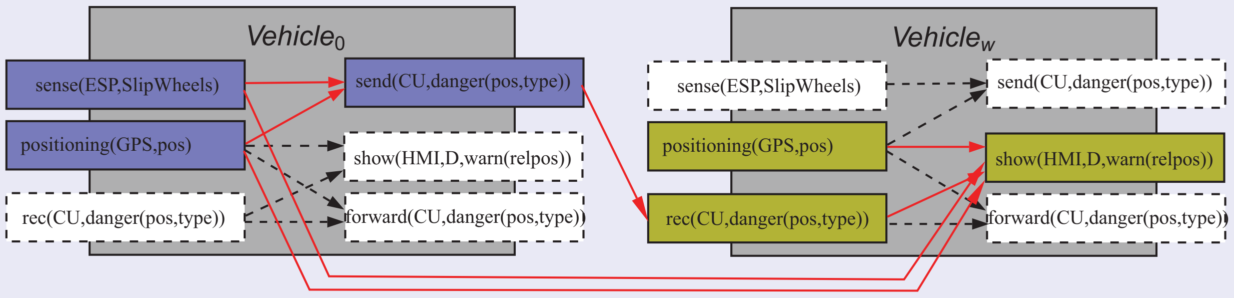
Functional Security Requirement Identification



Formally, the functional flow among actions can be interpreted as an ordering relation ζ_i on the set of actions Σ_i in a certain system instance i.

$$\begin{aligned}\zeta_1 = \{ \; & (positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos))),\\ & (rec(CU_w, danger(pos, type)), show(HMI_w, D_w, warn(relpos))),\\ & (send(CU_0, danger(pos, type)), rec(CU_w, danger(pos, type))),\\ & (sense(ESP_0, SlipWheels), send(CU_0, danger(pos, type))),\\ & (positioning(GPS_0, pos), send(CU_0, danger(pos, type)))\}\end{aligned}$$

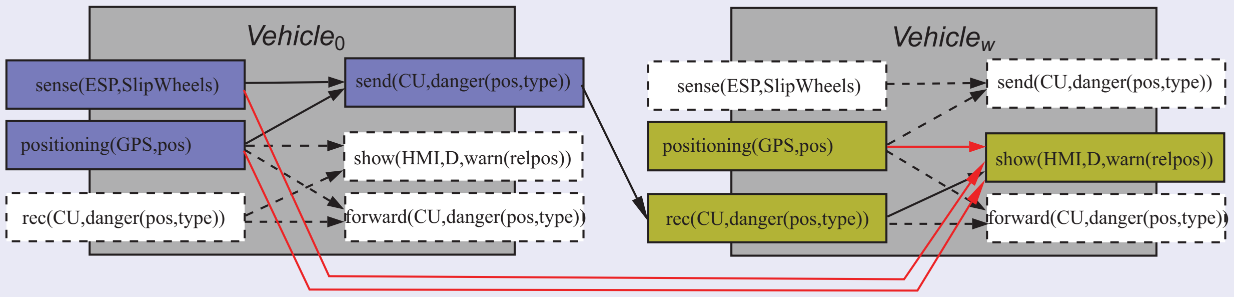
Functional Security Requirement Identification



NOT restrict architecture to hop-by-hop security $\rightarrow$ use transitive closure.

$$\begin{aligned}\zeta_1^* = \zeta_1 \cup \{(x, x) \mid x \in \Sigma\} \cup \{ \; & \\ & (sense(ESP_0, SlipWheels), rec(CU_w, danger(pos, type))),\\ & (sense(ESP_0, SlipWheels), show(HMI_w, D_w, warn(relpos))),\\ & (positioning(GPS_0, pos), rec(CU_w, danger(pos, type))),\\ & (positioning(GPS_0, pos), show(HMI_w, D_w, warn(relpos))),\\ & (send(CU_0, danger(pos, type)), show(HMI_w, D_w, warn(relpos)))\}\end{aligned}$$

Functional Security Requirement Identification



Restrict ζ_i^* to outgoing (max_i) and incoming boundary actions (min_i).

$$\chi_i = \{(x,y) \in \Sigma_i \times \Sigma_i \mid (x,y) \in \zeta_i^* \wedge x \in min_i \wedge y \in max_i\}$$

$$\begin{aligned}\chi_1 = \{ \; & (sense(ESP_0, SlipWheels), show(HMI_w, D_w, warn(relpos))),\\ & (positioning(GPS_0, pos), show(HMI_w, D_w, warn(relpos))),\\ & (positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos)))\}\end{aligned}$$

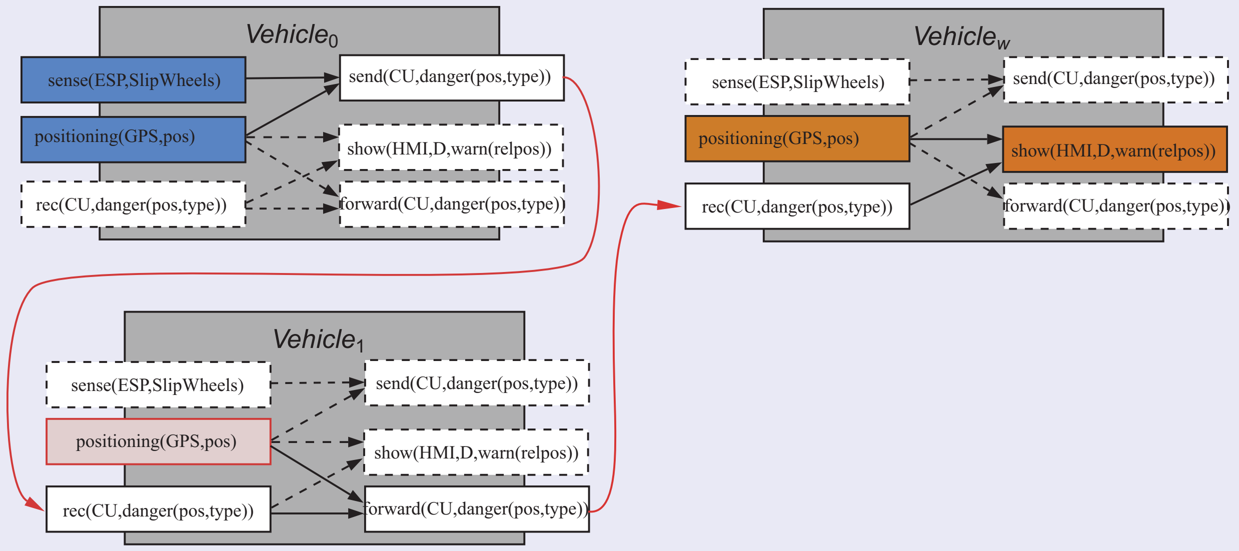
For all $x, y \in \Sigma_i$ *with* $(x,y) \in \chi_i$: $auth(x, y, stakeholder(y))$ is a requirement.

Resulting Authenticity Requirements

For all possible Systems of Systems (SoS) instances for the action $show(HMI_w, D_w, warn(relpos))$ it must be authentic for the driver that:

1. $auth(positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos)), D_w)$
   the relative position of the danger she is warned about is based on correct position information of the drivers vehicle
2. $auth(positioning(GPS_0, pos), show(HMI_w, D_w, warn(relpos)), D_w)$
   the position of the danger she is warned about is based on correct position information of the vehicle issuing the warning
3. $auth(sense(ESP_0, SlipWheels), show(HMI_w, D_w, warn(relpos)), D_w)$
   the danger she is warned about is based on correct sensor data

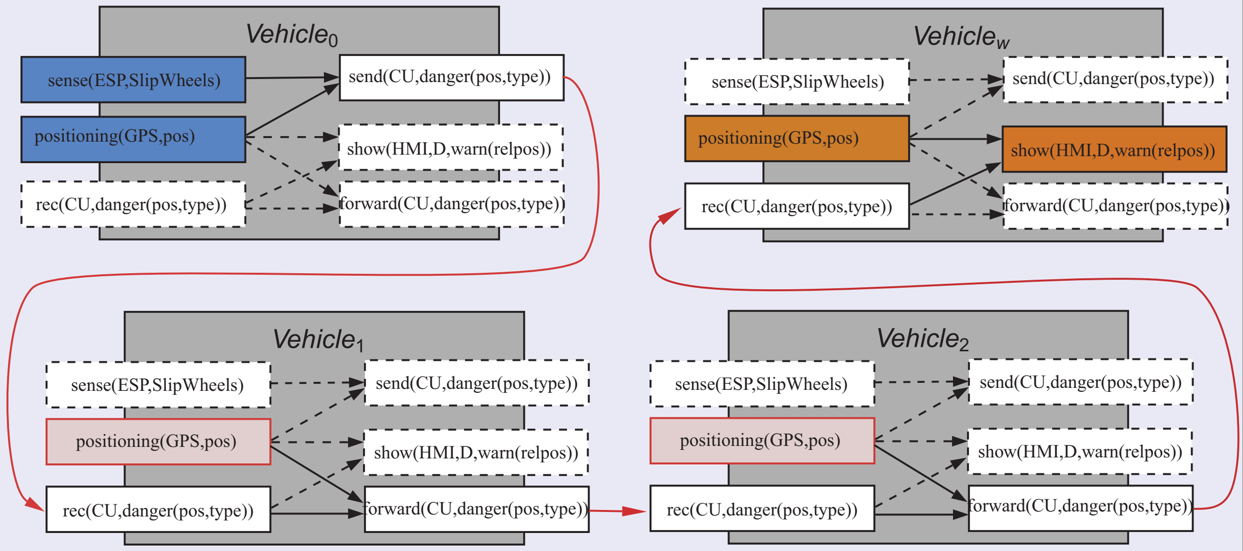
System of Systems (SoS) Instances



An analysis for the second instance will result in:

$$\chi_2 = \chi_1 \cup \{(positioning(GPS_1, pos), show(HMI_w, D_w, warn(relpos)))\}$$

System of Systems (SoS) Instances



An analysis for the second instance will result in:

$$\chi_2 = \chi_1 \cup \{(positioning(GPS_1, pos), show(HMI_w, D_w, warn(relpos)))\}$$

And the third system of systems instance will result in:

$$\chi_3 = \chi_2 \cup \{(positioning(GPS_2, pos), show(HMI_w, D_w, warn(relpos)))\}$$

$$\chi_i = \chi_{i-1} \cup \{(positioning(GPS_{i-1}, pos), show(HMI_w, D_w, warn(relpos)))\}$$

Resulting Authenticity Requirements

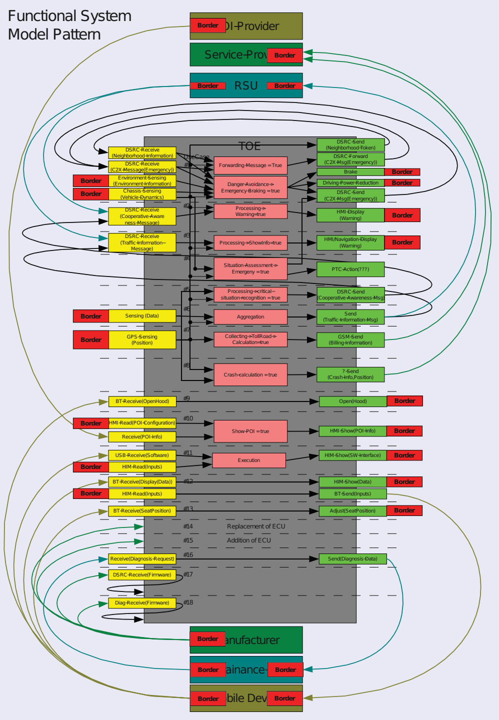
For all possible SoS instances for the action $show(HMI_w, D_w, warn(relpos))$ it must be authentic for the driver that:

1. $auth(positioning(GPS_w, pos), show(HMI_w, D_w, warn(relpos)), D_w)$
   the relative position of the danger she is warned about is based on correct position information of her vehicle
2. $auth(positioning(GPS_0, pos), show(HMI_w, D_w, warn(relpos)), D_w)$
   the position of the danger she is warned about is based on correct position information of the vehicle issuing the warning
3. $auth(sense(ESP_0, SlipWheels), show(HMI_w, D_w, warn(relpos)), D_w)$
   the danger she is warned about is based on correct sensor data
4. $\forall V_x \in V_{forward}$:
   $auth(positioning(GPS_x, pos), show(HMI_w, D_w, warn(relpos)), D_w)$
   position of forwarding vehicles is authentic
   - Breaking (4) would result in a smaller or larger broadcasting area.
   - This cannot cause the warning of a driver that should not be warned.
   - So it is NOT a safety related authenticity requirement.

EVITA (E-Safety Vehicle Intrusion Protected Applications)

In practice, the method has been applied in EVITA to derive authenticity requirements for a new automotive on-board architecture

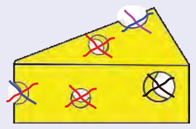
- 17 additional use cases, e.g.
  - safety reaction: active brake
  - traffic information
  - e-Tolling
  - eCall
  - remote car control
  - remote diagnosis/flashing
- 29 authenticity requirements elicited
- system model comprising 38 component boundary actions
- 16 system boundary actions (9 max, 7 min elements)



http://www.evita-project.org/Deliverables/EVITAD2.3.pdf

Contribution of Proposed Approach

Identification of a consistent and complete set of authenticity requirements



For every safety critical action in a system of systems all information that is used in the reasonig process that leads to this action has to be authentic

Security mechanism independence

avoid to break down the overall security requirements to requirements for specific components or communication channels prematurely
$\leadsto$ requirements are independent of decisions on concrete security enforcement mechanisms and structure (e.g. hop-by-hop, end-to-end)

Formal base approach fits to formal definition of security requirements

- Authenticity: A set of actions $\Gamma \subseteq \Sigma$ is authentic for $P \in \mathbb{P}$ after a sequence of actions $\omega \in S$ with respect to W_P if $alph(x) \cap \Gamma \neq \emptyset$ for all $x \in \lambda_P^{-1}(\lambda_P(\omega)) \cap W_P$.

Further Work w.r.t. EVITA Security Requirements Engineering

- Description of Security Engineering Process
- Attack trees
- Further security requirements w.r.t.
  - Integrity,
  - Controlled access,
  - Freshness,
  - Non-repudiation,
  - Anonymity, Privacy, Confidentiality,
  - Availability
- Risk Analysis
  - security threat severity classification
  - probability of successful attacks

`http://www.evita-project.org/Deliverables/EVITAD2.3.pdf`

References

Andreas Fuchs and Roland Rieke.
Identification of Authenticity Requirements in Systems of Systems by Functional Security Analysis.
In *Workshop on Architecting Dependable Systems (WADS 2009), in Proceedings of the 2009 IEEE/IFIP Conference on Dependable Systems and Networks, Supplementary Volume*, 2009.
http://sit.sit.fraunhofer.de/smv/publications/.

Andreas Fuchs and Roland Rieke.
Identification of Security Requirements in Systems of Systems by Functional Security Analysis.
In C. Gacek A. Casimiro, R. de Lemos, editor, *Architecting Dependable Systems 7*. Springer, to appear.

Alastair Ruddle, David Ward, Benjamin Weyl, Sabir Idrees, Yves Roudier, Michael Friedewald, Timo Leimbach, Andreas Fuchs, Sigrid Gürgens, Olaf Henniger, Roland Rieke, Matthias Ritscher, Henrik Broberg, Ludovic Apvrille, Renaud Pacalet, and Gabriel Pedroza.
Security requirements for automotive on-board networks based on dark-side scenarios.
EVITA Deliverable D2.3, EVITA project, 2009.
http://evita-project.org/deliverables.html.

Thank You

“It seems unarguable that the key challenge facing modern ICT is the management of a transition from systems comprising many relatively isolated, small-scale elements to large-scale, massively interconnected systems that are physically distributed yet must remain secure, robust, and efficient.”

Seth Bullock and Dave Cliff, Complexity and Emergent Behaviour in ICT Systems, HP Laboratories Bristol, 2004

Legal Requirements for a secure on-board architecture

Christophe Geuens

Researcher, ICRI-K.U.Leuven-IBBT

Zusammenfassung

The presentation is intended to provide a general overview of the legal requirements regarding a secure on-board architecture. The starting point will be the relevant legislation. This will serve as a frame of reference for the requirements. The legislation discussed will concern product safety, product liability and data protection rules. With regard to product safety the Motor Vehicle Directive and the General Product Safety Directive will be discussed. The goal of these Directives is to prevent unsafe products from entering the market. Associated to that they also implement a series of measures for notification in case safety issues relating to a product were to surface. With regard to product liability the product liability directive and tort law will be discussed. These deal with compensation for damage caused by defective products. Because of differing scopes they have a different field of application. For data protection attention will be paid to the Data Protection Directive. The impact of that Directive on an on-board architecture is the main issue to be discussed. Most important is that the Data Protection Directive does not impose any requirements on the architecture but rather on those implementing the architecture.

CV

Christophe Geuens (°1982) obtain ed his law degree at K .U.Leuven in 2007. As a student h e worked on issues at the intersection between criminal law a nd the use of GPS. He joined ICRI in May 2008. His main field of expertise i s l iability law, contract l aw and privacy and data protection law. With r egard to pr ivacy and data protect ion law he mainly focuses on th e problems regarding tracing and use of data by law enforcement.

Currently h e is working on projects related to Intelligent Transport systems and Automotive Applications. He is active on FP7-EVITA t hat is aiming at developing a secur e on-board architecture. He is work ing on liability and dat a protection issues involved. Amo ng his past Projects is I BBT-NextGenITS. In IBBT-Next GenITS he worked on liabilit y and privac y and data protection issues of ITS. He mainly focused on the privacy implications of eTolling and eCall.

Since 2008 he is participating in the eSecurity Working Group of the eSafety Forum.

Kontakt

Christophe Geuens
Sint-Michielsstraat 6
B-3000 Leuven
Tel. +32 16 320-782
Fax. +32 16 325-438
eMail: christophe.geuens@law.kuleuven.be

Literatur

[1] Guidance document on the relationship between the general product safety directive and certain sector directives with provisions on product safety, website DG SANCO: http://ec.europa.eu/consumers/safety/rapex/key_docs_en.htm.

[2] G., HOWELLS, "Europe's solution to the product liability phenomenon", *Anglo-Am. L. Rev.* 1991, 209

[3] A., STOPPA, "The concept of defectiveness in the Consumer Protection Act 1987: a critical analysis", *Legal Stud.* 1992, 211.

[4] E., KOSTA, P., VALCKE, "Retaining the data retention directive", *Computer law and security report*, nr. 22, p 374.

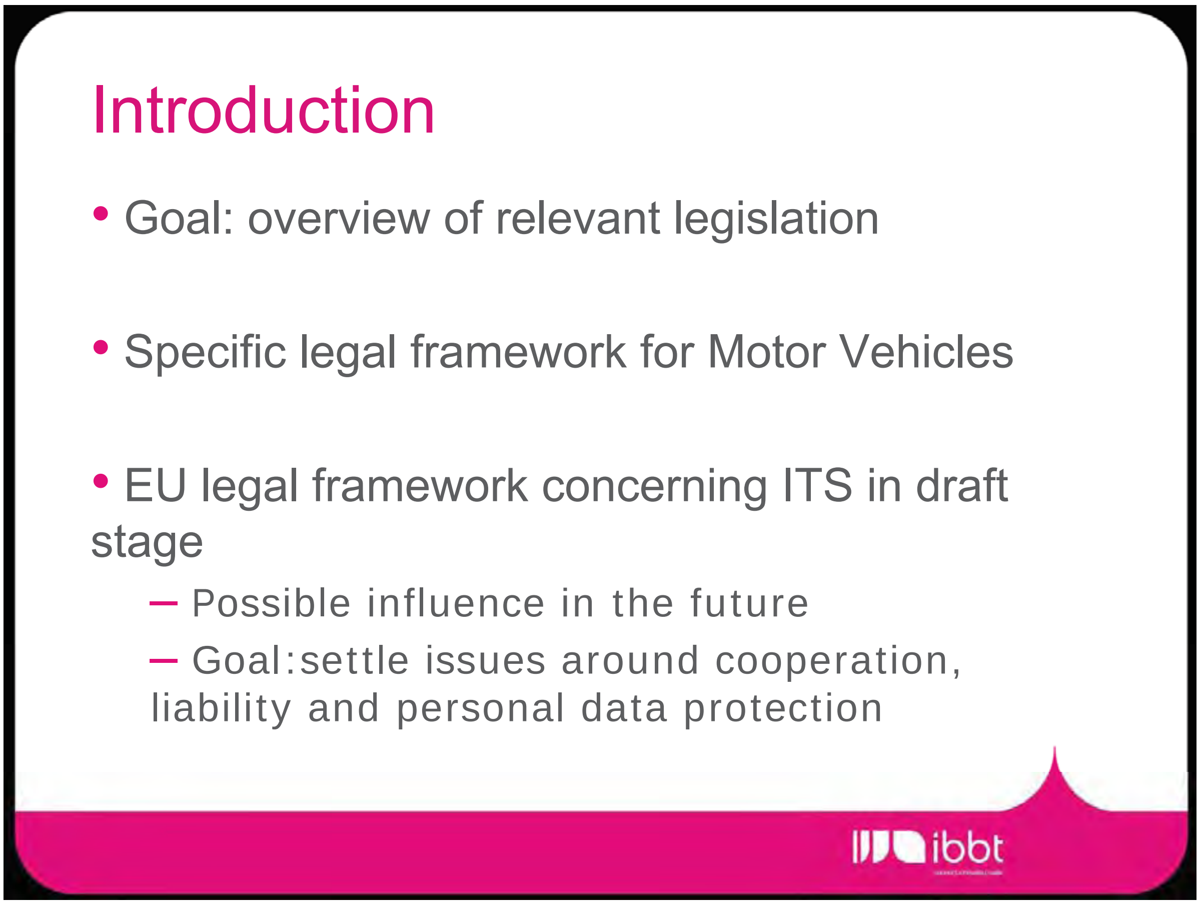
[5] B.,DOCQUIR, "le droit a la vie privée", de boeck Larcier, Brussels, 2008

Legal Requirements for a secure on-board architecture

Christophe Geuens
Researcher
ICRI-K.U.Leuven-IBBT
01/07/2010 Darmstadt

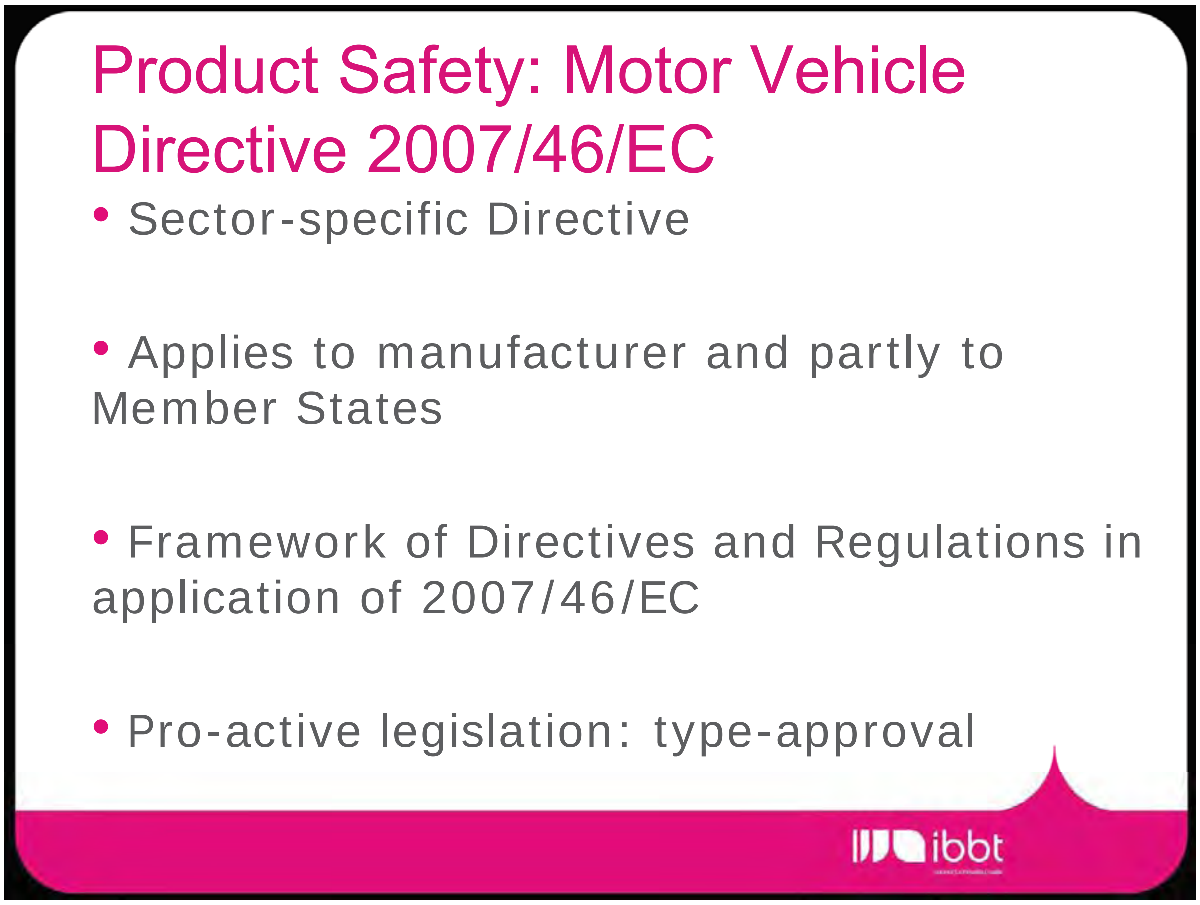
What's on today's agenda?

- Introduction
- Product safety and product liability
  - Motor Vehicle Directive
  - General Product Safety Directive
  - Product liability
- Data Protection
  - Data Protection Directive



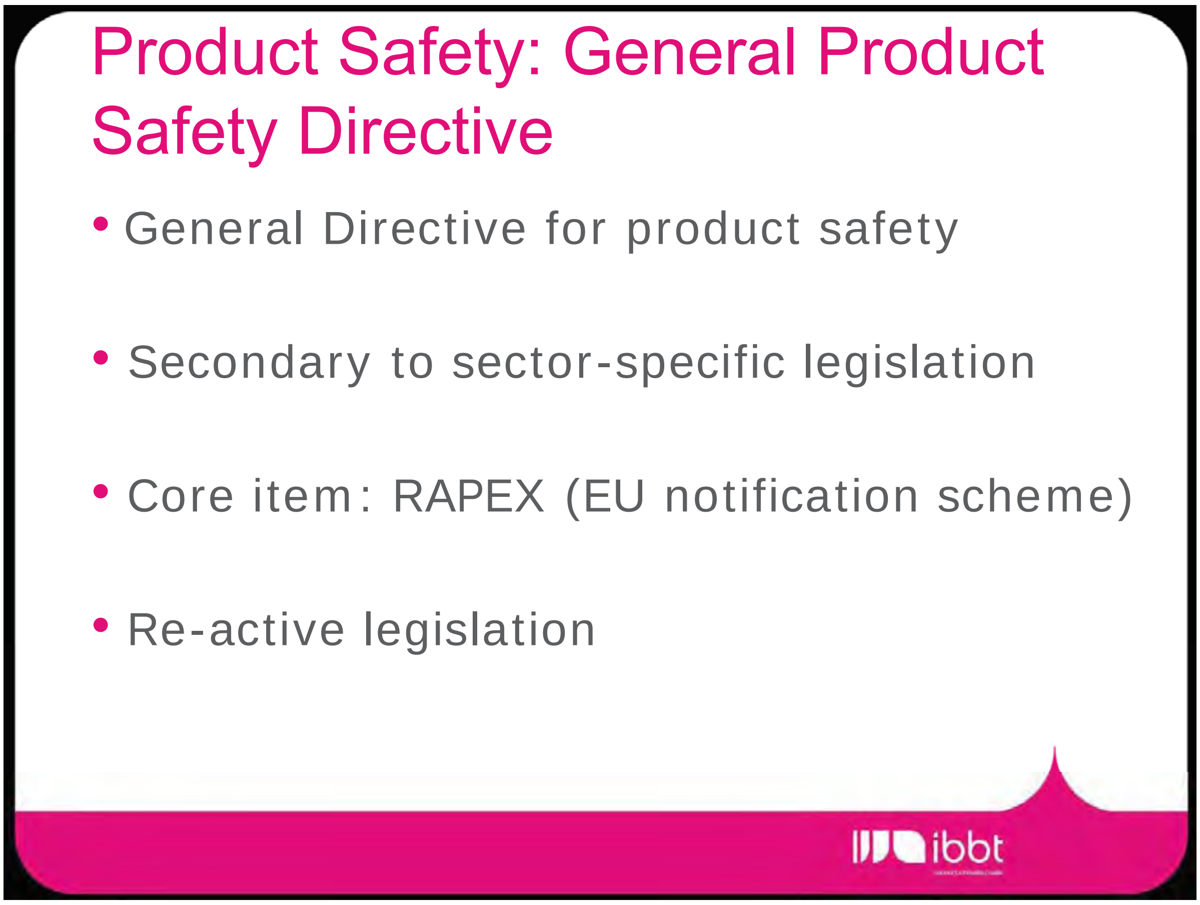
Introduction

- Goal: overview of relevant legislation
- Specific legal framework for Motor Vehicles
- EU legal framework concerning ITS in draft stage
  - Possible influence in the future
  - Goal:settle issues around cooperation, liability and personal data protection



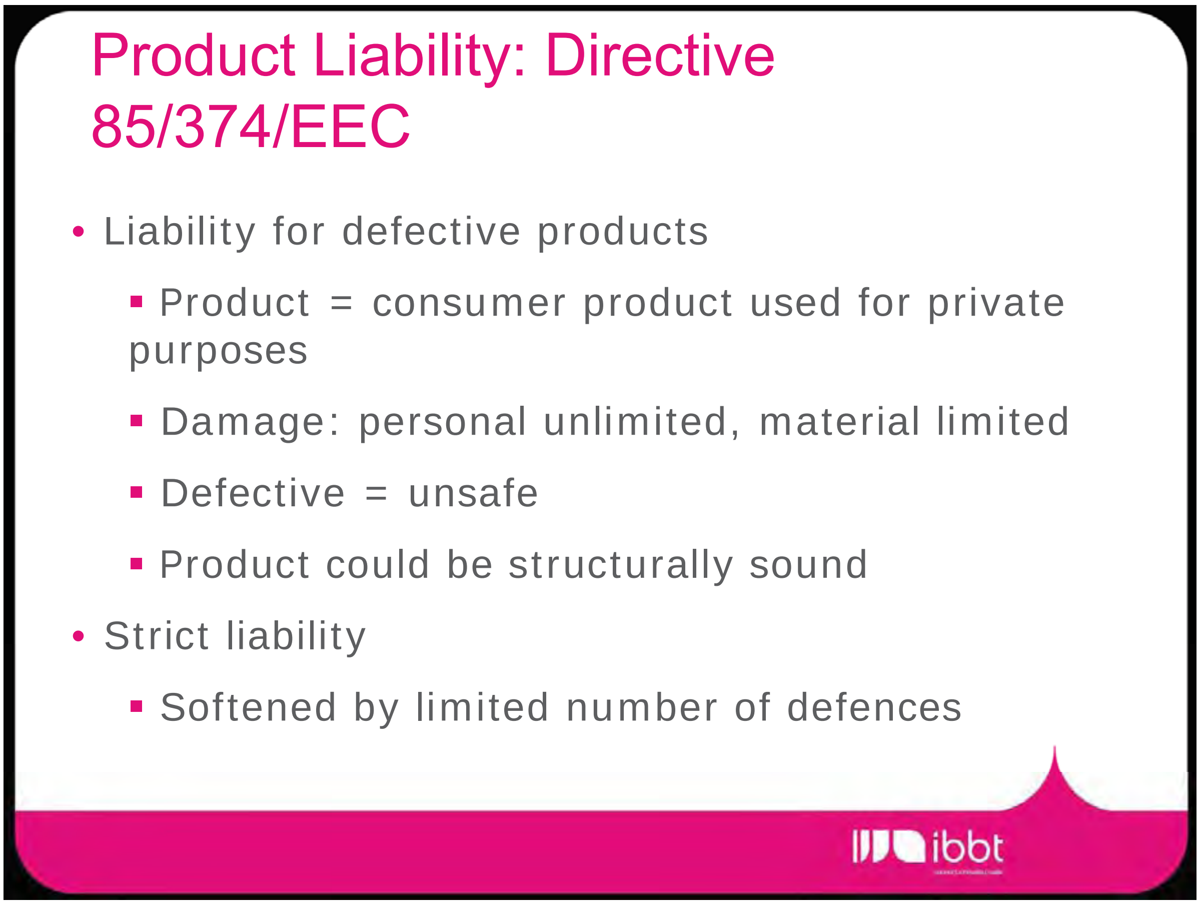
Product Safety: Motor Vehicle Directive 2007/46/EC

- Sector-specific Directive
- Applies to manufacturer and partly to Member States
- Framework of Directives and Regulations in application of 2007/46/EC
- Pro-active legislation: type-approval



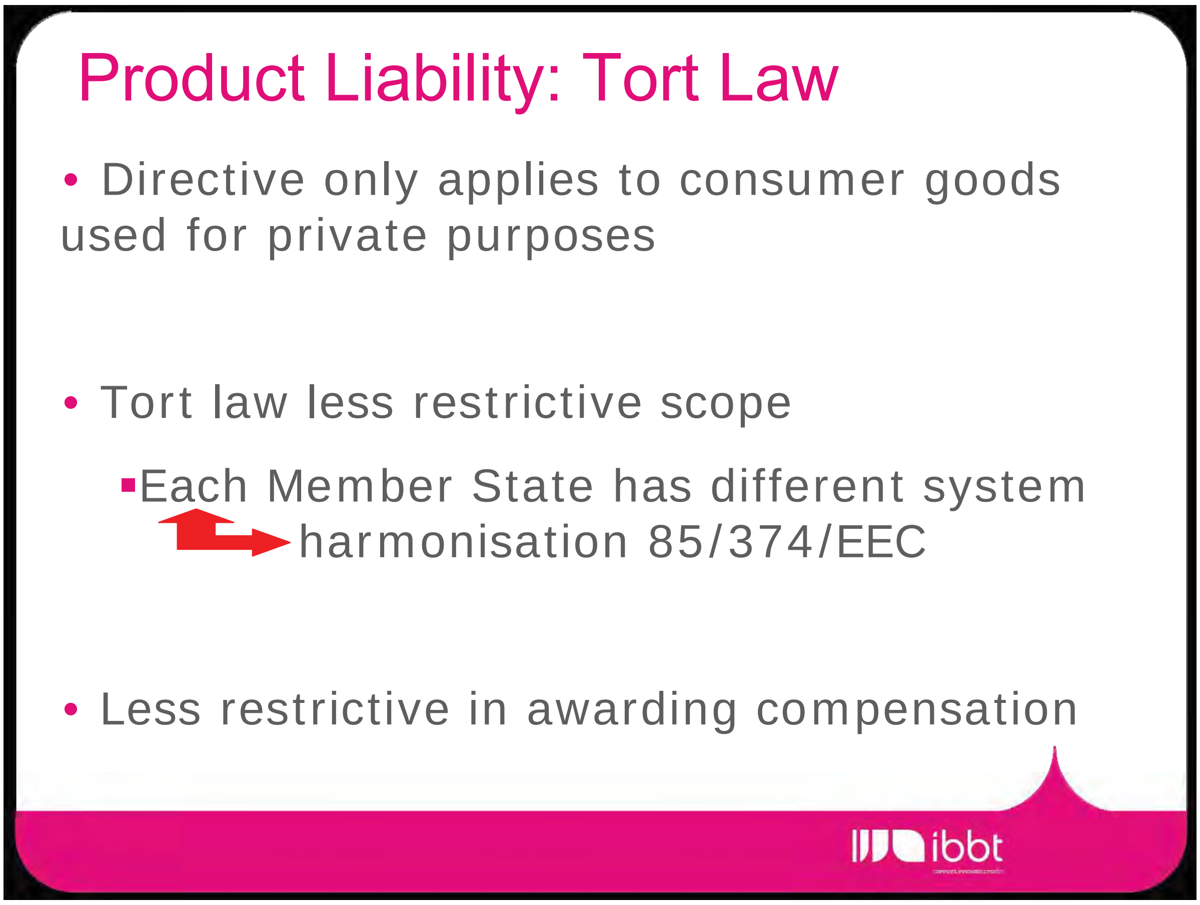
Product Safety: General Product Safety Directive

- General Directive for product safety
- Secondary to sector-specific legislation
- Core item: RAPEX (EU notification scheme)
- Re-active legislation



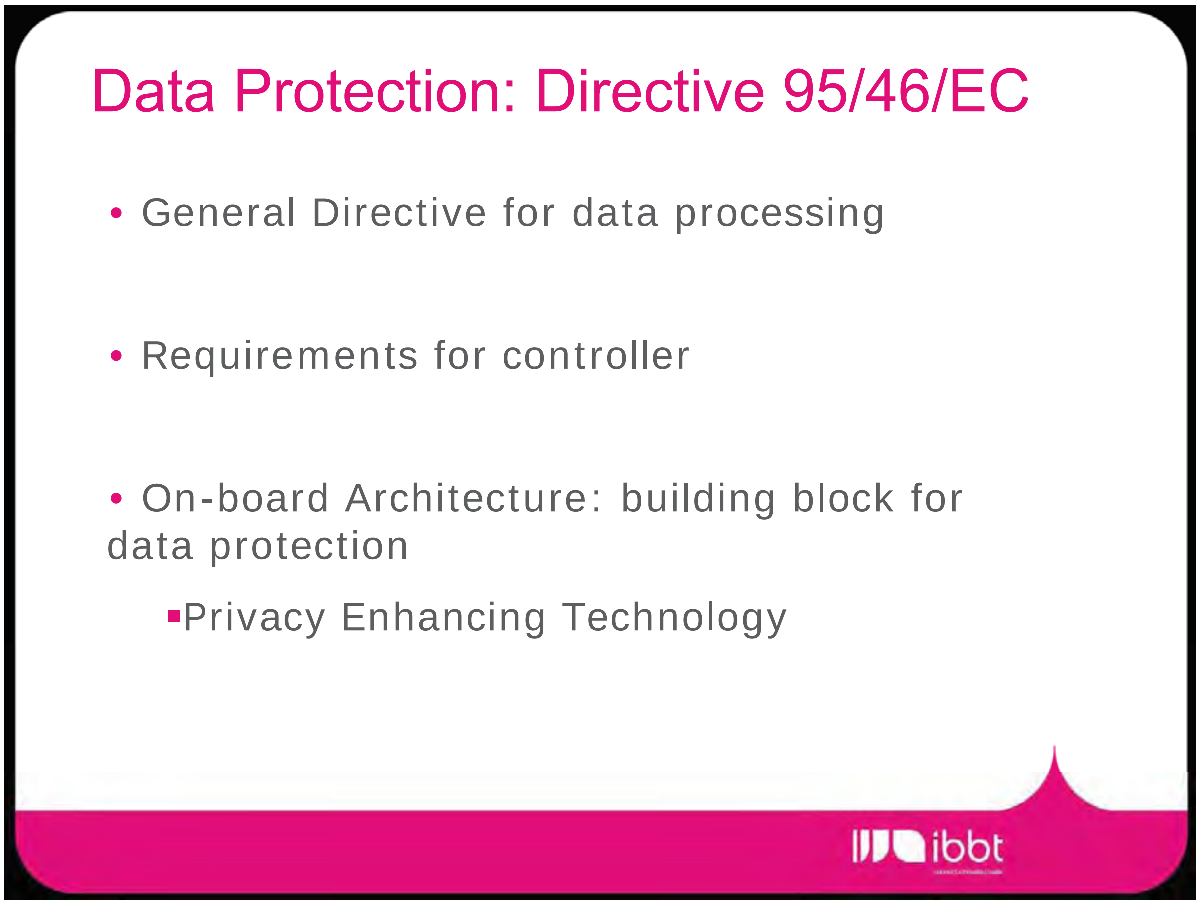
Product Liability: Directive 85/374/EEC

- Liability for defective products
  - Product = consumer product used for private purposes
  - Damage: personal unlimited, material limited
  - Defective = unsafe
  - Product could be structurally sound
- Strict liability
  - Softened by limited number of defences



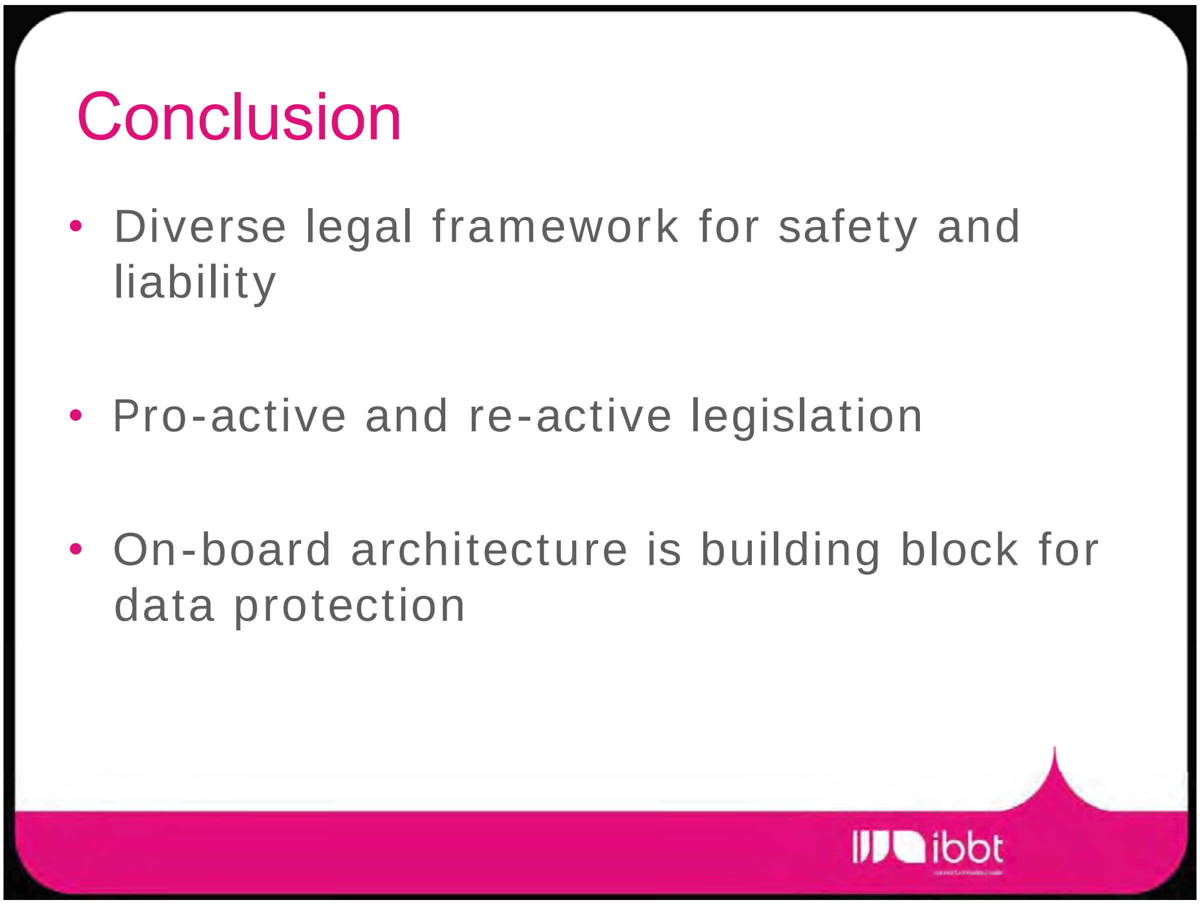
Product Liability: Tort Law

- Directive only applies to consumer goods used for private purposes
- Tort law less restrictive scope
  - Each Member State has different system → harmonisation 85/374/EEC
- Less restrictive in awarding compensation



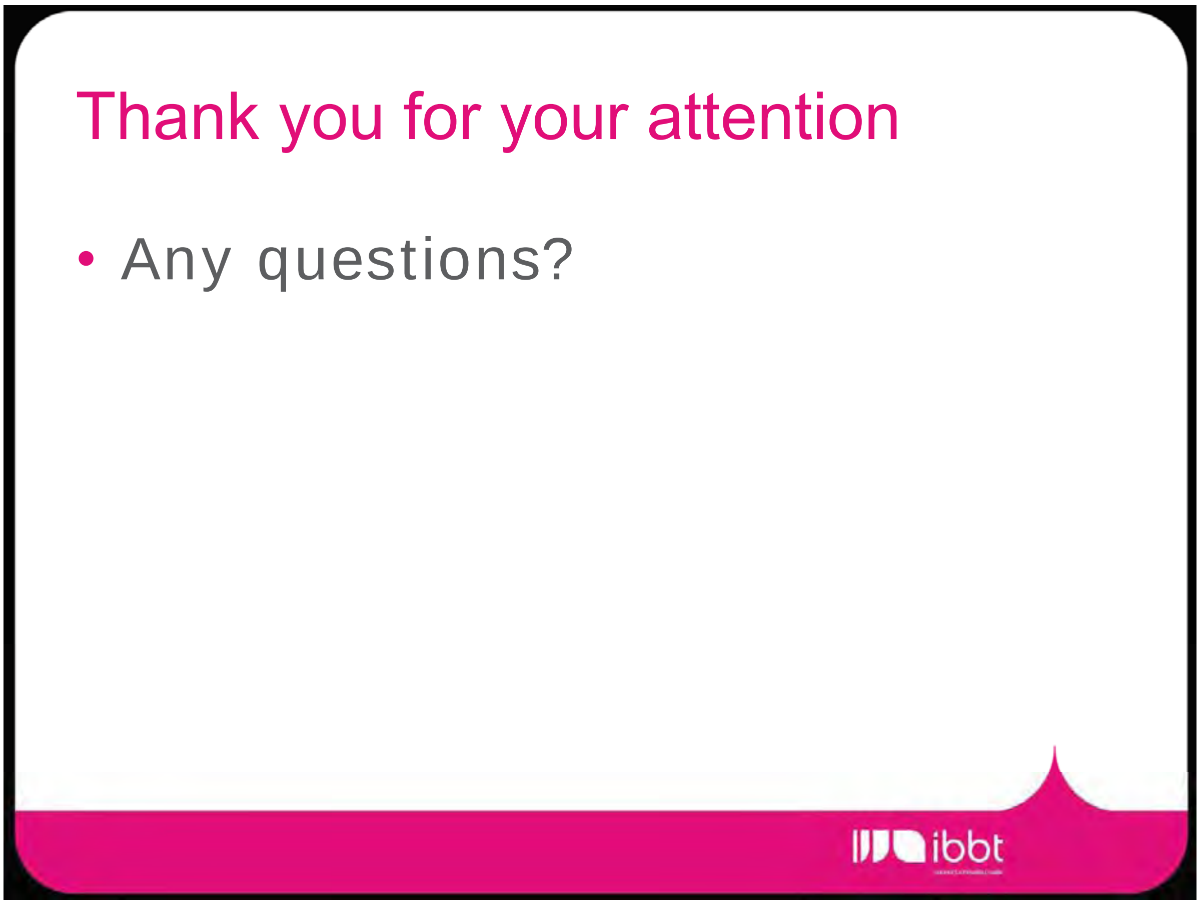
Data Protection: Directive 95/46/EC

- General Directive for data processing
- Requirements for controller
- On-board Architecture: building block for data protection
  - Privacy Enhancing Technology



Conclusion

- Diverse legal framework for safety and liability
- Pro-active and re-active legislation
- On-board architecture is building block for data protection



Thank you for your attention

- Any questions?

Christophe Geuens

Researcher

http://be.linkedin.com/in/christophegeuens

t	+32 16320782
e	Christophe.Geuens@Law.Kuleuven.be
www	www.law.kuleuven.be/icri
www	www.ibbt.be

Interdisciplinary Center for Law and ICT
Sint-Michielsstraat 6
B-3000 Leuven
Belgium

ibbt

The EVITA Hardware Security Module (HSM)

Interface Specification and Basic Hardware Security Functionality

Marko Wolf

Senior Security Engineer, escrypt GmbH – Embedded Security, Munich

Summary

The need for vehicular hardware security mea sures is no w undispute d [1]. In order to ensure the security of in-vehicle security mechanisms, we need an appropriate protect ed hardware security anchor that is capable to withstand even physical in-vehicle attackers acco rdingly. The hardware security anchor protects security me chanisms by enabling se cure generation, secure storage, an d secure p rocessing o f all securit y-critical material, while being shielded from potential maliciou s intr usions with the help of hardware protection measures that require significant technical and financial efforts to become compromised.

This contri bution will give an insight into t he interface specifi cation and ba sic se curity functionality of the hardware security module (HSM) de veloped by the EVITA project [2]. Therefor, the talk first shortly recaps, why ha rdware security measu res are essential fo r ensuring ve hicular IT security. It th en presents the general system architecture of the EVITA approach with focus on the underlying hardware security architecture(s). The talk introduces the corresponding security building blocks and security functionality of the EVITA HSM specification and gives some descriptive usage exa mples. The presentation closes with some remarks on the already ongoing implementation.

CV

Dr.-Ing. Ma rko Wolf is senior eng ineer at escrypt – Embedded Security Gmb H and there primarily concerned with automotive IT securit y. Wolf has studied ele ctrical engin eering and information technology at Purdue University (USA) and Ruhr-University Bochum (German y). After he received his M.Sc. in 200 3, he starte d his PhD in the area of automotive security , trusted computing, and secure oper ating syste ms at the Chair for Embedded Security hold by Prof. Dr. Christof Paar. Wolf completed his PhD in 2008 with the first comprehensive work about vehicular se curity engineering. He is author of the book “Security Engineering for Vehicular IT Systems” [3], editor of the book “Embedded Security in C ars: Securin g Current a nd Future Automotive IT Applications” [4] and has written over 30 international publications.

Contact

escrypt GmbH – Embedded Security
Leopoldstraße 244
80804 München
Tel. +49 89 802039-131
E-Mail: marko.wolf@escrypt.com
Internet: www.marko-wolf.de

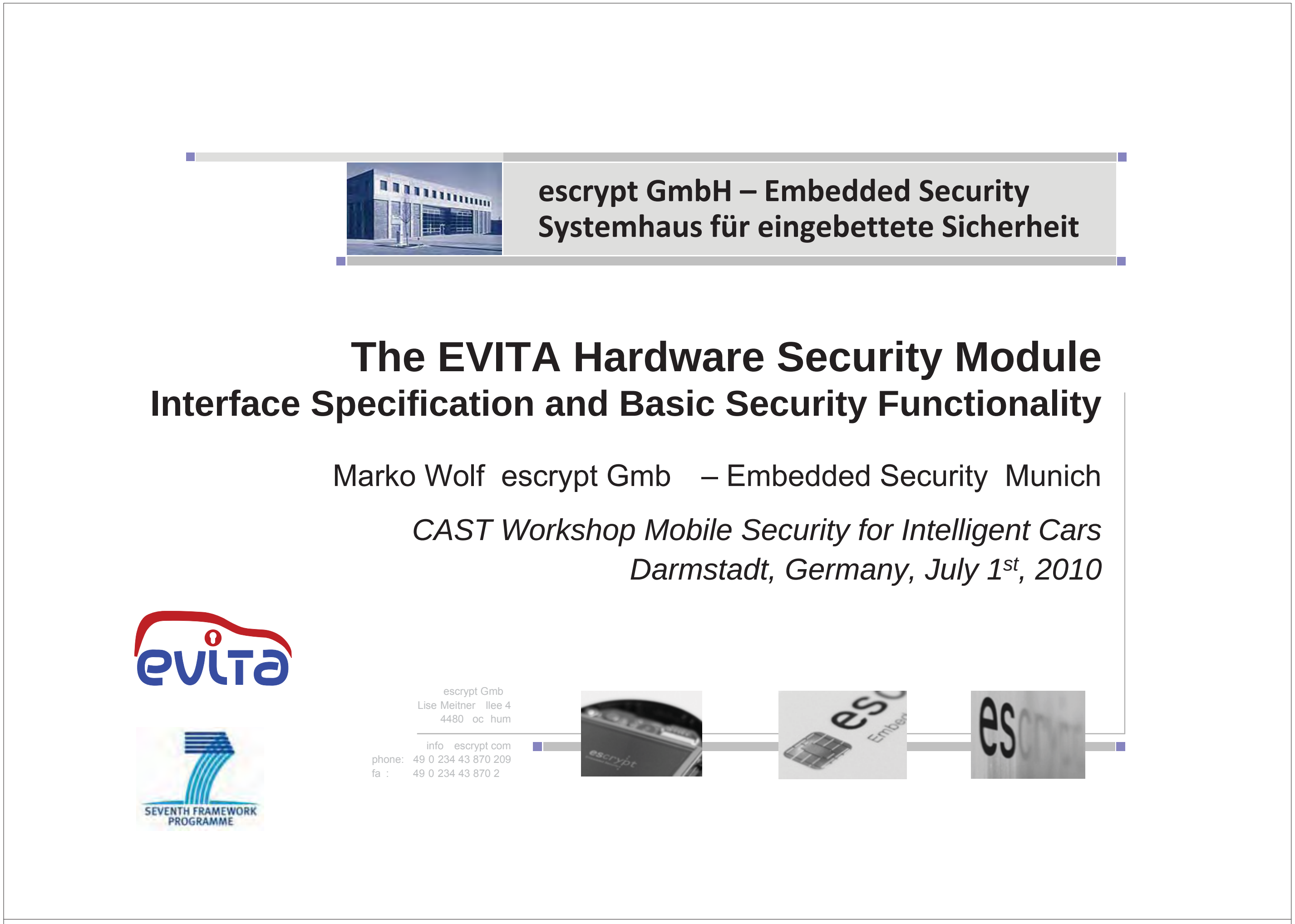
References

[1] A.Weimerskirch, M.Wolf, T.Wollinger: "State of the Art: Embedding Security in Vehicles", In EURASIP Journal on Embedded Systems (EURASIP JES), Special Issue: Embedded Systems for Intelligent Vehicles, 2007.

[2] EVITA: E-safety vehicle intrusion protected applications. www.evita-project.org, 2008.

[3] M.Wolf: "Security Engineering for Vehicular IT Systems — Improving Trustworthiness and Dependability of Automotive IT Applications", Vieweg+Teubner-Verlag, 2009.

[4] K.Lemke, C.Paar, M.Wolf (Eds.): " Embedded Security in Cars — Securing Current and Future Automotive IT Applications", Springer-Verlag, 2006.



escrypt GmbH – Embedded Security
Systemhaus für eingebettete Sicherheit

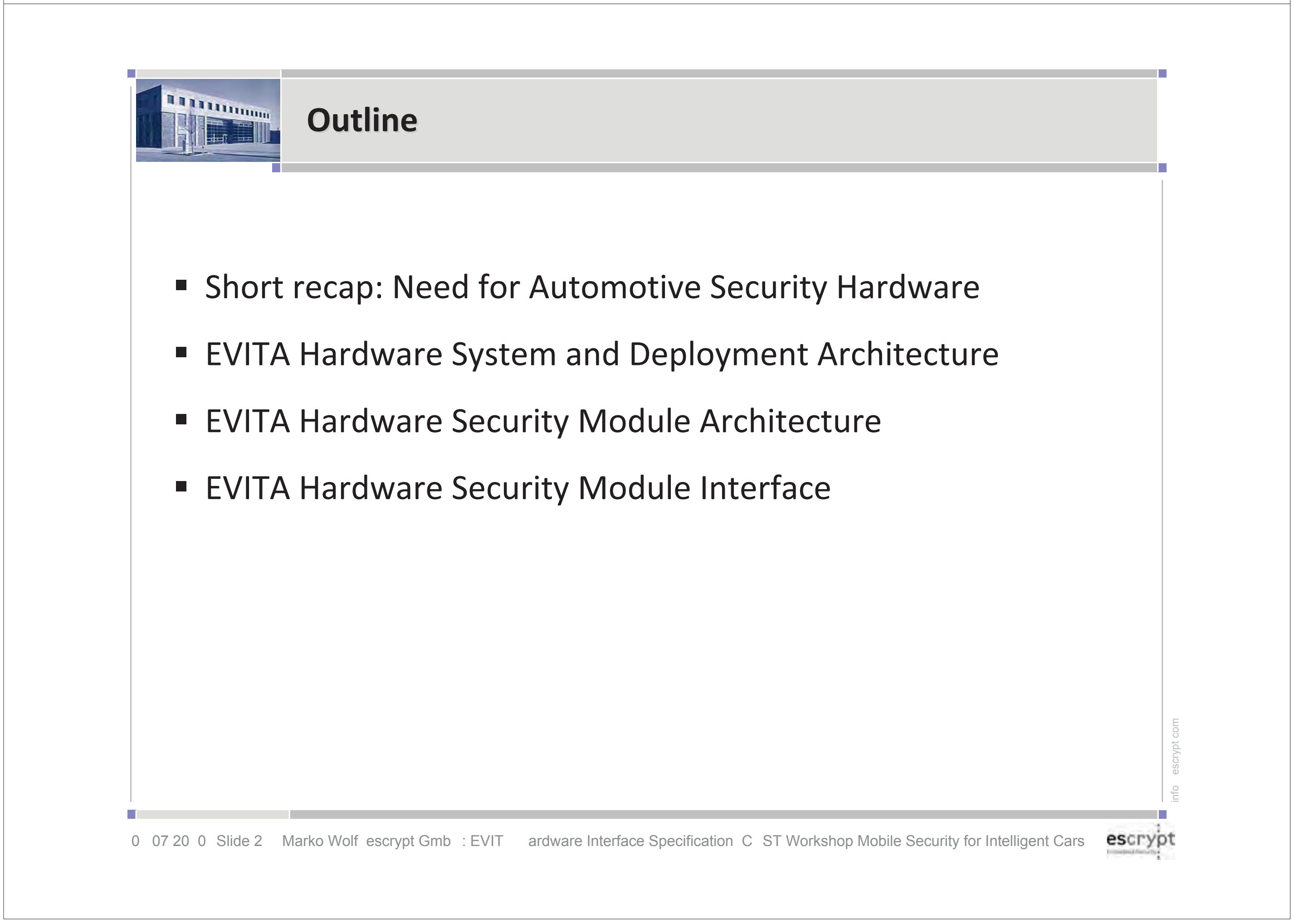
The EVITA Hardware Security Module
Interface Specification and Basic Security Functionality

Marko Wolf escrypt Gmb   – Embedded Security Munich

CAST Workshop Mobile Security for Intelligent Cars
Darmstadt, Germany, July 1st, 2010

escrypt Gmb
Lise Meitner llee 4
4480 oc hum

info escrypt com
phone: 49 0 234 43 870 209
fa : 49 0 234 43 870 2



Outline

- Short recap: Need for Automotive Security Hardware
- EVITA Hardware System and Deployment Architecture
- EVITA Hardware Security Module Architecture
- EVITA Hardware Security Module Interface

info escrypt com

0 07 20 0 Slide 2   Marko Wolf escrypt Gmb : EVIT   ardware Interface Specification C ST Workshop Mobile Security for Intelligent Cars

Outline

- Short recap: Need for Automotive Security Hardware
- EVITA Hardware System and Deployment Architecture
- EVITA Hardware Security Module Architecture
- EVITA Hardware Security Module Interface

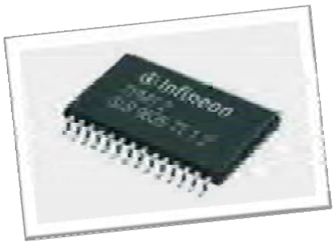
Short Intro: Recap on Vehicular Hardware Security

- Vehicular attacks are beyond "standard attacks" ..
  - o **Insider** attacks
    - Attacker can be also legitimate owner w/ extended access rights (e.g., physical access)
    - Attacker can prevent emergency protection measures or security updates
    - Attacker seldom has to fear non-technical protection measures (e.g., legal penalties)
  - o **Offline** attacks
    - Attacker has virtually unlimited time
    - Attacker has virtually unlimited trials
    - Attacker and attack are hard to detect

  - o **Physical** attacks
    - Asset manipulations or read-outs via debug interfaces, probing, side-channels etc.
    - Disabling, manipulating of any (physical) inputs, outputs and processing
  - o **Logical** attacks
    - Little security-validated, but highly interconnected interfaces (even to outside world)
    - Little security-validated, but enormous amounts of (mainly safety-driven) software
    - Not seldom, proprietary and non-public security mechanisms (security by obscurity)
  - o ...

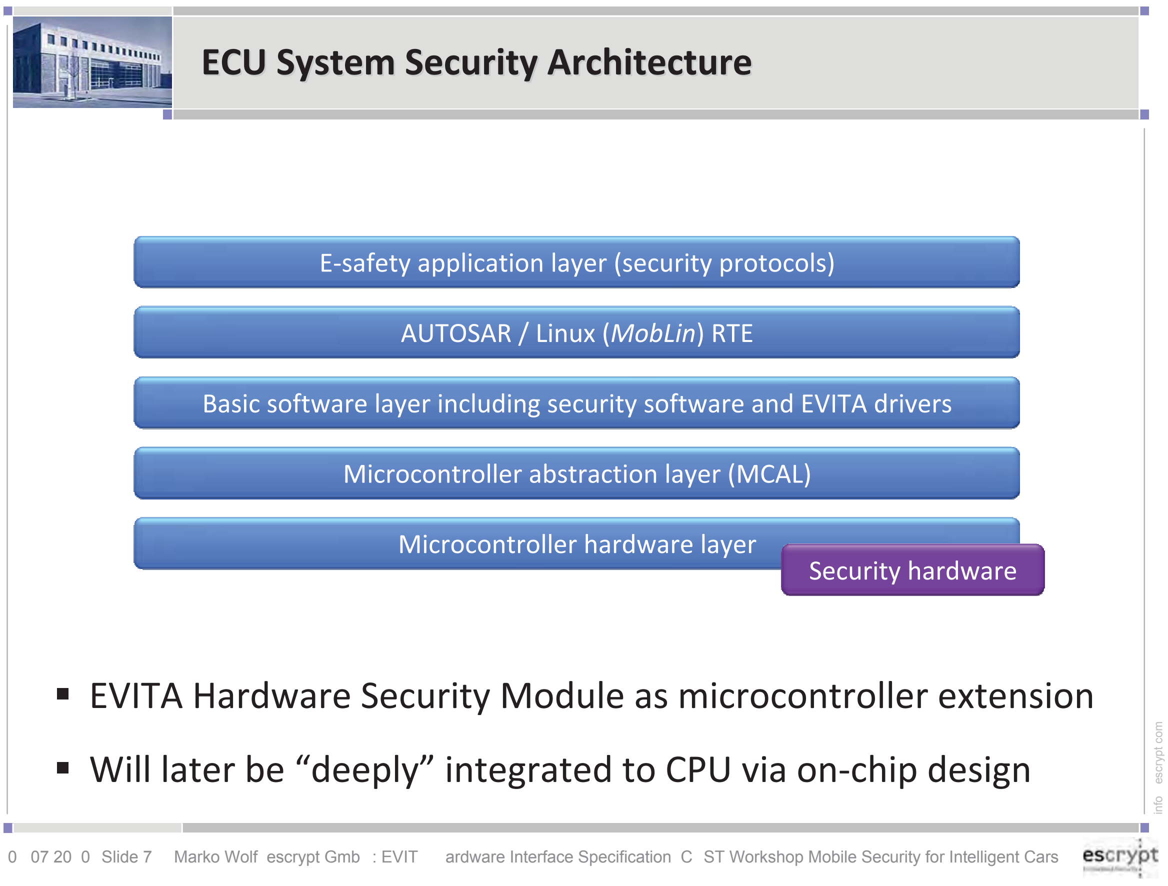
Short Intro: Recap on Vehicular Hardware Security

- **Protects** software security mechanisms by
  - ➲ Providing a trustworthy *security anchor* for upper SW layers
  - ➲ *Secure generation, secure storage, and secure processing* of security-critical material shielded from all pot. malicious SW
- **Prevents** hardware tampering attacks by
  - ➲ Applying *tamper-protection* measures
- **Accelerates** security mechanisms by
  - ➲ Applying *cryptographic accelerators*
- **Reduces** security costs on high volumes by
  - ➲ Applying highly optimized special circuitry instead of costly general purpose hardware



Outline

- Short recap: Need for Automotive Security Hardware
- EVITA Hardware System and Deployment Architecture
- EVITA Hardware Security Module Architecture
- EVITA Hardware Security Module Interface



ECU System Security Architecture

E-safety application layer (security protocols)

AUTOSAR / Linux (*MobLin*) RTE

Basic software layer including security software and EVITA drivers

Microcontroller abstraction layer (MCAL)

Microcontroller hardware layer

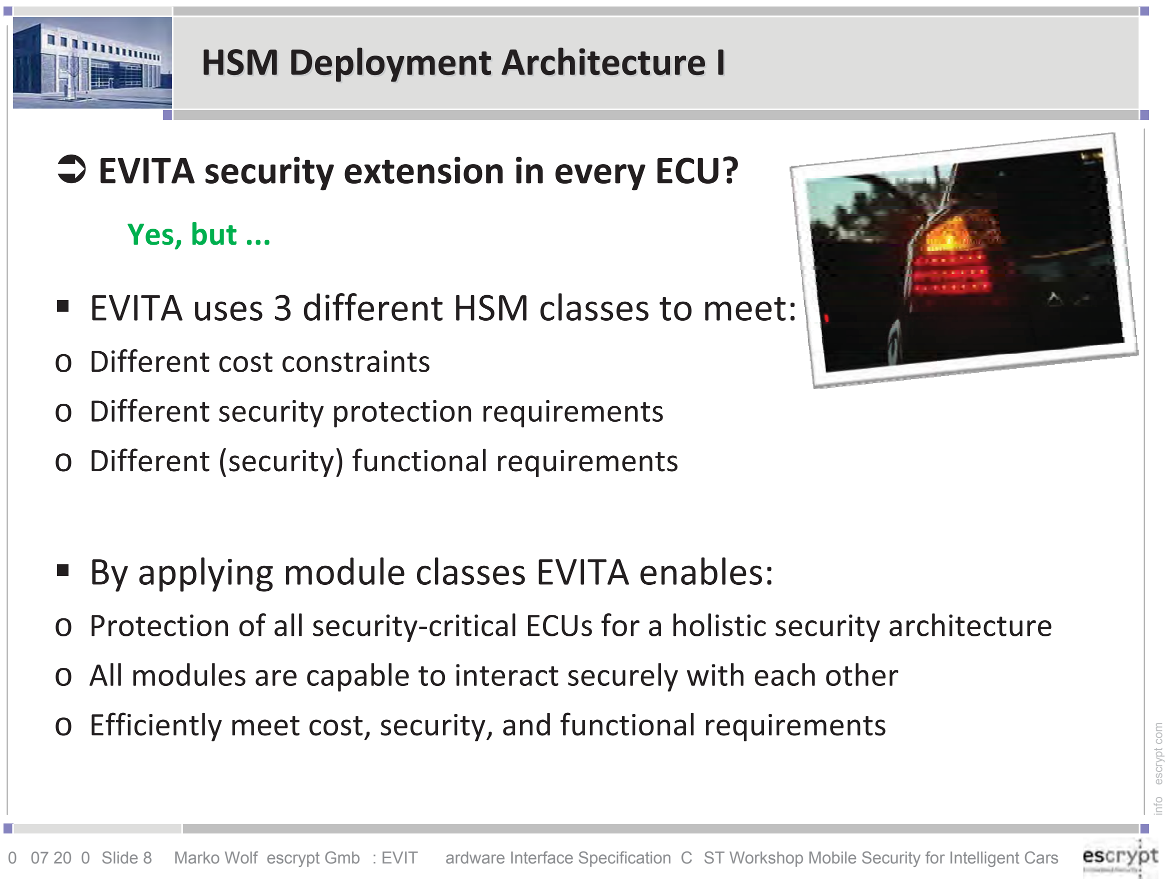
Security hardware

- EVITA Hardware Security Module as microcontroller extension
- Will later be "deeply" integrated to CPU via on-chip design

info escrypt com

0 07 20 0 Slide 7 Marko Wolf escrypt Gmb : EVIT ardware Interface Specification C ST Workshop Mobile Security for Intelligent Cars

escrypt



HSM Deployment Architecture I

➲ **EVITA security extension in every ECU?**

Yes, but ...

- EVITA uses 3 different HSM classes to meet:
  - o Different cost constraints
  - o Different security protection requirements
  - o Different (security) functional requirements

- By applying module classes EVITA enables:
  - o Protection of all security-critical ECUs for a holistic security architecture
  - o All modules are capable to interact securely with each other
  - o Efficiently meet cost, security, and functional requirements

info escrypt com

0 07 20 0 Slide 8 Marko Wolf escrypt Gmb : EVIT ardware Interface Specification C ST Workshop Mobile Security for Intelligent Cars

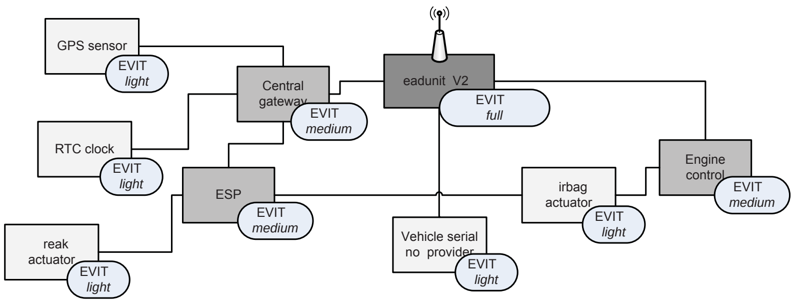
escrypt

HSM Deployment Architecture II

- EVITA *full* feat. module in 1 – 2 high-performance comm. ECUs
  - V2X communication unit (head unit)
  - Central gateway (possibly)
- EVITA *medium* feat. module in 2 - 4 central multi-purpose ECUs
  - Engine control
  - Front/rear module
  - Immobilizer
- EVITA *small* feat. in less, but security-critical ECUs
  - Critical sensors: e.g., wheel, acceleration, pedal sensors
  - Critical actuator: e.g., breaks, door locks, turn signal indicator
  - Critical small controllers: e.g., GPS module, lighting, clock

HSM Deployment Architecture III

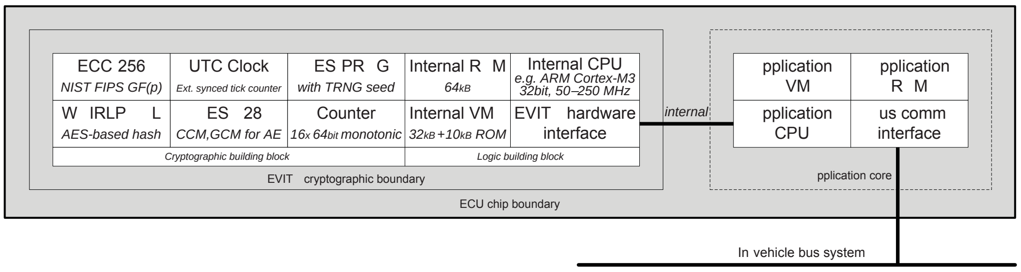
- **Efficient**, **cost-effective**, **flexible**, and **holistic** in-vehicle EVITA HSM deployment regarding the different cost, performance constraints and functional requirements



Outline

- Short recap: Need for Automotive Security Hardware
- EVITA Hardware System and Deployment Architecture
- EVITA Hardware Security Module Architecture
- EVITA Hardware Security Module Interface

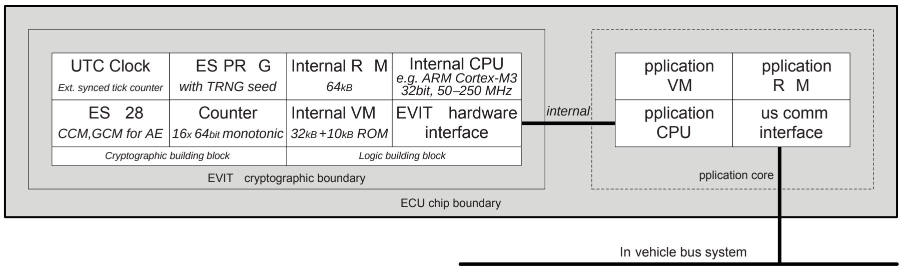
EVITA Full HSM Architecture



- **ECC-256-GF(p)**: High-performance 256-bit standardized elliptic curve arithmetic*) that can generate and verify ≈ **250 signatures/s**
- **WHIRLPOOL**: Generic hash function (allows ASIC w/ **SHA-3**) actually using AES-based **NIST standardized** hash function with ≈ **1 Gbit/s** throughput
- **AES-128**: Symmetric **NIST standard** ECB/CBC block encryption/decryption but also advanced **AE modes** e.g. GCM/CCM with ≈ **1 Gbit/s** throughput
- **AES-PRNG**: PRNG using a **true random seed** based an internal AES engine according to **BSI-AIS20 standard** with ≈ **500 Mbit/s** throughput
- **COUNTER**: 16 x 64-bit monotonic counters at 1 Hz to act as **“secure clock”**

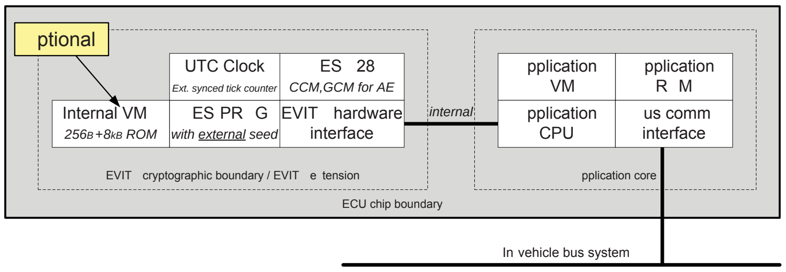
**) Pure GF(p) arithmetics only, so the curve parameters can be changed easily.*

EVITA Medium HSM Architecture



- Designed to **suit both**: stringent **security** requirements and significant **cost pressures** of powerful multi-purpose ECUs (e.g., engine control, immobilizer)
- Virtually identical to the EVITA *full* version except in that it has **no dedicated ECC hardware** and **no dedicated hash function hardware**
- Very **fast symmetric cryptography** in hardware, but rather slow (i.e., software based) – but nonetheless practicable – asymmetric cryptography
- Meets **all in-vehicle security** use cases, but not suitable for V2X

EVITA Light HSM Architecture



- **Integrates** and protects **small ECUs**, **sensors** and **actuators** that provide or process security critical information (e.g., pedals, lighting, GPS)
- Reduced to a single very **cost-optimized symmetric AES hardware** accelerator (i.e., all security credentials are handled by the application processor)
- Cannot provide any hardware-based security (i.e. attacks from application core), but enables sensors and actuators to **efficiently process and generate protected information**

Hardware Security Module Architecture

- Overview and comparison with other HSMs availble

HSM	EVITA full	EVITA medium	EVITA light	SHE	TPM	Usual smartcard
Boot integrity protection	Auth. & Secure	Auth. & Secure	Auth. & Secure	Secure	Auth	None
HW crypto algorithms (incl. key generation)	ECDSA,ECDH, AES/MAC, WHIRLPOOL/HMAC	ECDSA,ECDH, AES/MAC, WHIRLPOOL/HMAC	AES/MAC	AES/MAC	RSA, SHA-1/ HMAC	ECC, RSA, AES,3DES, MAC, SHA-x..
HW crypto acceleration	ECC,AES, WHIRLPOOL (could be even FPGA)	AES	AES	AES	None	None
Internal CPU	Programmable	Programmable	None	None	Preset	Programmable
RNG	TRNG	TRNG	PRNG w/ ext. seed	PRNG w/ ext. seed	TRNG	TRNG
Counter	16x64bit	16x64bit	None	None	4x32bit	None
Internal NVM	Yes	Yes	Optional	Yes	Indirect (via SRK)	Yes
Internal Clock	Yes w/ ext. UTC sync	Yes w/ ext. UTC sync	Yes w/ ext. UTC sync	No	No	No
Parallel Access	Multiple sessions	Multiple sessions	Multiple sessions	No	Multiple sessions	No
Tamper Protection	Indirect (passive, part of ASIC)	Indirect (passive, part of ASIC)	Indirect (passive, part of ASIC)	Indirect (passive, part of ASIC)	Yes (mfr. dep.)	Yes (active, up to EAL5)

Outline

- Short recap: Need for Automotive Security Hardware
- EVITA Hardware System and Deployment Architecture
- EVITA Hardware Security Module Architecture
- EVITA Hardware Security Module Interface

HSM Hardware Interface: General Features

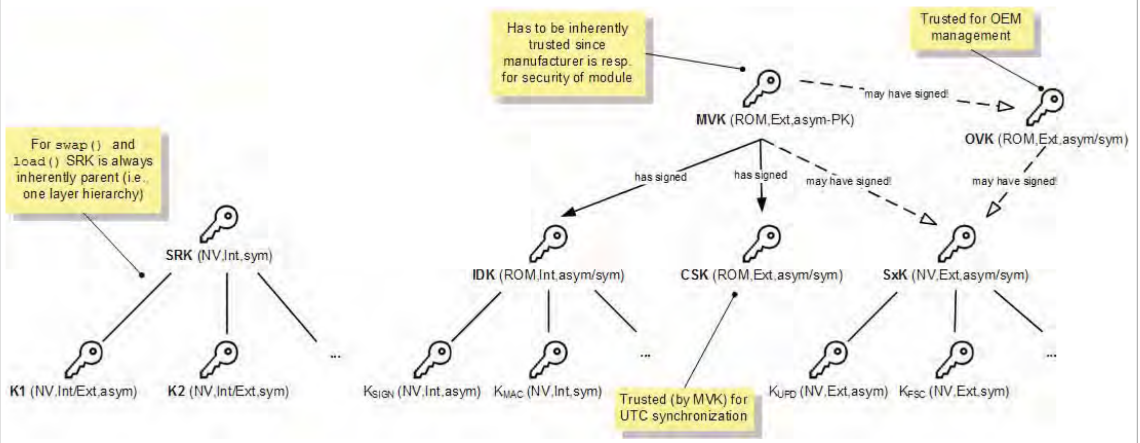
- **Asynchronous** (i.e., non-blocking) hardware interface
- **Multi-sessions** (i.e., interruptible) for most hardware security blocks (e.g., AES, MAC, digital signatures, and hash functions) via session identifier
- EVITA key uses can (but do not necessarily have to) have additional **individual authorizations** via:
  - o ***password*** given on function invocation (including failure counter)
  - o inherent ***bootstrap*** verification by verifying an bootstrap reference
  - o ***combination*** of password and bootstrap reference
- EVITA **understands all HIS** ("Herstellerinitiative Software") **SHE commands** (i.e., SHE compliance)

HSM Hardware Interface: General Constraints

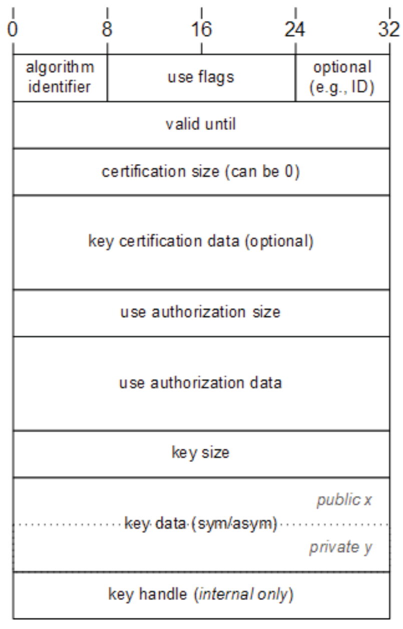
- Some **single-session** (i.e., non-interruptible) interface for some small hardware security blocks (e.g., RNG)
- **Single-thread per hardware block**, but **limited multi-threading** for different hardware blocks (e.g., one can call PRNG and AES in parallel)
- EVITA **commands are not** explicitly and **individually protected** at hardware level (but remember on-chip integration)
  - o i.e., they are in plain and w/o any replay and authenticity protection at hardware level
  - o in case this is required, we propose to a TPM-like approach (based on a simple user management) to establish a session key and "rotate nonces"
- EVITA has **no user management** at hardware level

HSM Hardware Interface: Internal Key Hierarchy

- **SRK** = Storage Root Key
- **MVK** = Module Manufacturer Verification Key
- **IDK** = Device Identity Key
- **CSK** = Clock Synchronization Key(s)
- **SxK** = Stakeholder Key (with x=**S** symmetric or x=**A** asymmetric)
- **OVK** = OEM Verification Key



Important HSM Data Structures: Internal Key Object



- Internal Key Object
  - o (asymmetric) algorithm identifier
  - o use flags = {`sign, verify, encrypt, decrypt, timestamp, secureboot, securestorage, dhke, utcsync, transport,` … } each with individual authorizations for usage and transportation
  - o can be time-limited
  - o can be certified by issuer
  - o usage authorization by password, bootstrap or combination of both
  - o individual key data structure (depending on algorithm identifier)
  - o internal key handle for reference

Important HSM Data Structures: Internal Key Object

use authorization size
use authorization data

use_flag	trnsp_flag	auth_flag	auth_value
..	..	..	..
encrypt	int	pw	H_x = H(„abc")
decrypt	int	ecr	H_x = ECR(1)
sign	mig	ecrpw	H_x = ECR(1) ⊗ H(„abc")
verify	ext	none	H_x = Ø

Exemplary SRK

use_flag	trnsp_flag	auth_flag	auth_value
encrypt	int	pw	H_x = H(„abc")
decrypt	int	ecr	H_x = ECR(1)

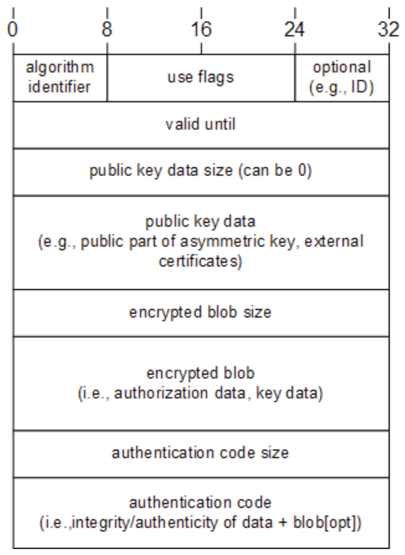
Exemplary MAC key (Master)

use_flag	trnsp_flag	auth_flag	auth_value
sign	int	pw	H_x = H(„abc")
verify	mig	ecr	H_x = ECR(1)

Exemplary MAC key (Client)

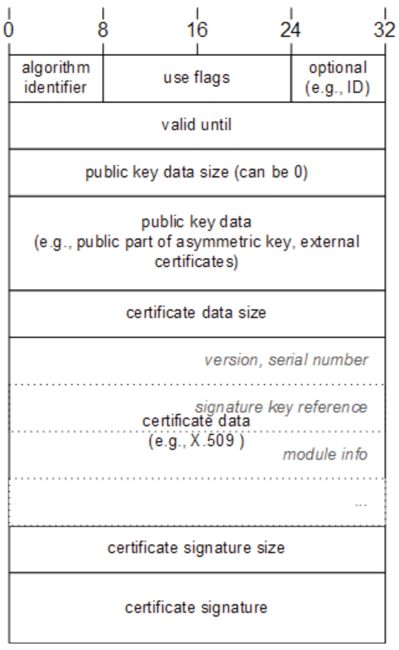
use_flag	trnsp_flag	auth_flag	auth_value
verify	mig	ecr	H_x = ECR(1)

Important HSM Data Structures: External Key Object


0
8
16
24
32
algorithm identifier
use flags
optional (e.g., ID)
valid until
public key data size (can be 0)
public key data (e.g., public part of asymmetric key, external certificates)
encrypted blob size
encrypted blob (i.e., authorization data, key data)
authentication code size
authentication code (i.e.,integrity/authenticity of data + blob[opt])

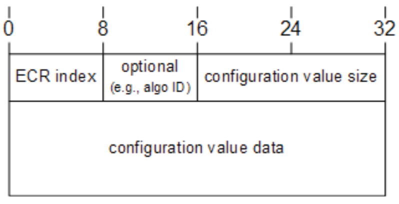
- External Key Object
  - o used only for transport, migrate or secure storage swapping
  - o key needs corresponding transport rights and authorizations (if set)
  - o algorithm, usage flags and validity interval are fully visible
  - o public key data is fully visible
  - o encrypted key blob = encrypted key internals such as key authorizations and private key parts
  - o fully visible authentication code (MAC/Sig) for key object integrity and authenticity protection

Important HSM Data Structures: Key Certificate

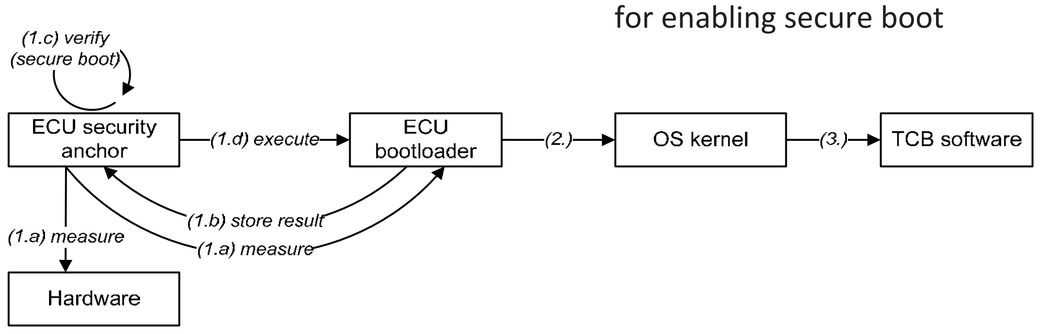


- Key Certificate
  - Used to certify public key information of internal keys (asymmetric and symmetric)
  - Certification signature is done with own device identity key (that in turn is certified by HSM manufacturer i.e. via MVK)
  - (Symmetric) keys may be identified via Hash(key data) if required

Important HSM Data Structures: ECU Configuration



- ECU configuration register (ECR)
  - Similar to Trusted Computing Platform Configuration Registers (PCR)
  - Enables trusted chain of measurements
  - Can be connected with references for enabling secure boot



Security Building Blocks: Overview

- Security Building Blocks (SBB) for
  - Encryption and decryption
  - Message authentication codes
  - Hashes and HMAC
  - Signature generation
  - Signature verification
  - Random numbers
  - Secure Counters
- Generic interface to use same SBBs with different concrete cryptographic algorithms (for capability, updates, ..)
- Session-based via `session_handle` and init(), update(), finish()

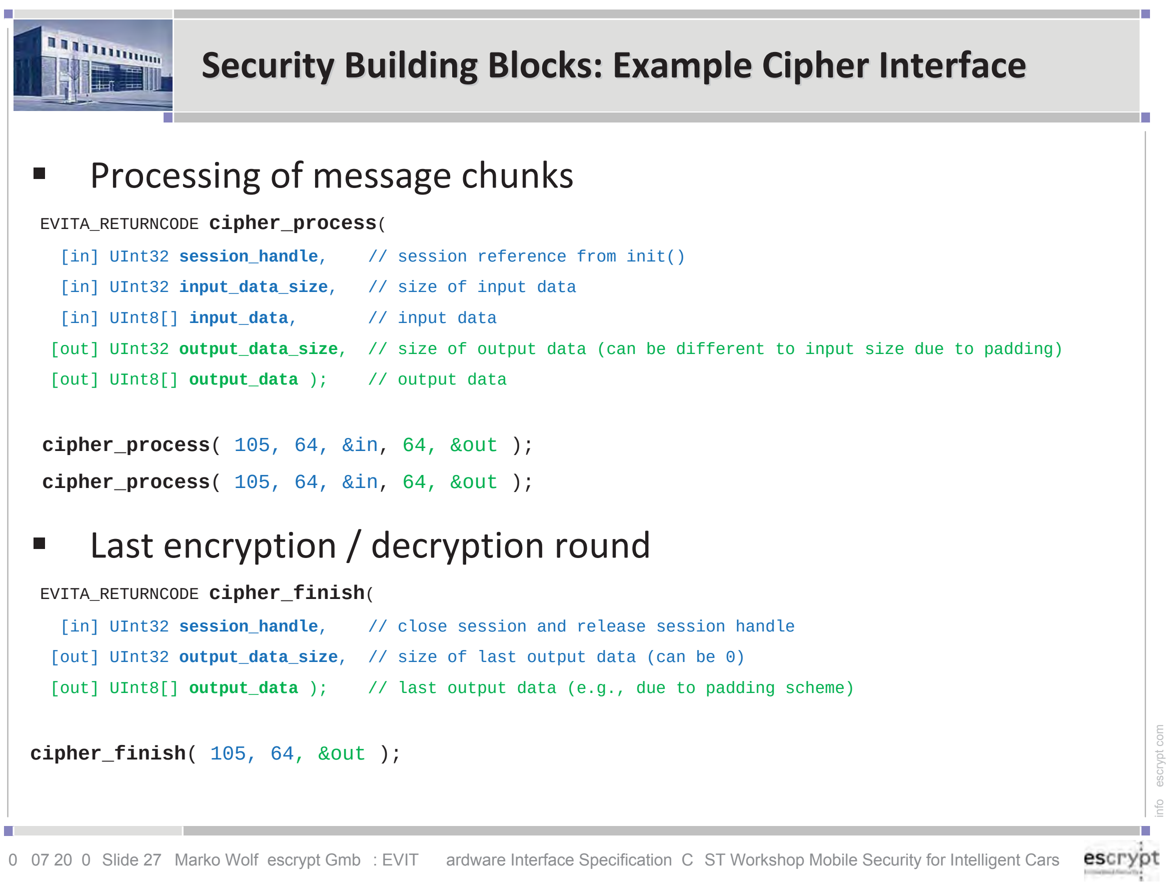
Security Building Blocks: Example Cipher Interface

- Initialization of hardware encryption/decryption session

```
EVITA_RETURNCODE cipher_init(
  [in] UInt8 algorithm_identifier,                // reference to associated symmetric algorithm
  [in] Enum cipher_mode{encrypt|decrypt},         // indicate decryption or encryption mode
  [in] Enum operation_mode{ECB|CBC|GCM|EAX|..},   // indicate cipher mode of operation
  [in] Enum padding_scheme{none|bit|byte|pkcsx|..}, // indicate padding scheme
  [in] UInt32 total_message_length,               // indicate message length (if req.by padding scheme)
  [in] UInt32 IV_size,                            // size of given initialization vector (can be 0)
  [in] UInt8[] IV,                                // set initialization vector (it's public)
  [in] UInt32 key_handle,                         // refer to internal key that will be used
  [in] UInt32 key_authorization_size,             // size of key usage authorization value (0 for none)
  [in] UInt8[] key_authorization_value,           // key usage authorization (i.e., password)
 [out] UInt32 max_chunk_size,                     // maximum size of a chunk on process()
 [out] UInt32 chunk_block_size,                   // chunk has to be a multiple of this block size
 [out] UInt32 session_handle );                   // enables interruption & parallel processing
```

- Invocation example

```
cipher_init( AES, encrypt, CBC, none, 128, 16, &IV, 11, 8, “password”, 64, 16, 105 );
```



Security Building Blocks: Example Cipher Interface

- Processing of message chunks

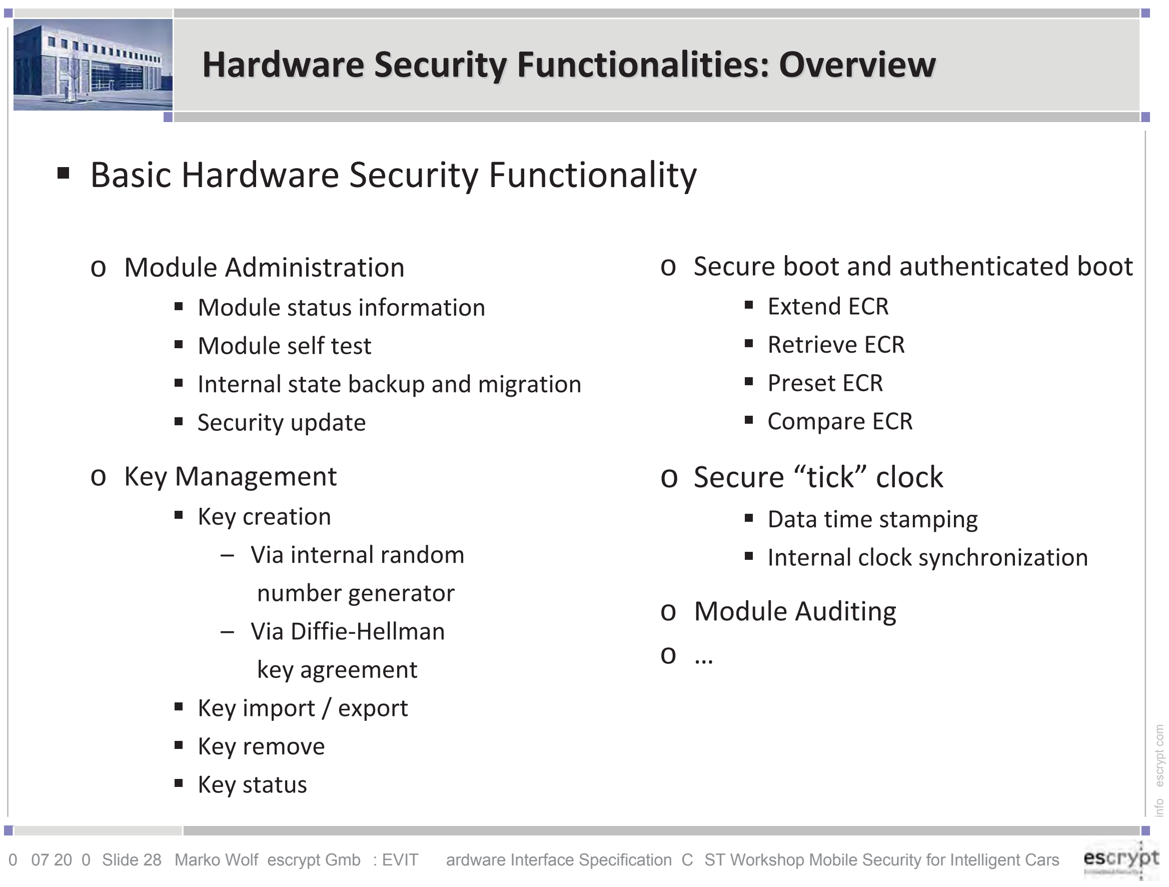
```
EVITA_RETURNCODE cipher_process(
   [in] UInt32 session_handle,    // session reference from init()
   [in] UInt32 input_data_size,   // size of input data
   [in] UInt8[] input_data,       // input data
  [out] UInt32 output_data_size,  // size of output data (can be different to input size due to padding)
  [out] UInt8[] output_data );    // output data

 cipher_process( 105, 64, &in, 64, &out );
 cipher_process( 105, 64, &in, 64, &out );
```

- Last encryption / decryption round

```
EVITA_RETURNCODE cipher_finish(
   [in] UInt32 session_handle,    // close session and release session handle
  [out] UInt32 output_data_size,  // size of last output data (can be 0)
  [out] UInt8[] output_data );    // last output data (e.g., due to padding scheme)

cipher_finish( 105, 64, &out );
```



Hardware Security Functionalities: Overview

- Basic Hardware Security Functionality
  - Module Administration
    - Module status information
    - Module self test
    - Internal state backup and migration
    - Security update
  - Key Management
    - Key creation
      - Via internal random number generator
      - Via Diffie-Hellman key agreement
    - Key import / export
    - Key remove
    - Key status
  - Secure boot and authenticated boot
    - Extend ECR
    - Retrieve ECR
    - Preset ECR
    - Compare ECR
  - Secure "tick" clock
    - Data time stamping
    - Internal clock synchronization
  - Module Auditing
  - …

Security Functionalities: Example Key Export

- Export of a key for transport, swapping or migration

```
EVITA_RETURNCODE key_export(
  [in] UInt32 key_handle,                      // reference to the internal key that becomes exported
  [in] Enum use_flags {encrypt|sign|..},       // define set of key use flags to become exported
  [in] UInt32 transport_key_handle,            // reference to the key used for transport encryption
    (use_flag = transport)
  [in] UInt32 transport_key_authorization_size,  // size of transport key usage authorization
  [in] UInt8[] transport_key_authorization,    // transport key usage authorization (i.e., password)
  [in] UInt32 authenticity_key_handle,         // reference to the key used for authenticity code
    creation (use_flag = sign)
  [in] UInt32 authenticity_key_authorization_size,// size of authenticity key usage authorization
  [in] UInt8[] authenticity_key_authorization,   // authenticity key usage authorization (e.g., PW)
 [out] UInt32 exported_key_size,               // returned (usually encrypted) export key blob size
 [out] UInt8[] exported_key,                   // returned (usually encrypted) export key blob
 [out] UInt32 key_authenticity_code_size,      // size of key authenticity code (signature or MAC)
    created by authenticity key
 [out] UInt8[] key_authenticity_code );        // key authenticity code (signature or MAC) to enable
    authenticity verification
```

- Invocation example

```
key_export( 11, encrypt, 12/TK, 0, NULL, 2/IDK, 8, “password”, 128, &blob, 128, &cert );
```

info escrypt com

escrypt

Security Functionalities: Example Key Import

- Import of a key after a transport, swapping or migration

```
EVITA_RETURNCODE key_import(
  [in] UInt32 transport_key_handle,            // reference to the internal key used for transport
    decryption (use_flag = transport)
  [in] UInt32 transport_key_authorization_size,  // size of transport key usage authorization
  [in] UInt8[] transport_key_authorization,    // transport key usage authorization (i.e., password)
  [in] UInt32 authenticity_key_handle,         // reference to the key used for authenticity code
    verification (use_flag = verify)
  [in] UInt32 authenticity_key_authorization_size,// size of authenticity key usage authorization
  [in] UInt8[] authenticity_key_authorization,   // authenticity key usage authorization (i.e., password)
  [in] Enum memory_target {nv|ram},            // import key into NV memory or RAM
  [in] UInt32 imported_key_size,               // given (usually encrypted) import key blob size
  [in] UInt8[] imported_key,                   // given (usually encrypted) import key blob
  [in] UInt32 key_authenticity_code_size,      // size of key authenticity code (signature or MAC)
  [in] UInt8[] key_authenticity_code,          // given key authenticity code (signature or MAC) to
    enforce and proof module internal protection
 [out] UInt32 key_handle );                    // reference to the (now) internal key that was imported
```

- Invocation example

```
key_import( 12/TK, 0, NULL, 13/IDK-EXT, 0, NULL, nv, 128, blob, 128, cert, 14 );
```

info escrypt com

escrypt

Conclusion: EVITA Hardware Security Architecture

- Provides a reliable security anchor for upper software layers through encapsulated generation, storage, and processing of security-critical material & provision of basic security functions
- Efficient, flexible and generic security interface
- Applies Trusted Computing ideas (e.g., authenticated boot) with meaningful extensions (e.g., use flags, individual authorizations)
- Accelerates security mechanisms by applying cryptographic accelerators (e.g., ECC, AES, WHIRLPOOL, RNG)
- Compatible with existing SHE specification for easy deployment
- Tamper-protection via on-chip integration (+ further measures)

info escrypt com

Dr.-Ing. Marko Wolf
Senior Security Engineer
marko.wolf@escrypt.com

escrypt Embedded Security

escrypt Gmb
Lise Meitner llee 4
4480 o chum

info escrypt com
phone: 49 0 234 43 870 209
fa : 49 0 234 43 870 2

Secure Software Architecture

Benjamin Weyl

BMW Group Research and Technology, Munich, Germany

Abstract

The EVITA [1] security architecture provides security services in order to fulfill the security requirements of today's and future applications. As the security requirements are successively increasing due to new application scenarios [1], the security architecture needs to be designed such, that it can be flexibly deployed for various sets of applications in very different on-board environments [3]. This is specifically motivated by partly monolithic integrated security solutions, where it is costly to adapt them according to the needs of new security requirements derived by new application scenarios or the ongoing development in IT- security solutions. With a monolithic design of security solutions, redundancy of functionality and complexity increases with the security requirements from different applications Therefore, a modular, scalable , configurable and adaptable security architecture for automotive on-board networks is proposed. This security architecture provides software security modules with dedicated abstract interfaces for accessing the security functionality. This security functionality can be flexibly integrated and applied within dedicated applications. Particular functionality can be defined by using a so-called plug-in mechanism that allows for the integration of various security mechanisms. The EVITA security services include, for example, encryption and decryption services , authentication, authorization and access control services, privacy services, secure communication and intrusion management services. The security architecture is complemented with the specification of EVITA hardware security modules in order to increase the security for certain applications. These HSMs in combination with separation technologies like virtualization, can serve as basis in order to provide a secure environment on multipurpose ECUs [4].

CV

Since graduation from Technical University of Munich (TUM) in 2003, BENJAMIN WEYL is engaged in research at BMW Research and Technology focusing on security for automotive on-board network, Car2Car and Car2Infrastructure scenarios. In 2007 he has received his Ph.D. from Darmstadt University of Technology. Mr. Weyl is chair of the Security and Privacy Working Group of the Car2Car-Communication Consortium and active within the EU FP7 IST project EVITA.

Contact

Dr. Benjamin Weyl
BMW Group Forschung und Technik
Hanauerstr. 46, 80993 München
Tel. +49 89 382-48951
eMail: benjamin.weyl@bmw.de

Literature

[1] EU FP7 Project EVITA. E-safety Vehicle Intrusion proTected Applications. www.evita-project.org.

[2] A. Ruddle, D. Ward, B. Weyl, S. Idrees, Y. Roudier, M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger, R. Rieke, M. Ritscher, H. Broberg, L. Apvrille, R. Pacalet, and G. Pedroza. Security requirements for automotive on-board networks based on dark-side scenarios. Technical Report Deliverable D2.3, EVITA Project, 2009.

[3] B. Weyl, M. Wolf, F. Zweers, T. Gendrullis, M. S. Idrees, Y. Roudier, H. Schweppe, H. Platzdasch, R. El Khayari, O. Henniger, D. Scheuermann, A. Fuchs, L. Apvrille, G. Pedroza, H. Seudié, J. Shokrollahi, and A. Keil. Secure on-board architecture specification. Technical Report Deliverable D3.2, EVITA Project, 2010.

[4] F. Stumpf, C. Meves, B. Weyl, M. Wolf. A Security Architecture for Multipurpose ECUs in Vehicles. In Proceedings of 25. VDI/VW-Gemeinschaftstagung: Automotive Security, Ingolstadt, Germany, October 19 - 20, 2009.

EVITA – Secure Software Architecture.

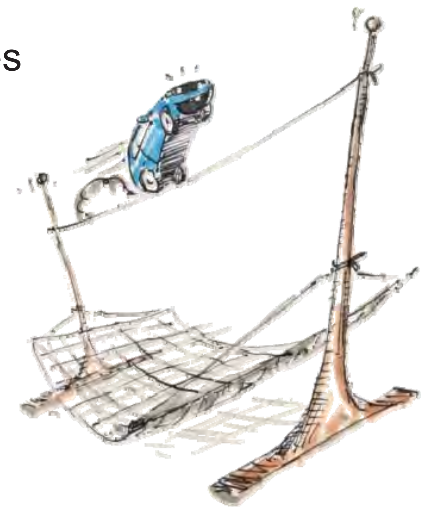
Dr.-Ing. Benjamin Weyl
BMW Group Research and Technology

Email: benjamin.weyl@bmw.de

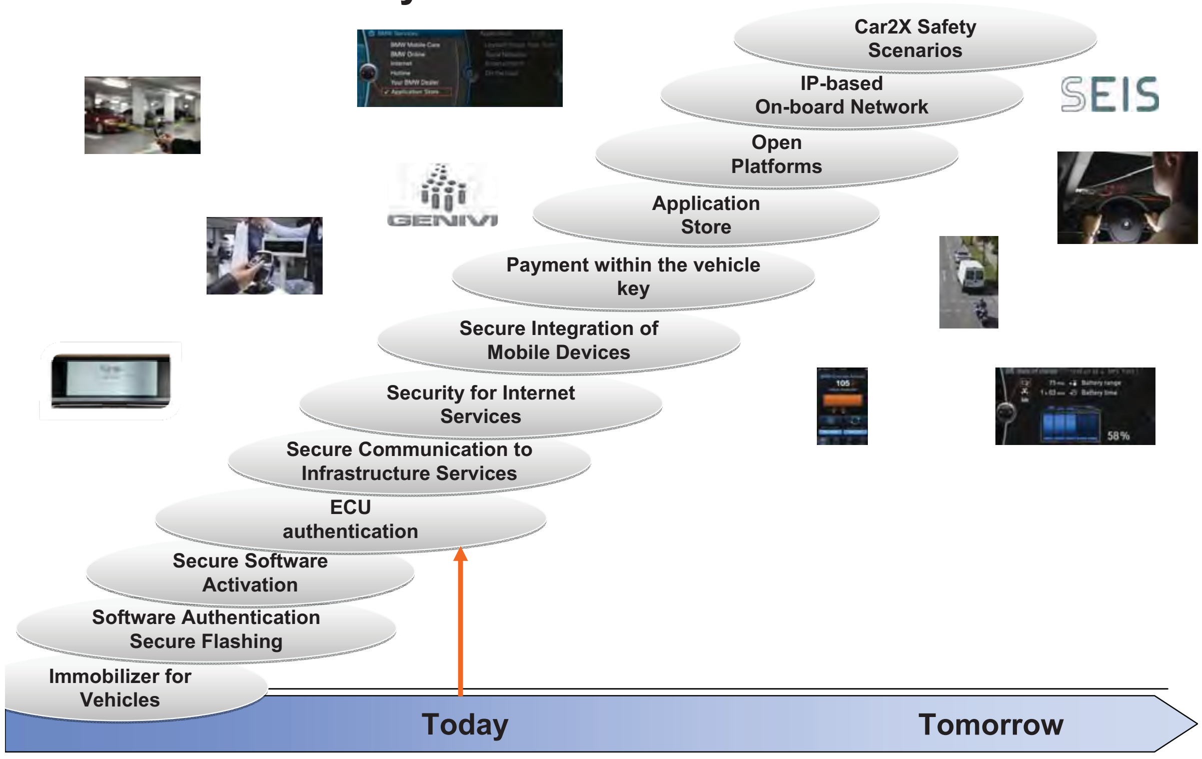
CAST Workshop „Mobile Security for Intelligent Cars“
Darmstadt, 01.07.2010

Outline.

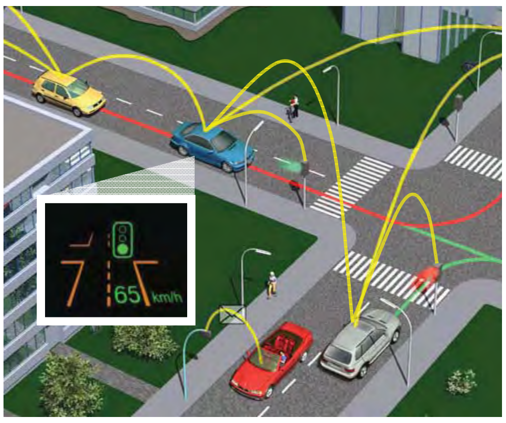
1. Automotive Security Use Cases
2. EVITA Project Overview
3. Secure Software Arcitecture
4. Summary



Automotive Security Use Cases.


Car2X Safety Scenarios
IP-based On-board Network
SEIS
Open Platforms
Application Store
GENIVI
Payment within the vehicle key
Secure Integration of Mobile Devices
Security for Internet Services
Secure Communication to Infrastructure Services
ECU authentication
Secure Software Activation
Software Authentication Secure Flashing
Immobilizer for Vehicles
Today
Tomorrow

Automotive Security Use Cases: Car2X Safety Scenarios.

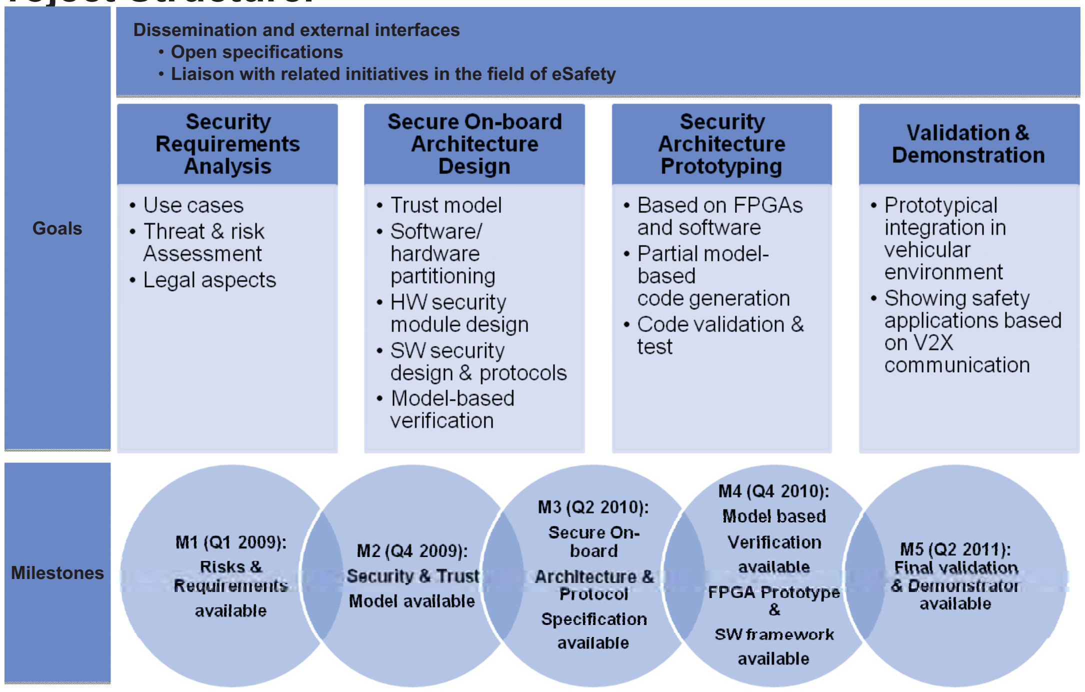


Project Objectives.

www.evita-project.org

- Modular, (cost-) efficient security for:
  - In-vehicular devices: sensors, actuators, ECUs with
  - HW and SW architecture securing SW applications based on the HW modules
- in order to:
  - enforce ECU software protection against SW attacks
  - plus optional selected HW attacks depending on the level of HW tamper protection
  - provide ECU HW/SW-configuration attestation (reliable proof of setup)
  - support/process ECU to ECU communication protection
  - support/process V2X communication and privacy protection
- based on:
  - hardware based security anchors
  - software security layer, mechanisms and API specification
  - that make use of HW security module BUT can also be built completely in SW

Project Structure.



Goals

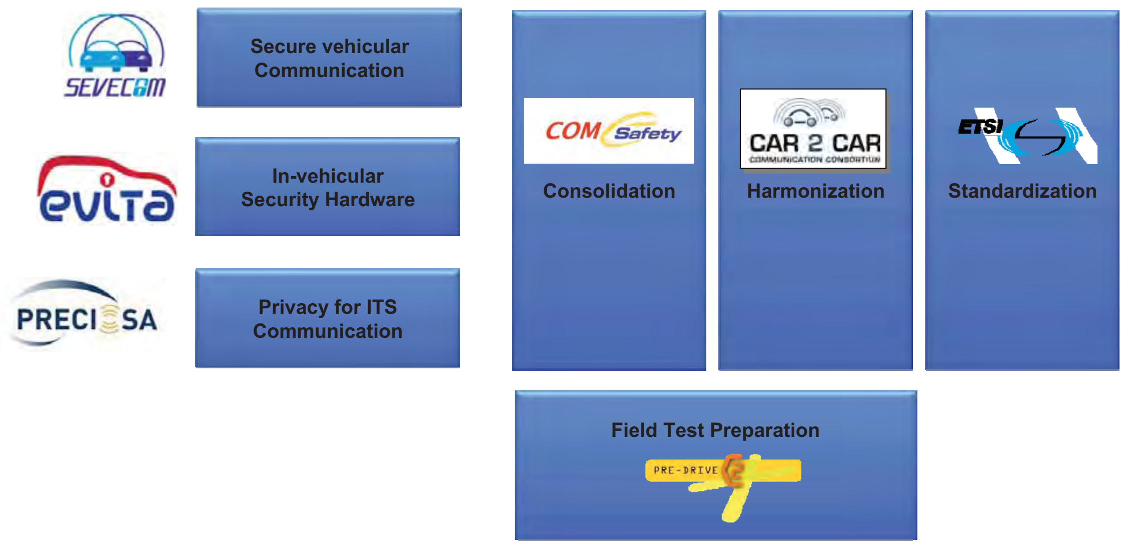
Dissemination and external interfaces
- Open specifications
- Liaison with related initiatives in the field of eSafety

Security Requirements Analysis	Secure On-board Architecture Design	Security Architecture Prototyping	Validation & Demonstration
• Use cases • Threat & risk Assessment • Legal aspects	• Trust model • Software/ hardware partitioning • HW security module design • SW security design & protocols • Model-based verification	• Based on FPGAs and software • Partial model-based code generation • Code validation & test	• Prototypical integration in vehicular environment • Showing safety applications based on V2X communication

Milestones

- M1 (Q1 2009): Risks & Requirements available
- M2 (Q4 2009): Security & Trust Model available
- M3 (Q2 2010): Secure On-board Architecture & Protocol Specification available
- M4 (Q4 2010): Model based Verification available FPGA Prototype & SW framework available
- M5 (Q2 2011): Final validation & Demonstrator available

Project Scope: Complementary Security Activities.

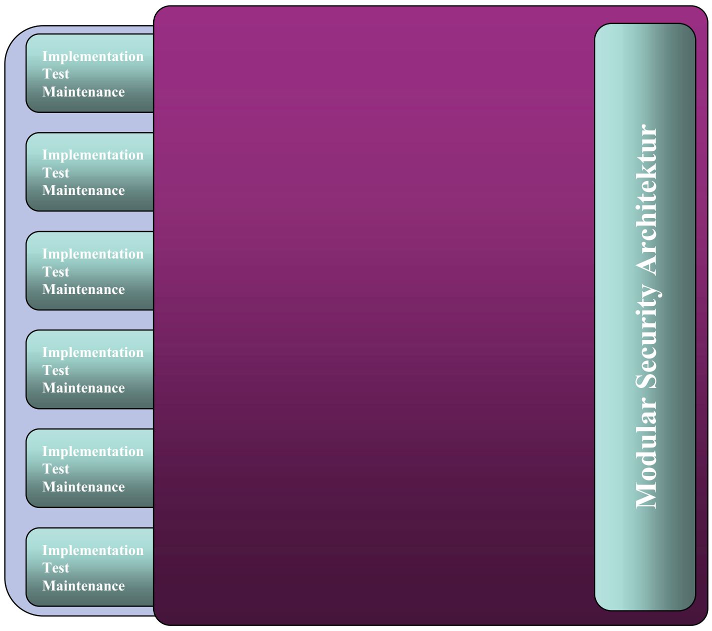

SEVECOM
Secure vehicular Communication
evita
In-vehicular Security Hardware
PRECIOSA
Privacy for ITS Communication
COM Safety
Consolidation
CAR 2 CAR COMMUNICATION CONSORTIUM
Harmonization
ETSI
Standardization
Field Test Preparation
PRE-DRIVE C2X

Project Scope: Focus on in-vehicular systems.

- The attacks on *external* communication:
  - must be prevented or
  - at least be detected and contained,
  - so that fake messages injected into the (wireless) communication infrastructure are properly identified and eliminated before influencing eSafety applications.

- Attacks on *in-vehicular* system infrastructure
    - via physical access or
    - via wireless interface
  - must be prevented or
  - at least be detected and contained,
  - so that fake messages are properly identified and eliminated before influencing applications.

Embedded Vehicular Security Architecture.



Focus on Security Requirements within a Sustainable and Scalable Security Architecture for Vehicular On-board Architectures

Embedded Vehicular Security: Software Architecture

1. Scalability:
   - Flexible configuration
   - Deployment of security funtionality according to use cases
   - Possible adaptation according to new use cases

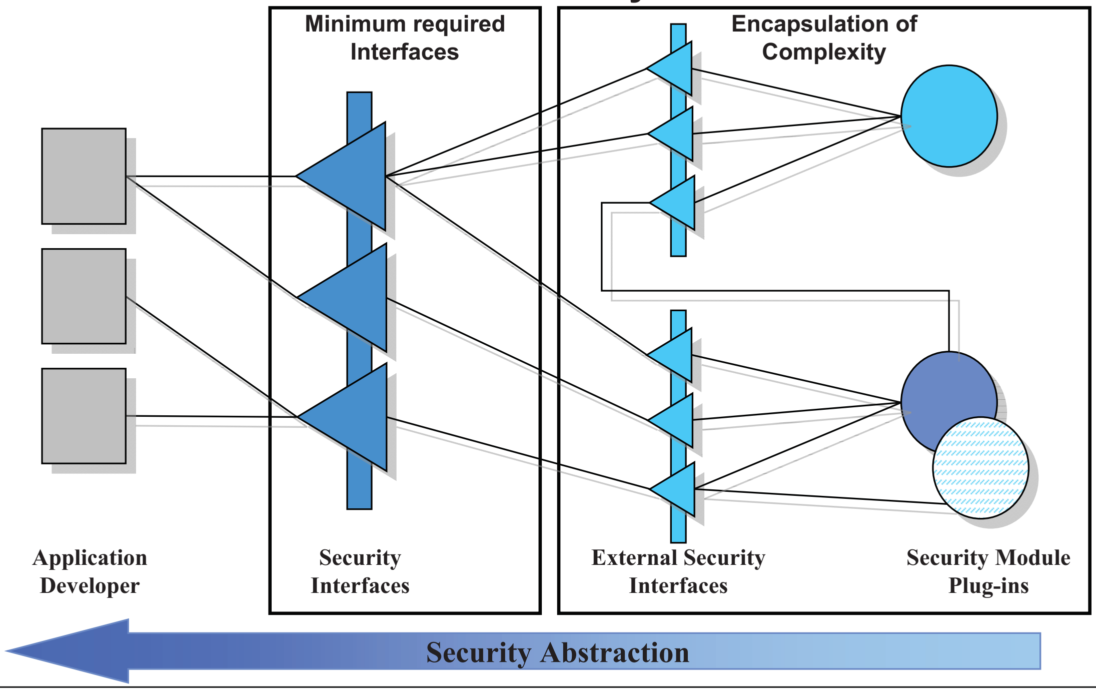
2. Encapsulation and abstraction:
   - Overall on-board security architecture
   - Easier integration into application
   - Centralizded maintenance of dedicated security modules

➡ **Modular and flexible security architecture**

Secure Software Architecture.

- Key capabilities:
  - **Flexible integration** of new security mechanisms and protocols into overall security architecture
  - **Flexible deployment** within the on-board network, e.g. centered or multi-centered approach, depending on the system environment and applications
  - **Static and dynamic Configuration** of security mechanisms, policies and credentials
  - **Secure update** mechanisms
  - Security **API for application developers**

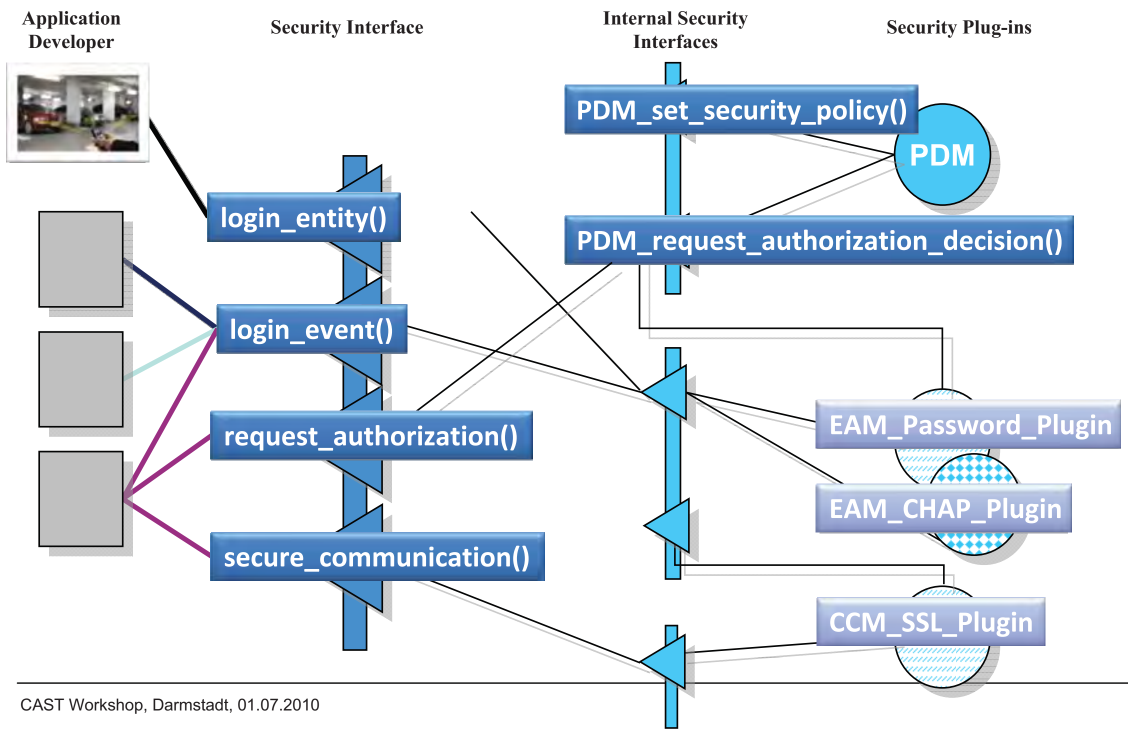
Modularization of Vehicular Security Architecture.



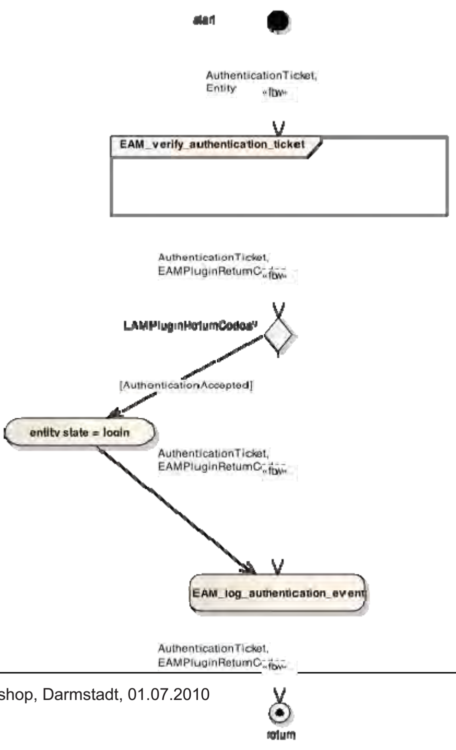
Embedded Vehicular Security Architecture: Modules.

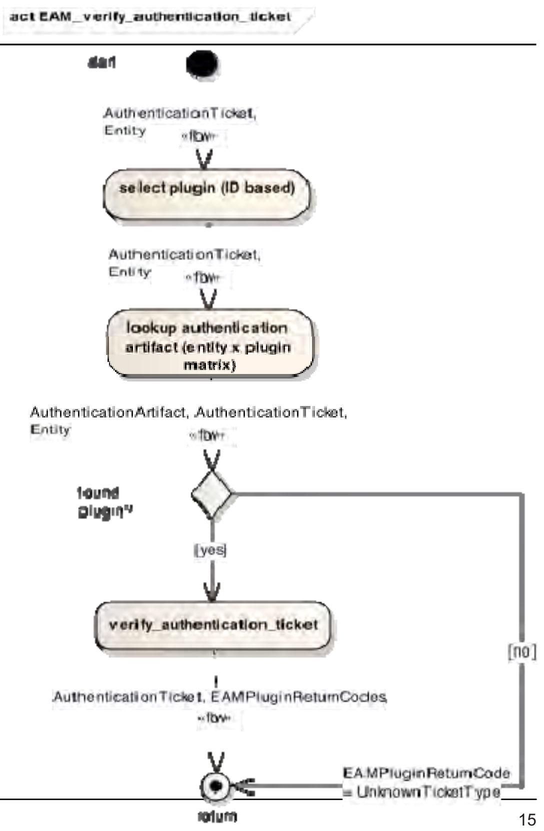
PDM Policy Decision Module	Managment of Security Policies e.g. for Authorization Decisions and Access Control
SWD Security Watchdog Module	Intrusion Management, Single Sign On
CCM Communication Control Module	Authentic and Confidential End2End Communication
EAM Entity Authentication Module	Authentication of Users and Applications Authentication of ECUs Privacy Mechanisms, e.g Identity Concealment
PIM Platform Integrity Module	Interfaces for Hardware Security Modules
SSM Secure Storage Module	Confidential Storage of Date and Personal Information

Secure Software Architecture: Example.

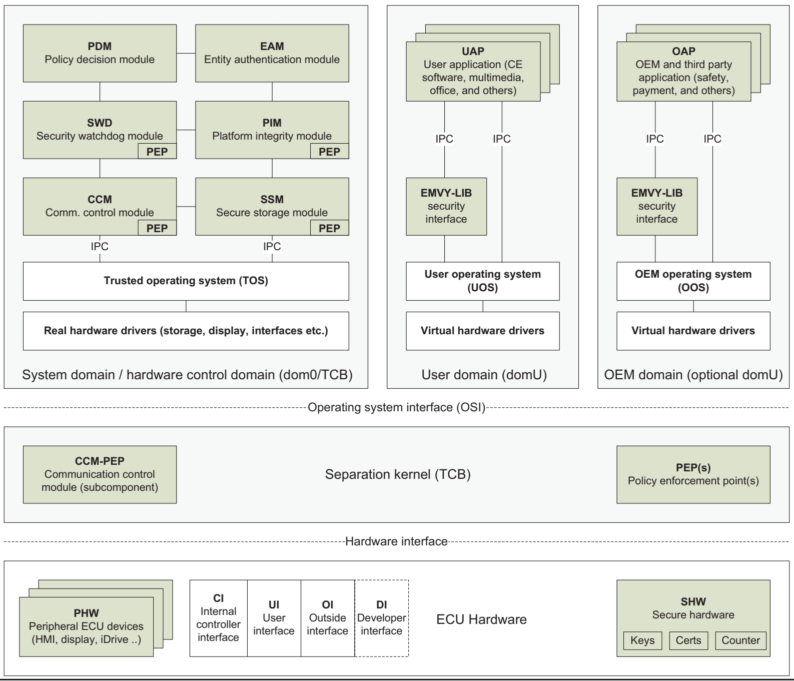


Example: EAM Module.


start
AuthenticationTicket, Entity «flow»
EAM_verify_authentication_ticket
AuthenticationTicket, EAMPluginReturnCodes «flow»
EAMPluginReturnCodes?
[AuthenticationAccepted]
entity state = login
AuthenticationTicket, EAMPluginReturnCodes «flow»
EAM_log_authentication_event
AuthenticationTicket, EAMPluginReturnCodes «flow»
return


act EAM_verify_authentication_ticket
start
AuthenticationTicket, Entity «flow»
select plugin (ID based)
AuthenticationTicket, Entity «flow»
lookup authentication artifact (entity x plugin matrix)
AuthenticationArtifact, AuthenticationTicket, Entity «flow»
found plugin?
[yes]
verify_authentication_ticket
[no]
AuthenticationTicket, EAMPluginReturnCodes «flow»
EAMPluginReturnCode = UnknownTicketType
return

Deployment Scenario: Multipurpose ECU.


PDM
Policy decision module
EAM
Entity authentication module
SWD
Security watchdog module
PEP
PIM
Platform integrity module
PEP
CCM
Comm. control module
PEP
SSM
Secure storage module
PEP
IPC
IPC
Trusted operating system (TOS)
Real hardware drivers (storage, display, interfaces etc.)
System domain / hardware control domain (dom0/TCB)
UAP
User application (CE software, multimedia, office, and others)
IPC
IPC
EMVY-LIB
security interface
User operating system (UOS)
Virtual hardware drivers
User domain (domU)
OAP
OEM and third party application (safety, payment, and others)
IPC
IPC
EMVY-LIB
security interface
OEM operating system (OOS)
Virtual hardware drivers
OEM domain (optional domU)
Operating system interface (OSI)
CCM-PEP
Communication control module (subcomponent)
Separation kernel (TCB)
PEP(s)
Policy enforcement point(s)
Hardware interface
PHW
Peripheral ECU devices (HMI, display, iDrive ..)
CI
Internal controller interface
UI
User interface
OI
Outside interface
DI
Developer interface
ECU Hardware
SHW
Secure hardware
Keys
Certs
Counter

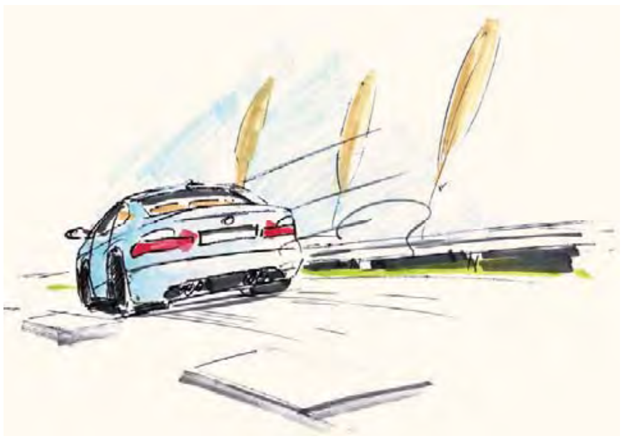
Summary.

– Modular and scalable Software Security Architecture:
  - On-board security architecture
  - Modularization and abstraction of interfaces
  - Plug-in architecture in order to integrate dedicated security mechanisms/protocols

– Advantages:
  - Overall on-board security architecture
  - Easy-to-use application developer API of the security services
  - Flexible deployment and configuration:
    - according to security requirements and
    - according to the design of the on-board architecture
  - Flexible security updates

Thank you for your attention.

www.evita-project.org



Benjamin Weyl
Chair WG Security & Privacy

CAR 2 CAR COMMUNICATION CONSORTIUM

www.car-2-car.org

benjamin.weyl@bmw.de

BMW Group
Research and Technology

Secure On-Board Protocols

Hendrik Schweppe

EURECOM, Sophia-Antipolis, France

Abstract

Vehicles have traditionally been a mechanical domain. In recent decades, this changed drastically: starting with electronic engine management in the 70s, vehicles have evolved to a multi-connected and computerized platform; simultaneously, safety systems that not only rely on mechanics but also on electronic systems (electronic stability, anti-lock brakes) have been introduced with great success. The more recent introduction of Car-to -Infrastructure technologies and that of Car-to-Car systems in the near future constitute the next step that will turn vehicles into communicating artifacts.

This situation is likely to generate new security threats with respect to communications between vehicles (VANETs), as well as within on-board embedded systems. Successful attacks on poorly designed communication protocols have recently been demonstrated for both external and internal protocols. This talk will focus on the latter.

The paradigms of on-board network architectures and communication will first be reviewed. After a description of attacks, the approach taken in the EVITA research project will be introduced. A particular focus will be on the cryptographic protocols currently being designed. Using these protocols, a chain of trust can be built, reaching from sensors to external entities. Mechanisms such as key exchange as well as integration issues for security payload are discussed and an outlook on future work is given.

CV

HENDRIK SCHWEPPE received the Dipl.-Ing. degree from Technische Universität Berlin , Germany, in 2008. His diploma thesis was performed during a research stay at Mercedes-Benz RDNA, Palo Alto, CA, where he designed and prototyped a new in-vehicle stream processing system. He is currently working towards the Ph.D. degree at EURECOM, Sophia-Antipolis, France.

He joined the research institute EURECOM, Sophia-Antipolis, France, in June 2009. He is member of the networking and security group, where he works on *in-vehicle security systems*, with a focus on *on-board communication*. He is involved in the EVITA EU project. His research interests include distributed systems, automotive and embedded systems as well as security.

Mr. Schweppe is a member of Gesellschaft für In formatik. Besides his activities in EVITA, he is also active in the security working g roup of the Car to Car Communication Consortium as well as ETSI ITS Security working group.

Contact

Hendrik Schweppe
EURECOM
F-06560 Sophia-Antipolis cedex
Tel. +33 4 93 00 81 00
Fax. +33 4 93 00 82 00
eMail: hendrik.schweppe@eurecom.fr

Literature

[1] B. Weyl, M. Wolf, F. Zweers, T. Gendrullis, M. S. Idrees, Y. Roudier, H. Schweppe, H. Platzdasch, R. El Khayari, O. Henniger, D. Scheuermann, A. Fuchs, L. Apvrille, G. Pedroza, H. Seudié, J. Shokrollahi, and A. Keil. Secure on-board architecture specification. Technical Report Deliverable D3.2, EVITA Project, 2010.

[2] A. Ruddle, D. Ward, B. Weyl, S. Idrees, Y. Roudier, M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger, R. Rieke, M. Ritscher, H. Broberg, L. Apvrille, R. Pacalet, and G. Pedroza. Security requirements for automotive on-board networks based on dark-side scenarios. Technical Report Deliverable D2.3, EVITA Project, 2009.

[3] T. Hoppe, S. Kiltz, and J. Dittmann. Security threats to automo- tive can networks — practical examples and selected short-term countermea- sures. In *SAFECOMP '08: Proceedings of the 27th international conference on Computer Safety, Reliability, and Security*, pages 235–248, Berlin, Heidelberg, 2008. Springer-Verlag.

[4] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. In *31st IEEE Symposium on Security and Privacy*, volume 31, 2010.

EURECOM
Sophia Antipolis

Secure On-Board Protocols

Hendrik Schweppe
EURECOM, France

2nd CAST Workshop on Mobile Security for Intelligent Cars, July 1, 2010

Outline

- Vehicular Communication: Architecture and Paradigm
  - Domain Background
  - On-Board Communication Architecture
  - Attacks on In-Car Communication
- Security in On-Board Networks
  - Application-Based Requirements for Security
  - Distinctive Constraints and Features
- Mechanisms
  - Authentication and Key Management
  - Synchronization and Updates
  - Coping with Embedded Constraints
- Outlook and Integration

Wiring a few years ago . . .

copyrighted

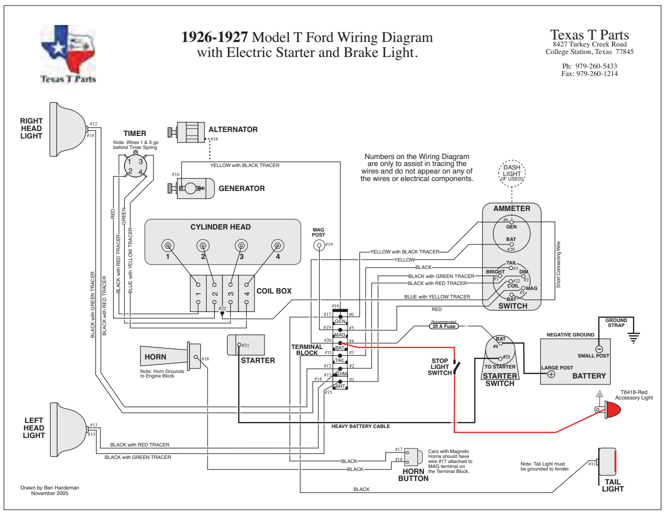
Wiring itself was limited to a few electric components:

- Lights
- Ignition
- Starter

- Introduced by Ford in 1915 (electrical lighting for Model T)

- The VW Käfer still only used an A4 page for complete wiring (even in 1970).

- In the late 70s, electronics came up to enhance efficiency (rudimentary engine management). Bosch's Jetronic started this.



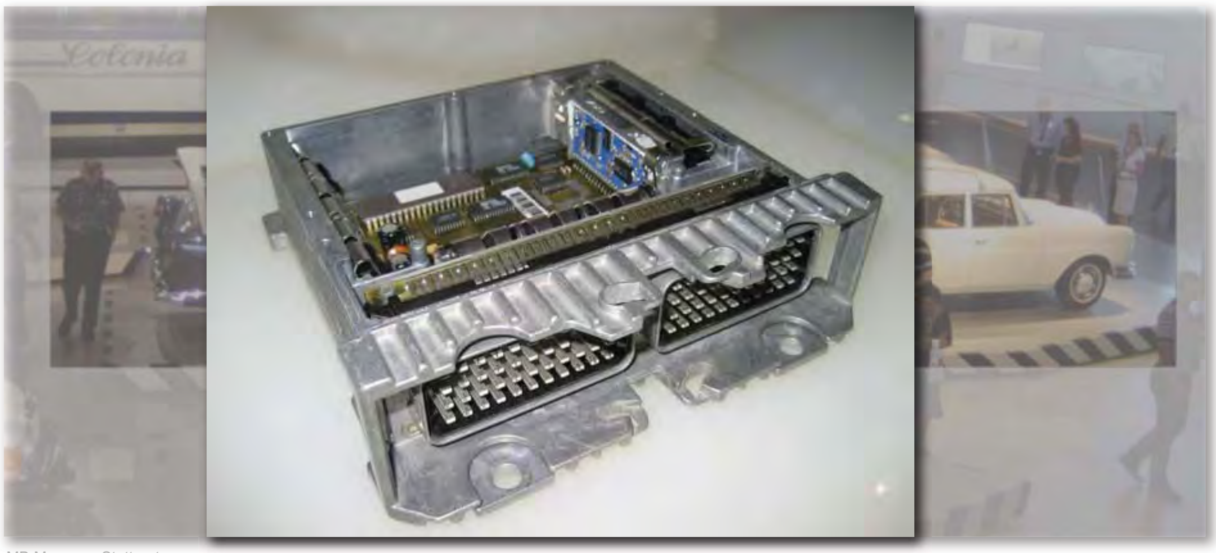
copyrighted

From Wiring to Electronics

MB Museum, Stuttgart

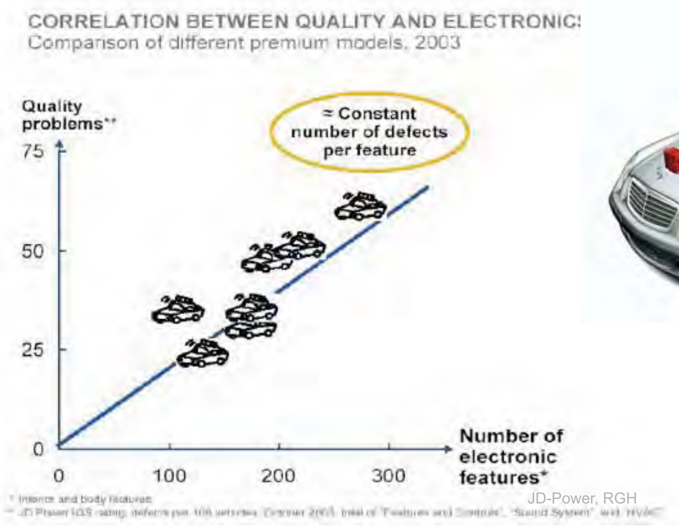
Wired transfer of sensor data to a following station wagon, equipped with oscilloscopes, plotters and a chair for the operator.

From Wiring to Electronics


MB Museum, Stuttgart

Wired transfer of sensor data to a following station wagon,
equipped with oscilloscopes, plotters and a chair for the operator.

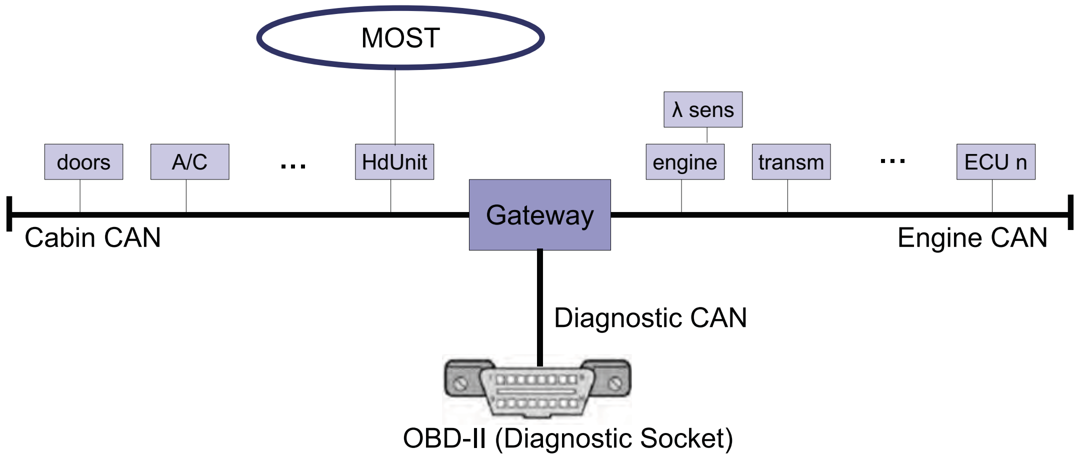
Features... lead to bugs!



JD-Power, RGH

DC-Media

Abstract Network Architecture



On-Board Networks

Data sent periodically between ECUs, sensors and actors

Paradigm:
- signal based
- service oriented

Functional requirements:
- low latency
- robust

- no security -

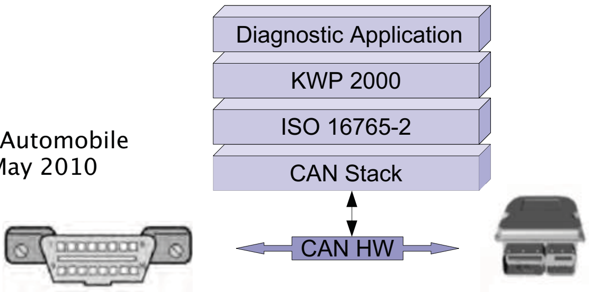
Physical Attacks

- Physical access necessary
  - Difficult, but not impossible. Exposed places are e.g. electric mirrors or tire pressure sensors [Hoppe, Kiltz, et al.]
- Cheap microcontrollers (Atmel) with CAN interfaces available
  - Easily create trigger condition:
    (if speed > 150): jam the bus or send some fake data to open windows

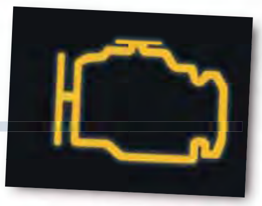
Atmel

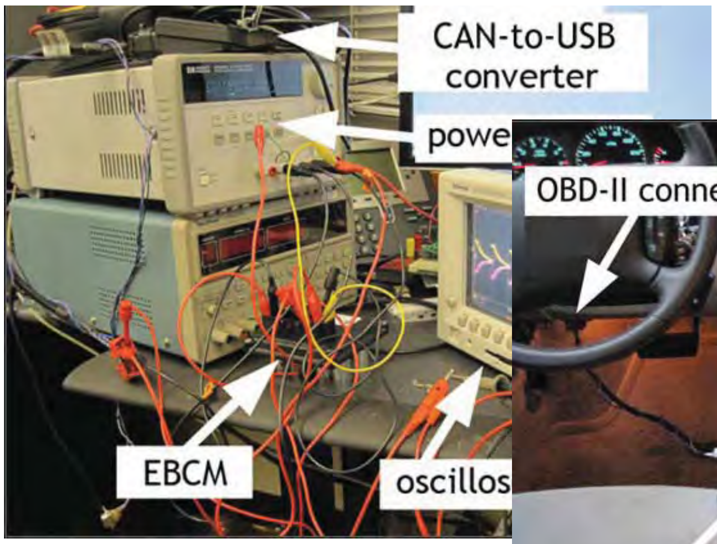
- Special “diagnosis” bus is openly accessible
  - OBD is short for the “On-Board Diagnosis” socket

    present in all new vehicles
    US: since 1996, Europe shortly after
- Research Paper Experimental Security Analysis of a Modern Automobile appeared in 31st IEEE Symposium on Security and Privacy, May 2010

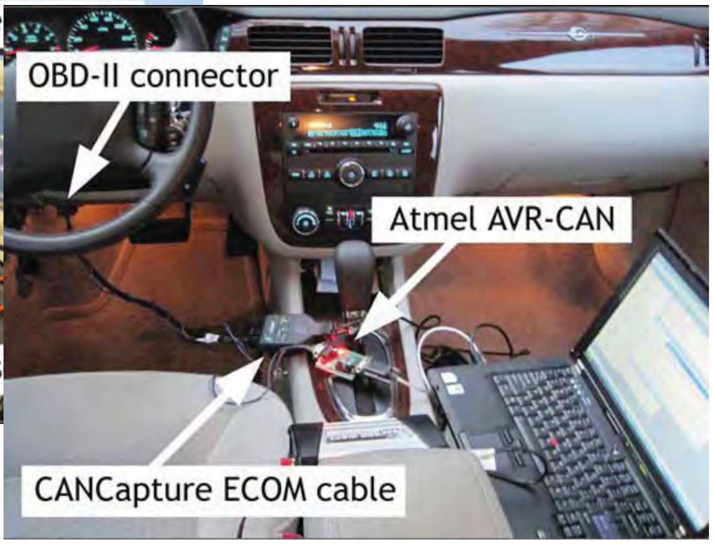


Experimental Analysis Paper (i)



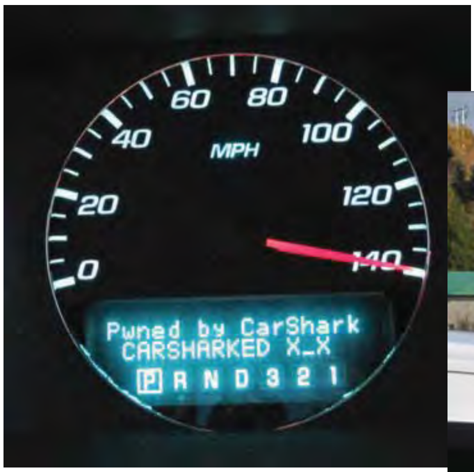


CAESS



CAESS

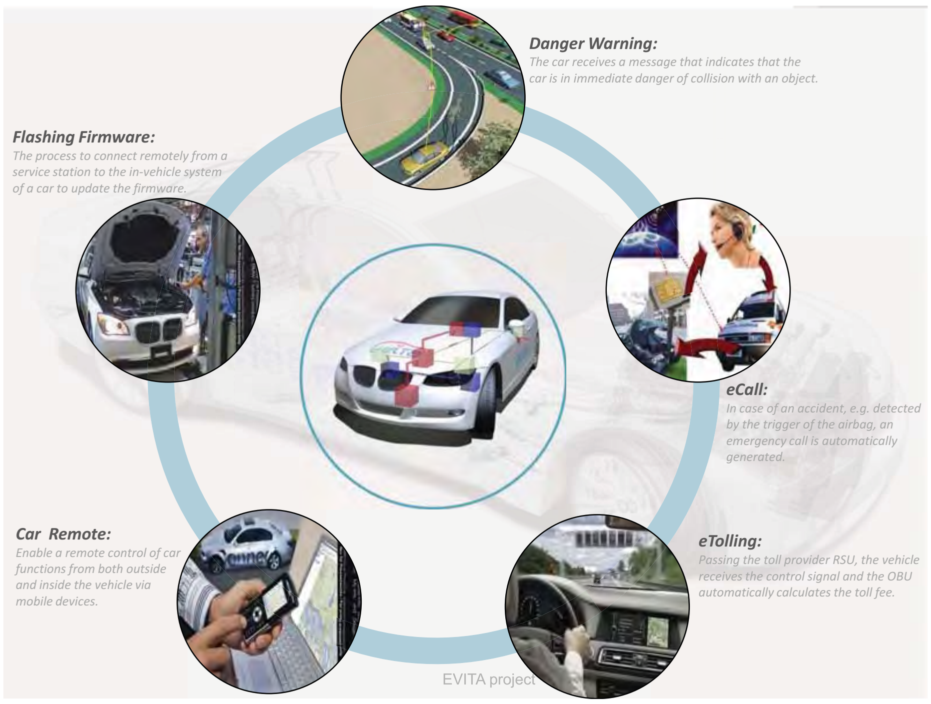
Experimental Analysis Paper (i)



CAESS

CAESS

Vehicle e-Services



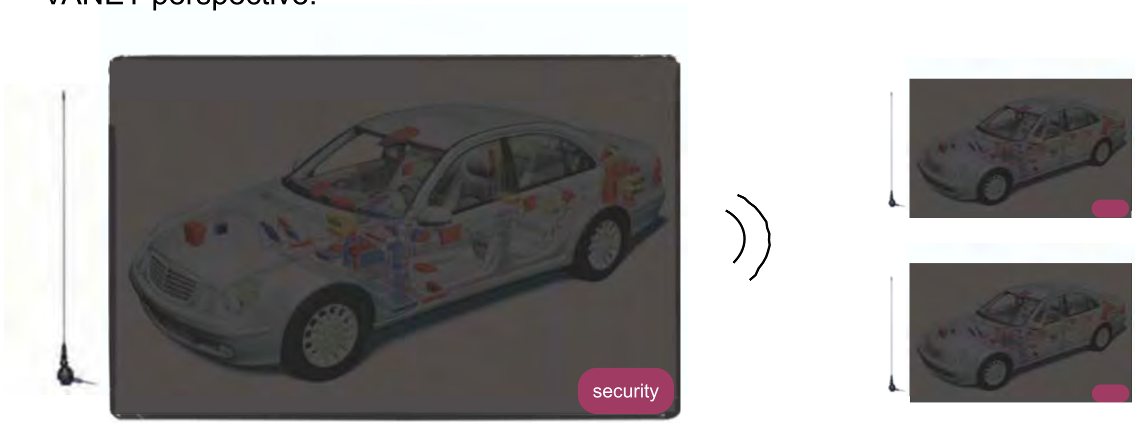
Danger Warning:
The car receives a message that indicates that the car is in immediate danger of collision with an object.
Flashing Firmware:
The process to connect remotely from a service station to the in-vehicle system of a car to update the firmware.
eCall:
In case of an accident, e.g. detected by the trigger of the airbag, an emergency call is automatically generated.
Car Remote:
Enable a remote control of car functions from both outside and inside the vehicle via mobile devices.
eTolling:
Passing the toll provider RSU, the vehicle receives the control signal and the OBU automatically calculates the toll fee.

EVITA project

External Communication

- The car as a black box

VANET perspective:

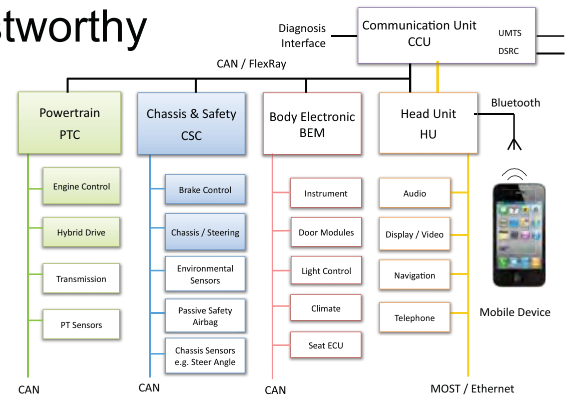


Trust in data?

- **Trust defined as:**
  "firm belief in the reliability, truth, ability, or strength of someone or something" [Oxford American Dictionary, 2010]

  - Security Applications
    - Take action that depends on incoming data
    - Need to know that data is trustworthy

- **Questions:**
  - Origin of data?
  - How to assure trust?



New Applications

… new security requirements

- Virtualized Approached
- Shielded Execution Environments
- Open Environments for third party devices and applications
- Access Control

Continental

OVERSEE

Multimedia Interfaces

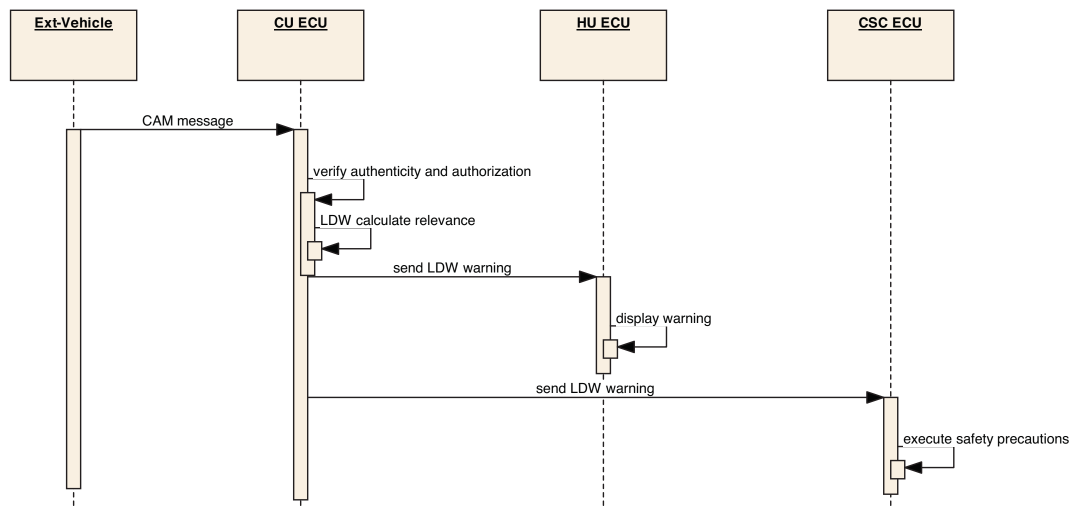
- Open Interfaces
  - Multimedia.
  - Users bring their own devices

BMW Press

Plug and enjoy.
The BMW interface for the Apple iPhone.

BMW online

Local Danger Warning

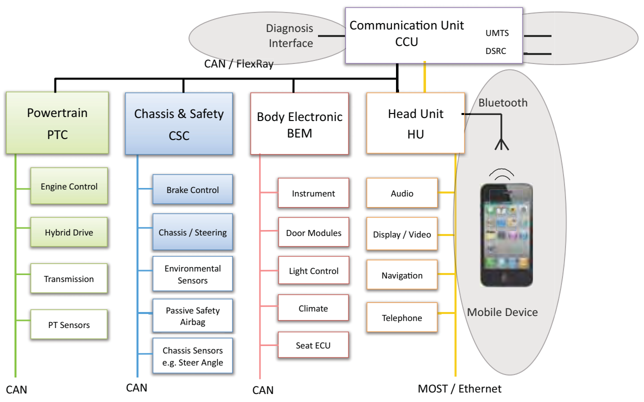


trust the signature (“verify authenticity and authorization”)
assumes that security process has taken place at other vehicle (not only origin!)

EVITA

EVITA: E-safety Vehicle Intrusion Protected Applications

- **Holistic approach: chain of trust from sensor to remote vehicle**
- Focus on preventing network attacks:
  - Communication centric (cryptographic protocols)
  - Dynamic Access Control
- Hardware protection: key storage and platform integrity

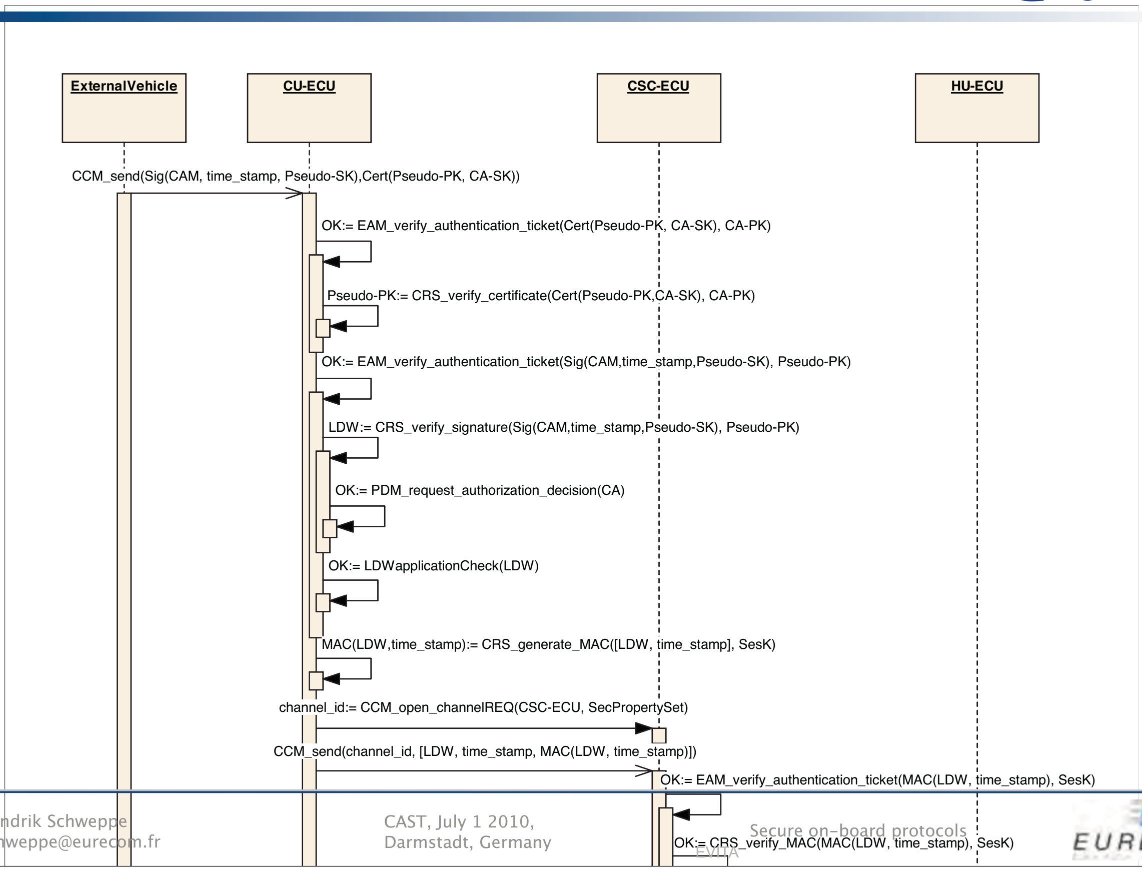


EVITA

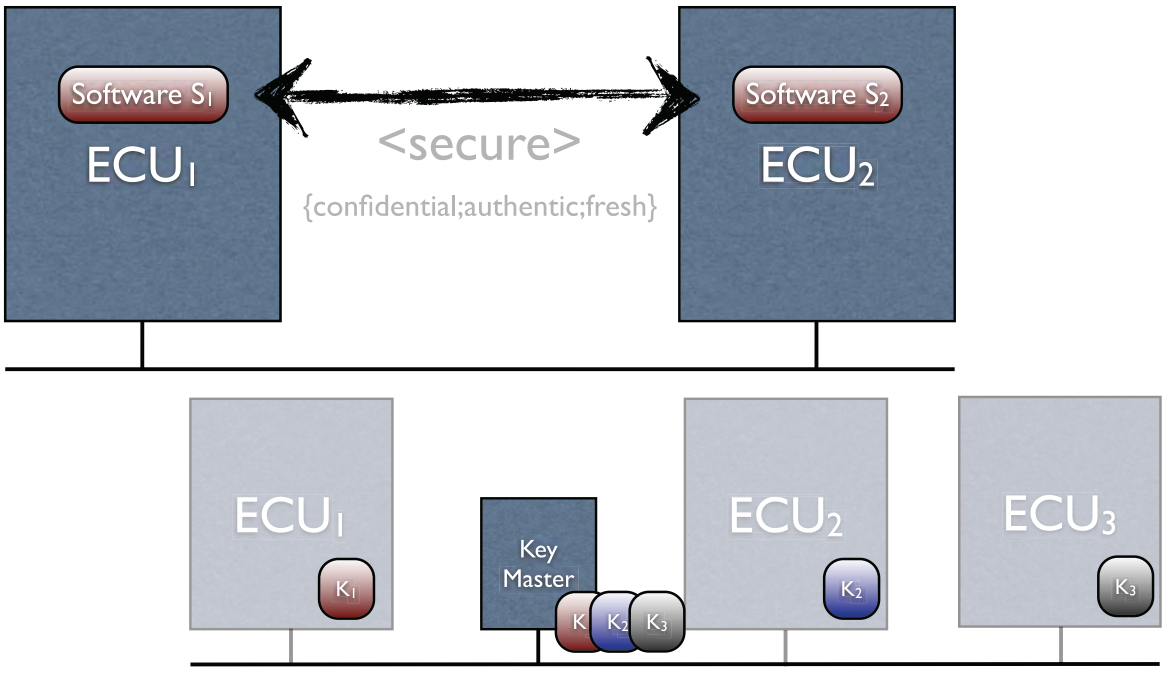
On-Board Protocols

- Principles:
  - Establish trust for applications that rely on external data
  - Based on cryptographic material
    - protected from attacks
    - attested by external trusted party
  - Based on integrity of the whole vehicle platform
- Design Goals:
  - Efficient - small security overhead
  - Scalable - number of ECUs
  - Network agnostic - usable with CAN, FlexRay, Ethernet,...
  - Portable - applicable to different RTEs
- Approach:
  - Service oriented and layered protocol design
  - Simulation-based overhead estimations
  - Combination of asymmetric (VANET) and symmetric cryptography (on-board)

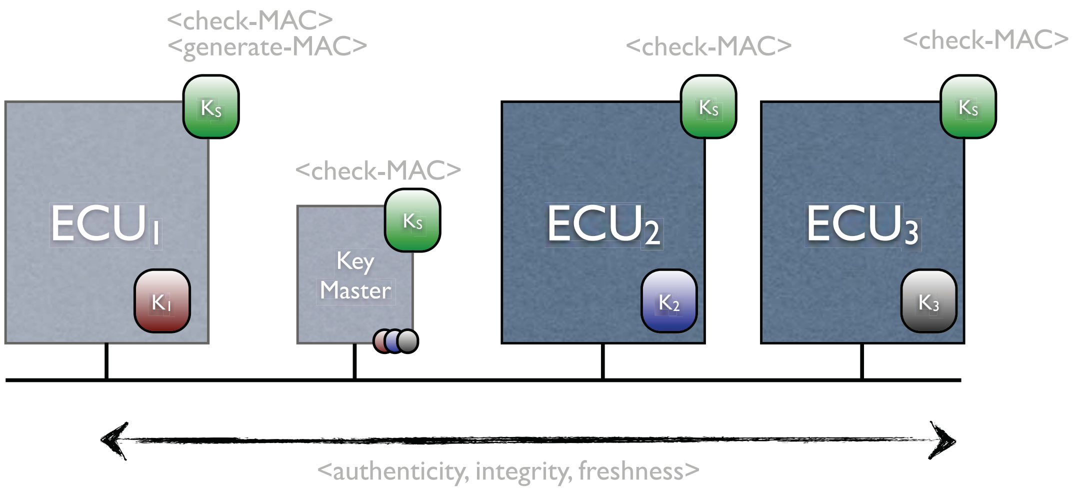
EVITA


ExternalVehicle
CU-ECU
CSC-ECU
HU-ECU
CCM_send(Sig(CAM, time_stamp, Pseudo-SK),Cert(Pseudo-PK, CA-SK))
OK:= EAM_verify_authentication_ticket(Cert(Pseudo-PK, CA-SK), CA-PK)
Pseudo-PK:= CRS_verify_certificate(Cert(Pseudo-PK,CA-SK), CA-PK)
OK:= EAM_verify_authentication_ticket(Sig(CAM,time_stamp,Pseudo-SK), Pseudo-PK)
LDW:= CRS_verify_signature(Sig(CAM,time_stamp,Pseudo-SK), Pseudo-PK)
OK:= PDM_request_authorization_decision(CA)
OK:= LDWapplicationCheck(LDW)
MAC(LDW,time_stamp):= CRS_generate_MAC([LDW, time_stamp], SesK)
channel_id:= CCM_open_channelREQ(CSC-ECU, SecPropertySet)
CCM_send(channel_id, [LDW, time_stamp, MAC(LDW, time_stamp)])
OK:= EAM_verify_authentication_ticket(MAC(LDW, time_stamp), SesK)
OK:= CRS_verify_MAC(MAC(LDW, time_stamp), SesK)

On-Board communication

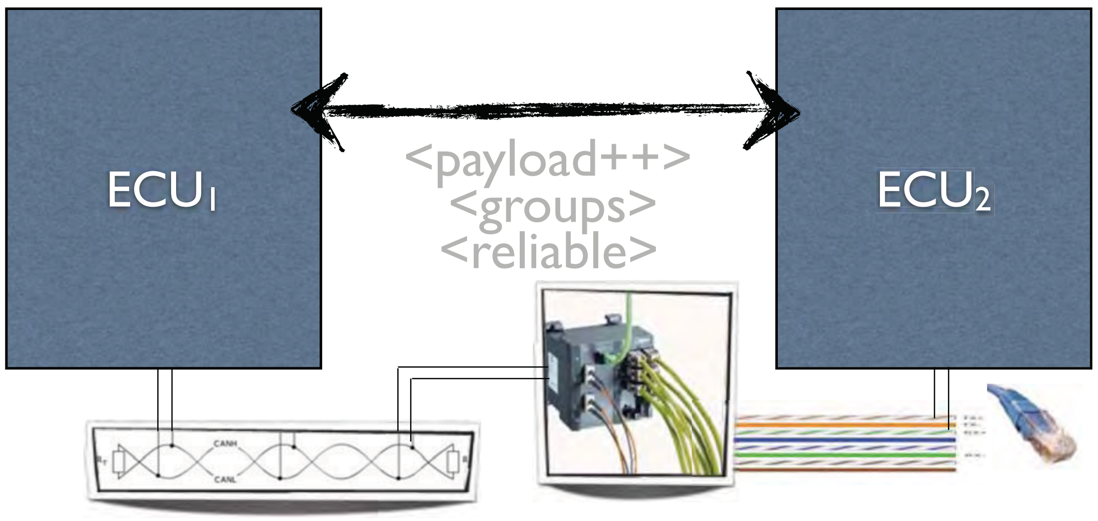


Secure Communication: one to many



- Basic usage control at ECU/HSM
- Comprehensive access control at KeyMaster

Data transport and addressing



Enable communication and routing on different buses:
Use of "the common transport protocol"

Secure Sessions

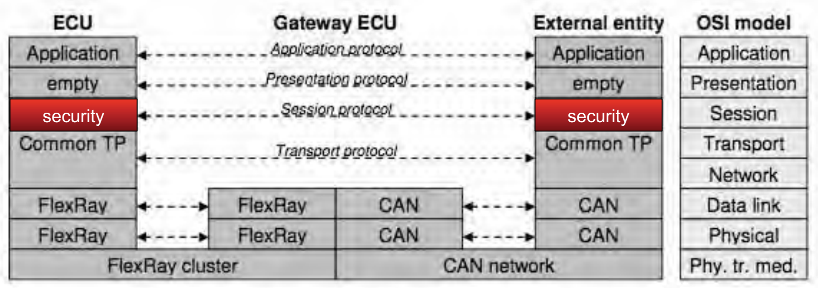


Figure 5-7. In-vehicle Data Exchange with Common Transport Protocol

The Common Transport Protocol CTP provides
- Sender & Destination addressing
- Large payload

EVITA adds
- Security Payload
- One-to-many communication

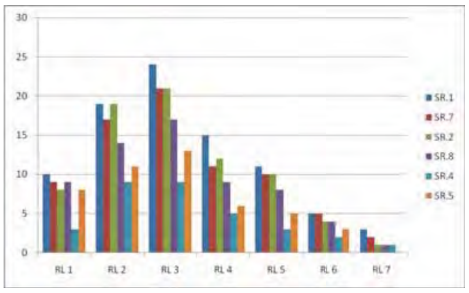
- Encoding of Security Payload
  - AES Encrypted
  - Whirlpool HMAC
  - SHA1 HMAC
  - AES CMAC
- Length of MAC

Security Requirements

Depending on

- the risk of an attack
- the severity of an attack

> choose level of protection EVITA [D2.3,D3.2]



Small exercise
- truncation of MAC increases risk
- number of trials limited by bus and HSM throughput
- limit of failed verifications: roughly 100 per second
- time of P(false-validation-of-MAC=1)=0.5

bits	time to collide
0	0
16	5.5 min
24	23.3 h
32	35.5 weeks
48	44750 years
64	2932747010 years
96	1.25961E+19 years
128	5.40996E+28 years
192	9.97962E+47 years
256	1.84092E+67 years

- Length of MAC:
  - up to 256 bits (for fast buses and critical data)
  - allow truncation down to 32 bits (low speed buses and non-critical data)

Summary and Outlook

- Add security to the "loaded" buses
  - Security Payload depends on requirements
  - Size matters...

- Lots of other protocols within EVITA protocols:
  - On-Board System Integrity Attestation
  - Maintenance: ECU replacement and upgrades
  - Time Synchronization
  - Filtering and Access Control Management
  - Intrusion Detection and Response
- To be found in EVITA [D3.3]

Thanks

Thank you for your attention!

I hope this presentation was interesting and I am

looking forward to your Q U E S T I O N S !

contact: schweppe@eurecom.fr

drive safely.

Hendrik Schweppe
schweppe@eurecom.fr

CAST, July 1 2010,
Darmstadt, Germany

Secure on-board protocols

26

EURECOM

Architecture and Protocol Verification and Attack Analysis

Ludovic Apvrille

Institut Telecom - Telecom ParisTech

Summary

The objective of the European-funded EVITA project is to design, verify, and prototype an architecture for automotive on-board networks where security-relevant components are protected against tampering and sensitive data are protected against compromise.
This presentation focuses on the verification part. The verification process targets two main issues. The first one is the performance impact the security architecture and the cryptographic protocols have on a usual automotive embedded system. The second one is the formal proof that the defined architecture and cryptographic protocols satisfy security properties identified at the first part of the EVITA project. We have addressed those two issues using modeling, simulation and formal verification techniques. More precisely, the EVITA system has been modeled using UML profiles (e.g., TURTLE [1] and DIPLODOCUS [2]) and their related toolkit named TTool [3]. TTool offers a press-button approach to simulation and verification techniques. Performance studies and formal proofs of security are exemplified over EVITA use cases, and a few results are presented.

CV

Ludovic Apvrille obtaine d his engin eering diplo ma and a M.Sc.in Computer Science, Network and Distrib uted Systems specialization, fro m *ENSEIRB* and *ISAE* in 1997 and 1998 , respectively. Then, he completed a Ph.D. at *LAAS-CNRS*, Toulouse, F rance, in th e research group *Software and Tools for Communication*. This Ph. D work was part of a collaboration between the Departmen t of Applied Mathematics and Computer Scien ce at *ISAE* and *Thalès Alenia Space*. After a p ostdoctoral term at Concordia University (Canada) in the *Electrical and Computer Engineering* department, he joined LabSoc as an Assista nt Professo r at Institut Telecom - Telecom ParisTech. H is research interests f ocus on to ols and methods for th e modeling of embedded systems and systems-on-chip. He has been involved in the definition of the TURTLE [1] and DIPLODOCUS [2] UML profiles, and is the main developer of TTool [3].

Contact

Dr. Ludovic Apvrille
Institut Telecom/Telecom ParisTech/COMELEC/LabSoC
B.P. 193, 2229 route des Cretes
06904 Sophia-Antipolis Cedex
France
Tel. +33 (0) 4 93 00 84 06
Fax. +33 (0) 4 93 00 82 00
eMail: ludovic.apvrille@telecom-paristech.fr

References

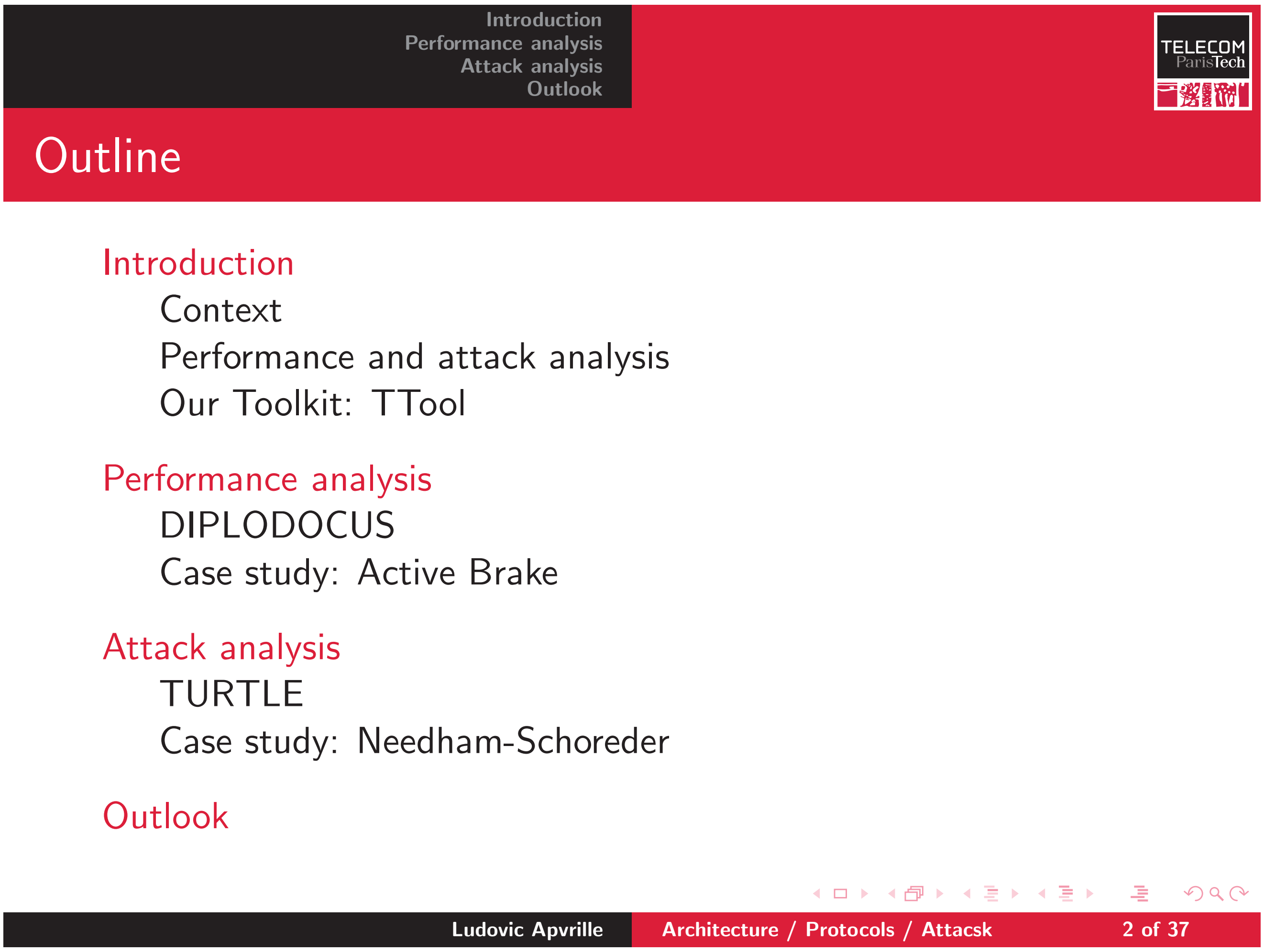
[1] L. Apvrille, J.-P. Courtiat, C. Lohr, P. de Saqui-Sannes , *TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit*, IEEE Transactions on Software Engineering, Vol. 30, No. 7, pp. 473-487, July 2004.
[2] L. Apvrille, W. Muhammad, R. Ameur-Boulifa, S. Coudert and R. Pacalet, *A UML-based Environment for System Design Space Exploration*, 13th IEEE International Conference on Electronics, Circuits and Systems (ICECS’2006), Nice, France, December 2006
[3] L. Apvrille, TTool, http://labsoc.comelec.enst.fr/turtle/ttool.html, 2010

Architecture and Protocol Verification and Attack Analysis

Ludovic Apvrille

Institut Telecom, Telecom ParisTech
ludovic.apvrille@telecom-paristech.fr

July, 1, 2010



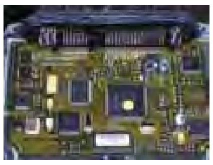
Outline

Outline

On-board Vehicle Systems

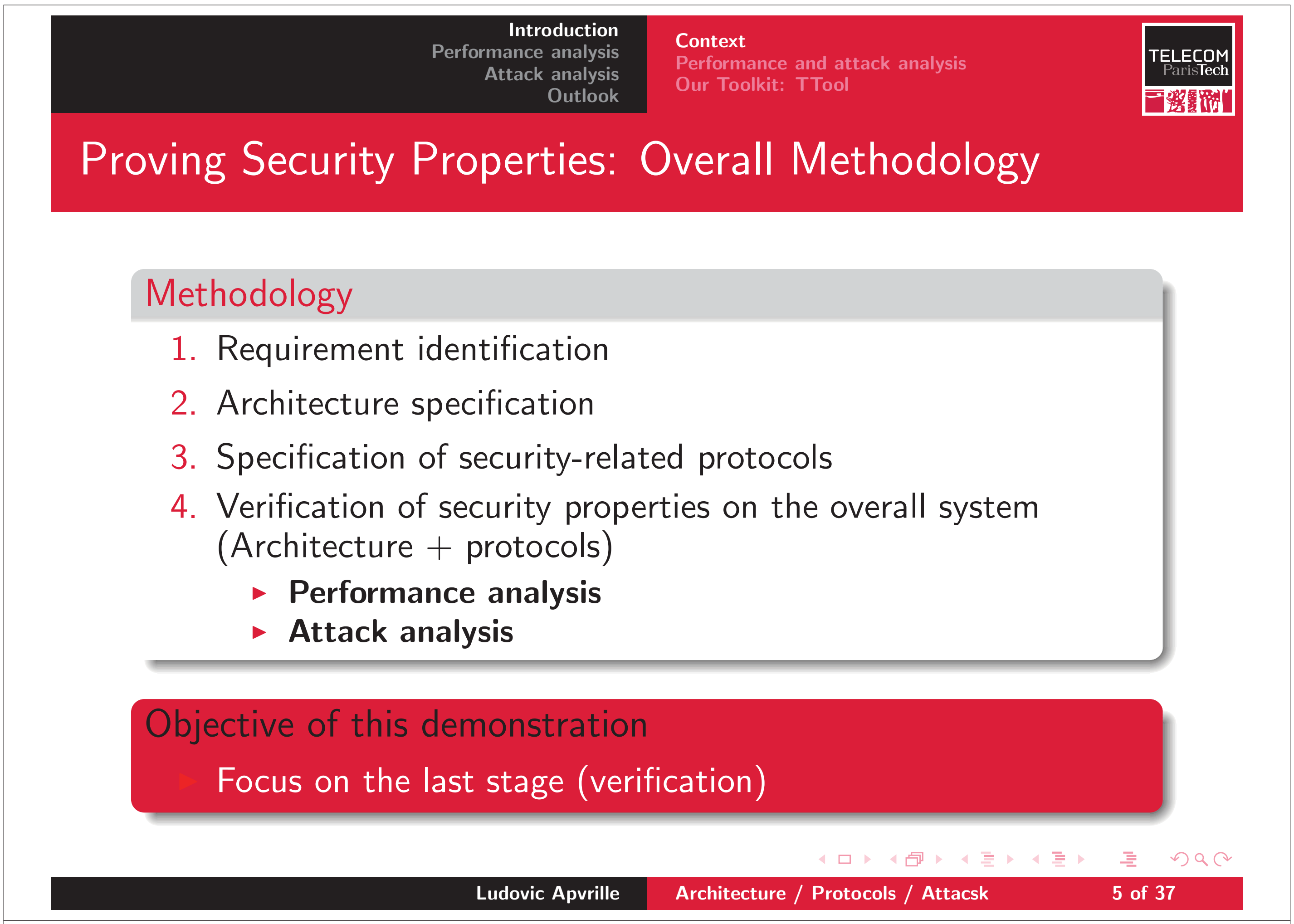
On-board vehicle system

- ECUs (Electronic Control Units) = set of hardware components
  - Execution elements (CPUs, HWAs)
  - Communication elements (e.g., busses)
  - Storage elements (e.g., RAM, flash)
  - I/O devices, including sensors / actuators
- Software components
  - Executed on CPUs



One of EVITA's goals:

Proving security properties on those systems



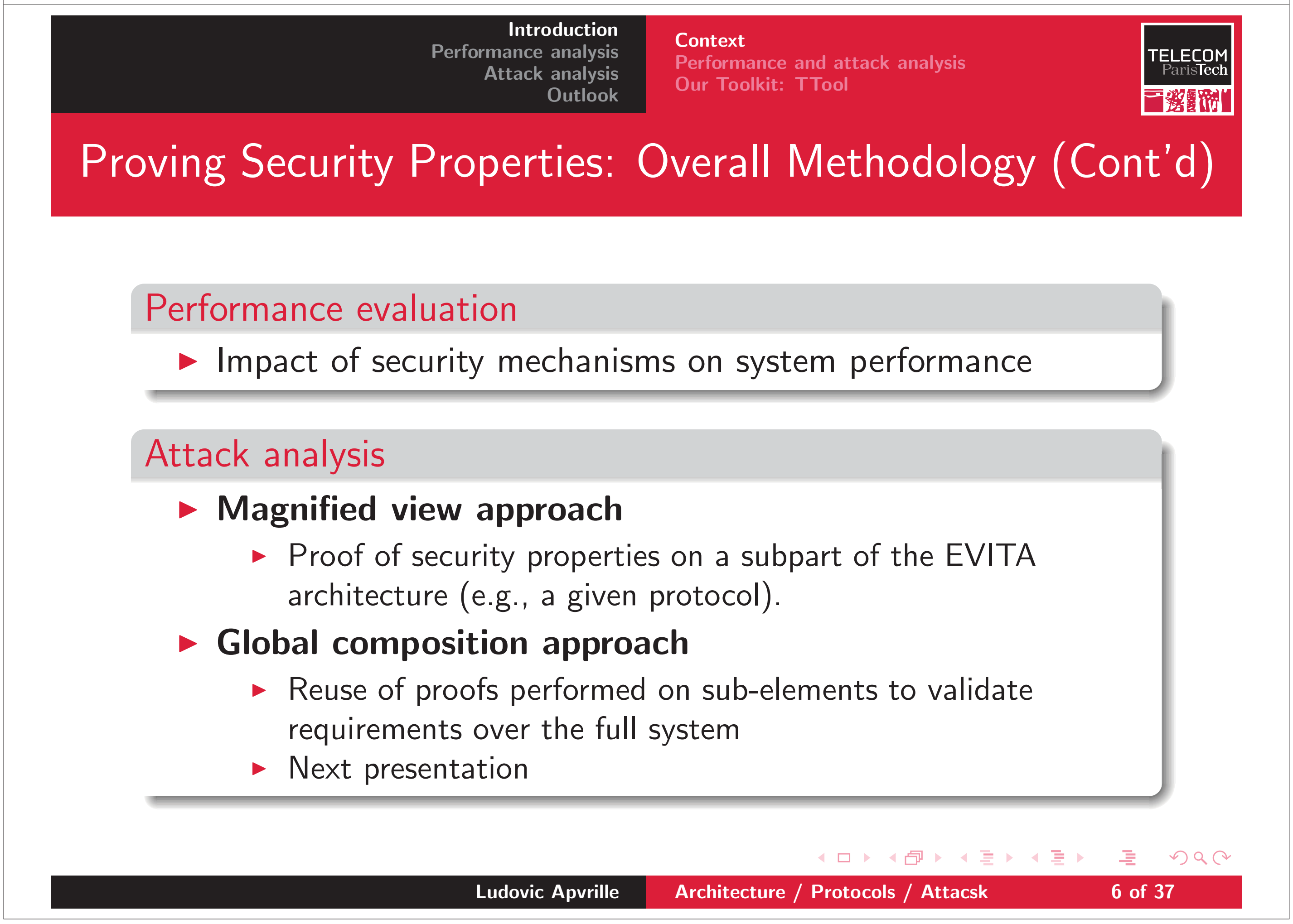
Proving Security Properties: Overall Methodology

Methodology

1. Requirement identification
2. Architecture specification
3. Specification of security-related protocols
4. Verification of security properties on the overall system (Architecture + protocols)
    - **Performance analysis**
    - **Attack analysis**

Objective of this demonstration

- Focus on the last stage (verification)



Proving Security Properties: Overall Methodology (Cont'd)

Performance evaluation

- Impact of security mechanisms on system performance

Attack analysis

- **Magnified view approach**
    - Proof of security properties on a subpart of the EVITA architecture (e.g., a given protocol).
- **Global composition approach**
    - Reuse of proofs performed on sub-elements to validate requirements over the full system
    - Next presentation

Issues

(1) Performance properties

- Impact of the EVITA security architecture on system performance?
  - Cryptographic algorithms and protocols
- Partitioning issue
  - Shall algorithms be software or hardware implemented? Distributed among ECUs or centralized in a given ECU

(2) Security properties

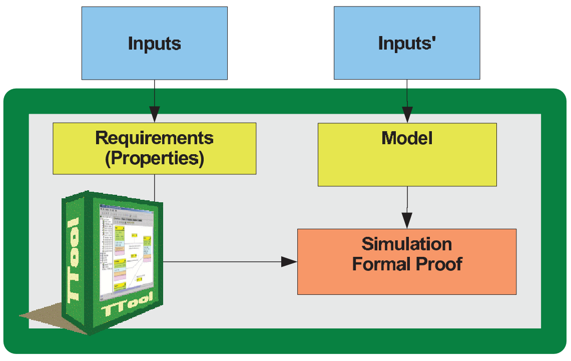
- Security requirements have been previously identified
- Derive attacks from requirements and ...
- Prove that those attacks are not possible in the EVITA infrastucture

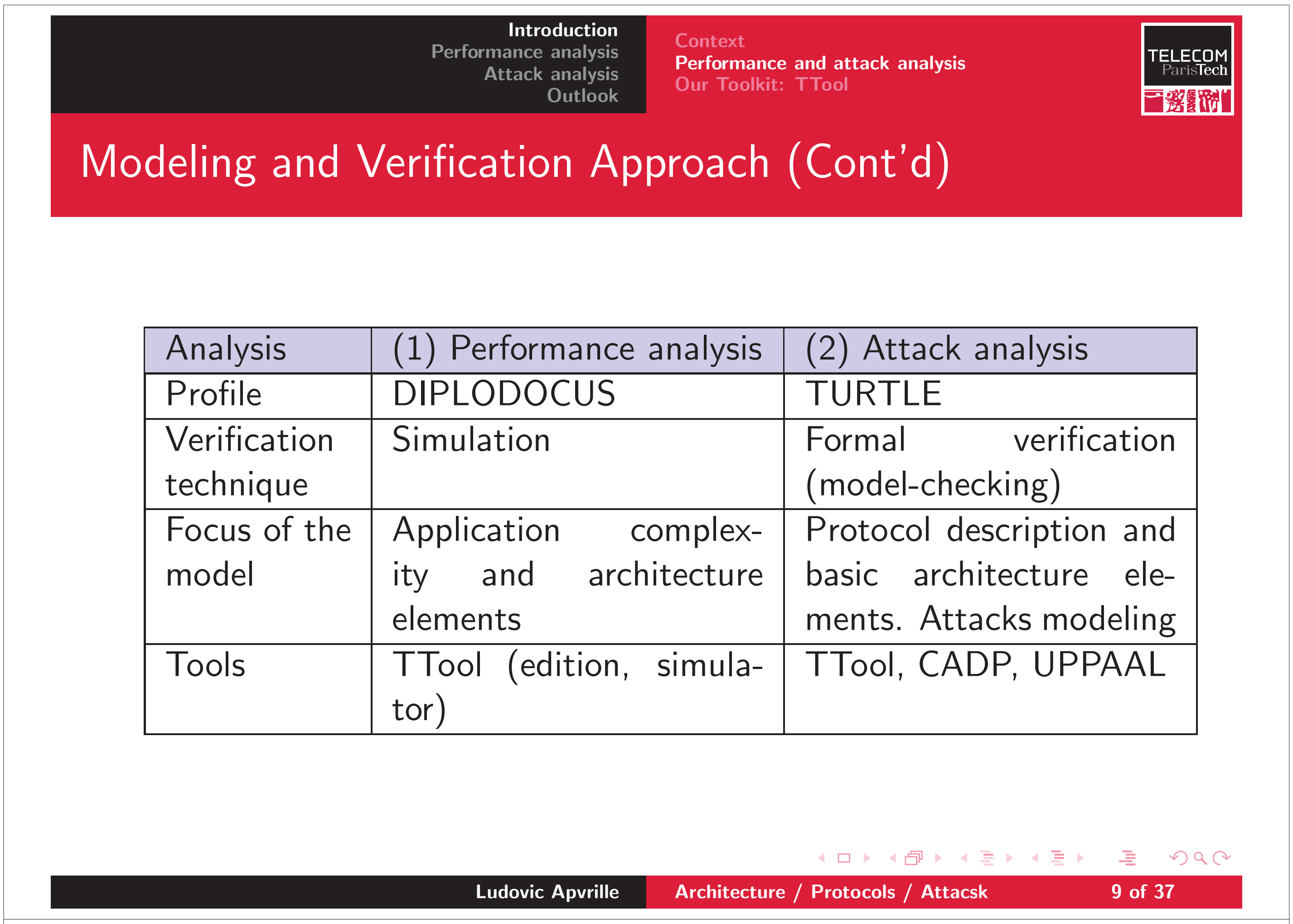
Modeling and Verification Approach

Objective

- Performance evaluation, Attack analysis (magnified view approach)

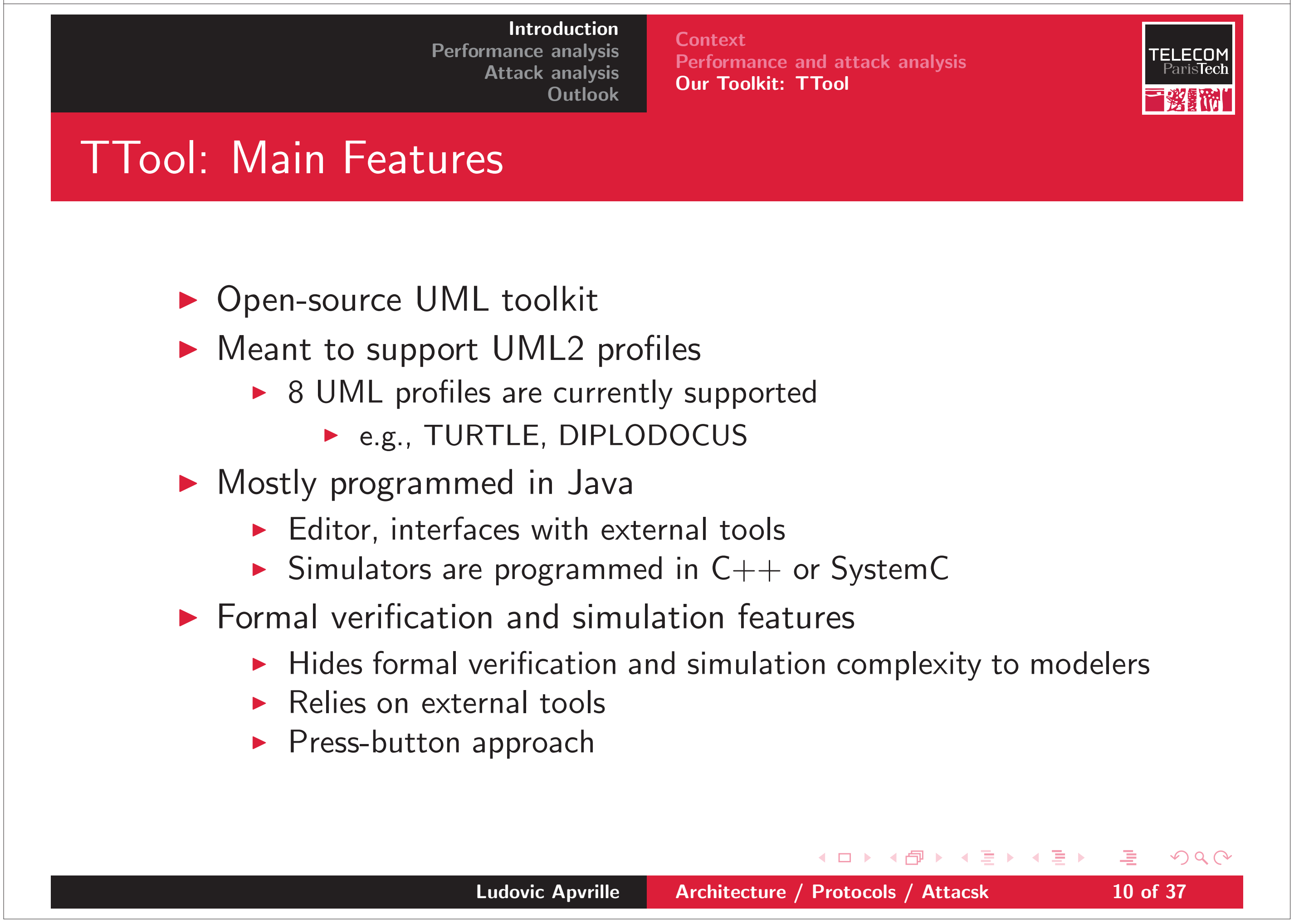
- Consider inputs (e.g., EVITA deliverables)
- Make a model, using e.g. SysML and UML models
- Verify properties using simulation or formal verification techniques





Modeling and Verification Approach (Cont'd)

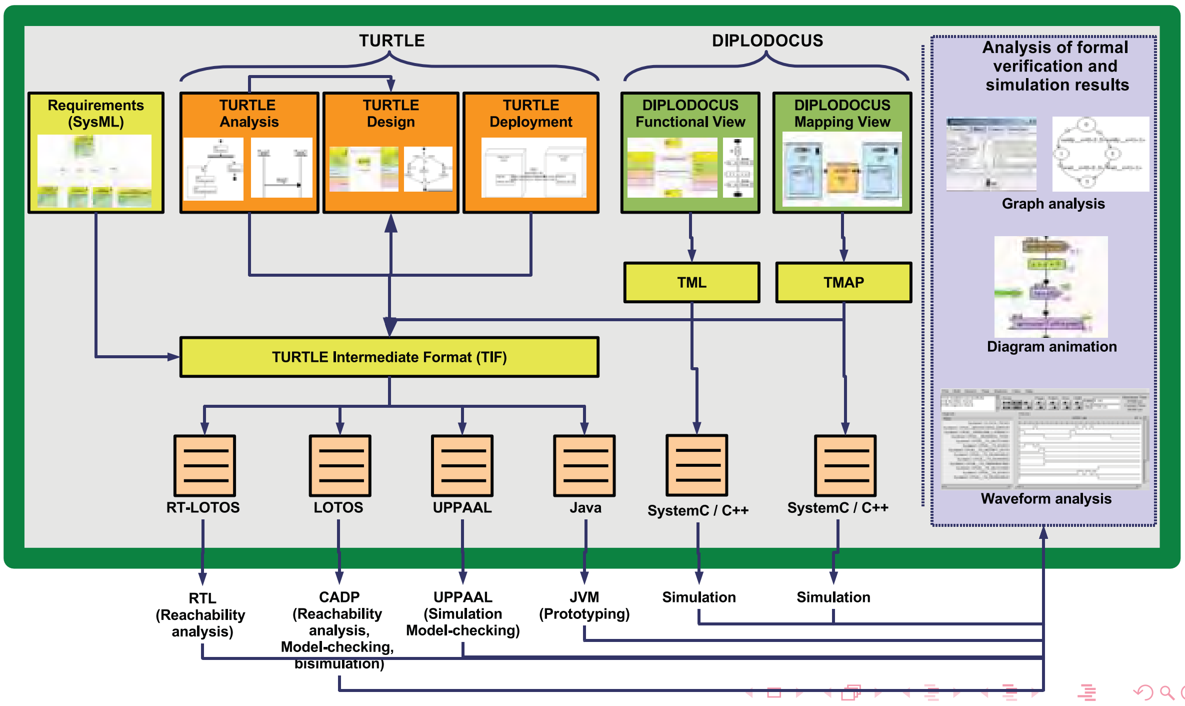
Analysis	(1) Performance analysis	(2) Attack analysis
Profile	DIPLODOCUS	TURTLE
Verification technique	Simulation	Formal verification (model-checking)
Focus of the model	Application complexity and architecture elements	Protocol description and basic architecture elements. Attacks modeling
Tools	TTool (edition, simulator)	TTool, CADP, UPPAAL



TTool: Main Features

- Open-source UML toolkit
- Meant to support UML2 profiles
  - 8 UML profiles are currently supported
    - e.g., TURTLE, DIPLODOCUS
- Mostly programmed in Java
  - Editor, interfaces with external tools
  - Simulators are programmed in C++ or SystemC
- Formal verification and simulation features
  - Hides formal verification and simulation complexity to modelers
  - Relies on external tools
  - Press-button approach

TTool: TURTLE and DIPLODOCUS



Outline

DIPLODOCUS in a Nutshell

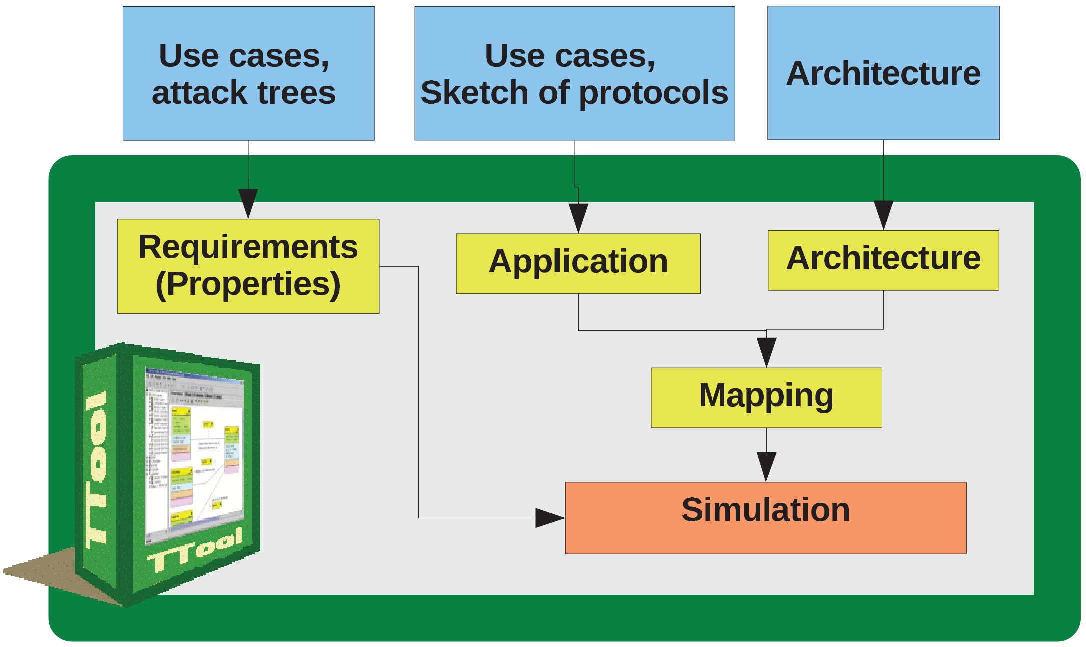
DIPLODOCUS = UML Profile

- System-level Design Space Exploration
- Y-Methodology
- MARTE compliant

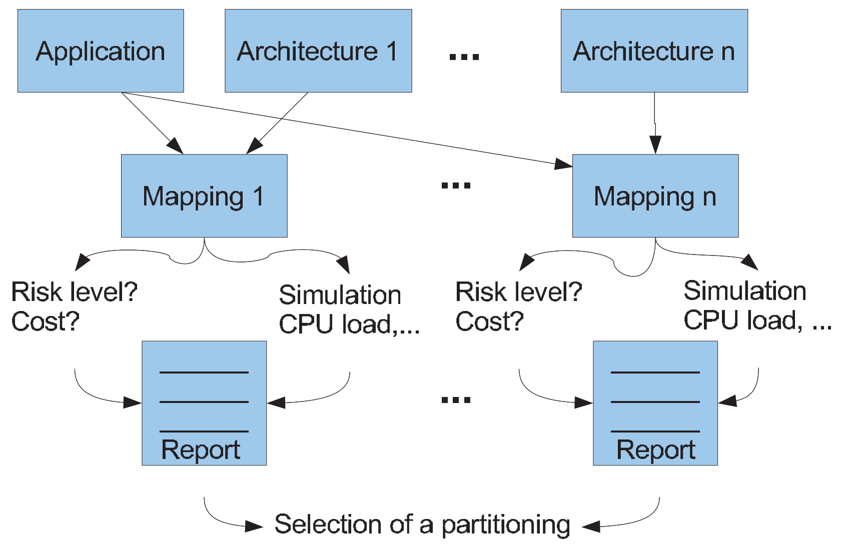
Main features

- Data are abstracted
- Formal semantics
- Very fast simulation support
- Fully supported by an open-source toolkit
  - TTool

DIPLODOCUS: Methodology for Performance Evaluation



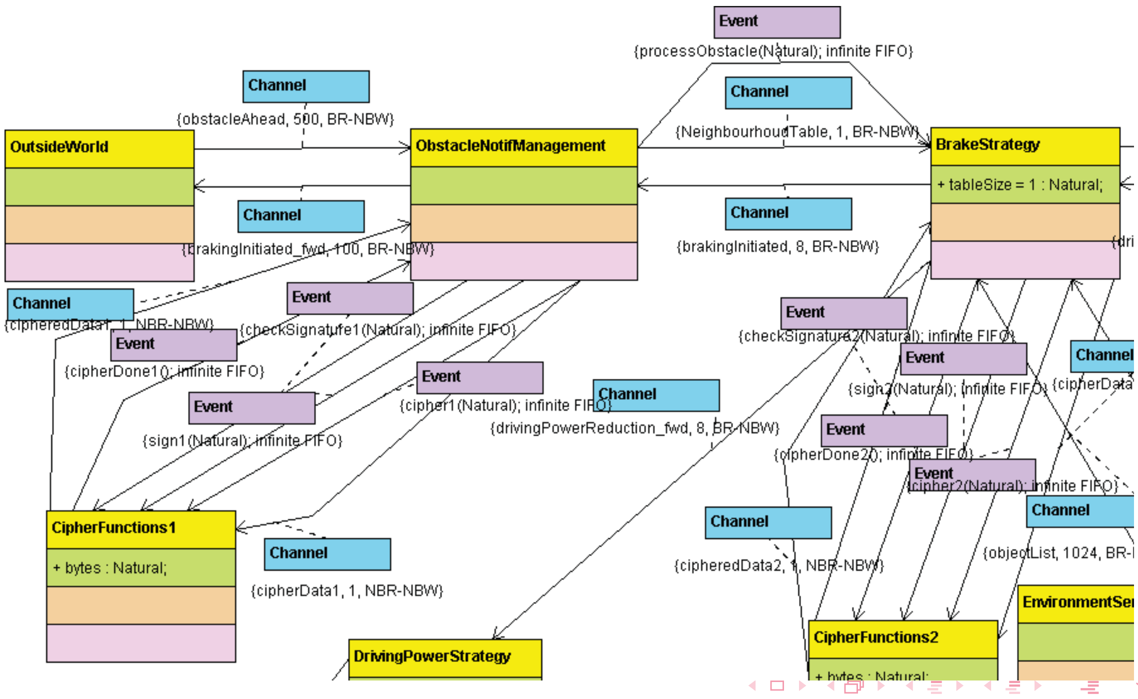
DIPLODOCUS: Methodology for Performance Evaluation (Cont'd)



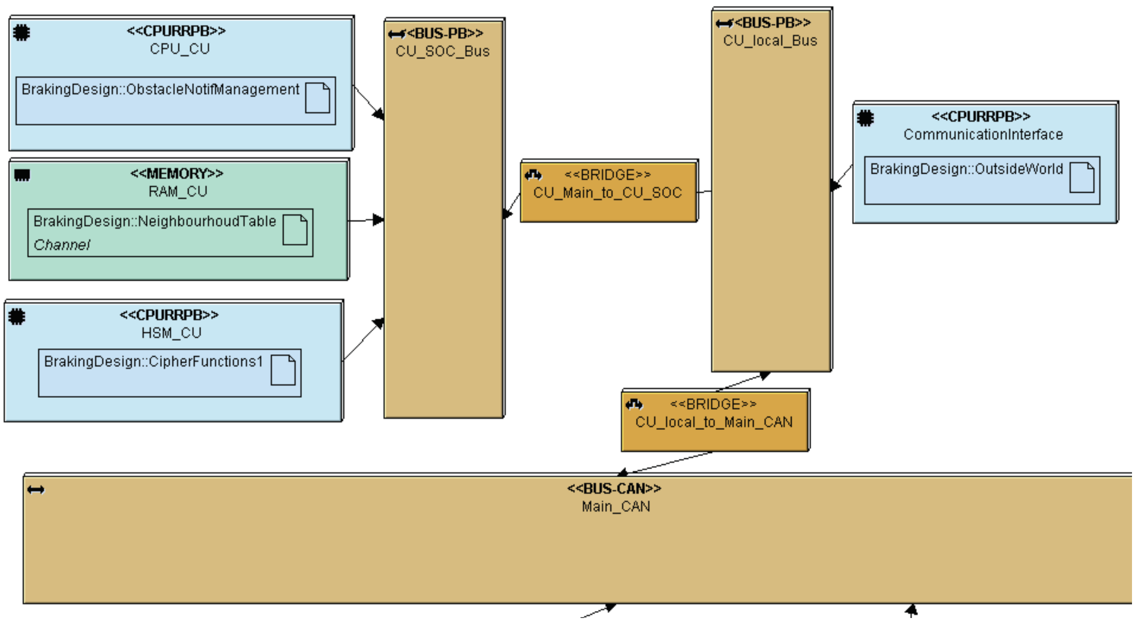
Description of the Active Brake Use Case

- Message sent from one car to another car (car2car)
  - Immediate danger of collision
  - Instant brake manoeuvre
- Message received and checked at Communication Unit level
- Plausibility check at Chassis Safety Controller level
  - If braking is the best solution, a brake order is sent to the brake control unit
    - Power Train Controller is also informed (to decelerate, etc.).
  - Braking information might be forwarded to other neighbour cars

Application Modeling


Event
{processObstacle(Natural); infinite FIFO}
Channel
{obstacleAhead, 500, BR-NBW}
Channel
{NeighbourhoodTable, 1, BR-NBW}
OutsideWorld
ObstacleNotifManagement
BrakeStrategy
+ tableSize = 1 : Natural;
Channel
{brakingInitiated_fwd, 100, BR-NBW}
Channel
{brakingInitiated, 8, BR-NBW}
Channel
{cipheredData1, 1, NBR-NBW}
Event
{checkSignature1(Natural); infinite FIFO}
Event
{cipherDone1(); infinite FIFO}
Event
{checkSignature2(Natural); infinite FIFO}
Event
{sign2(Natural); infinite FIFO}
Channel
{cipherData
Event
{cipher1(Natural); infinite FIFO}
Channel
{drivingPowerReduction_fwd, 8, BR-NBW}
Event
{sign1(Natural); infinite FIFO}
Event
{cipherDone2(); infinite FIFO}
Event
{cipher2(Natural); infinite FIFO}
CipherFunctions1
+ bytes : Natural;
Channel
{cipherData1, 1, NBR-NBW}
Channel
{cipheredData2, 1, NBR-NBW}
Channel
{objectList, 1024, BR-
EnvironmentSe
DrivingPowerStrategy
CipherFunctions2
+ bytes : Natural;
{dri

Architecture Modeling and Mapping



A Few Simulation Results

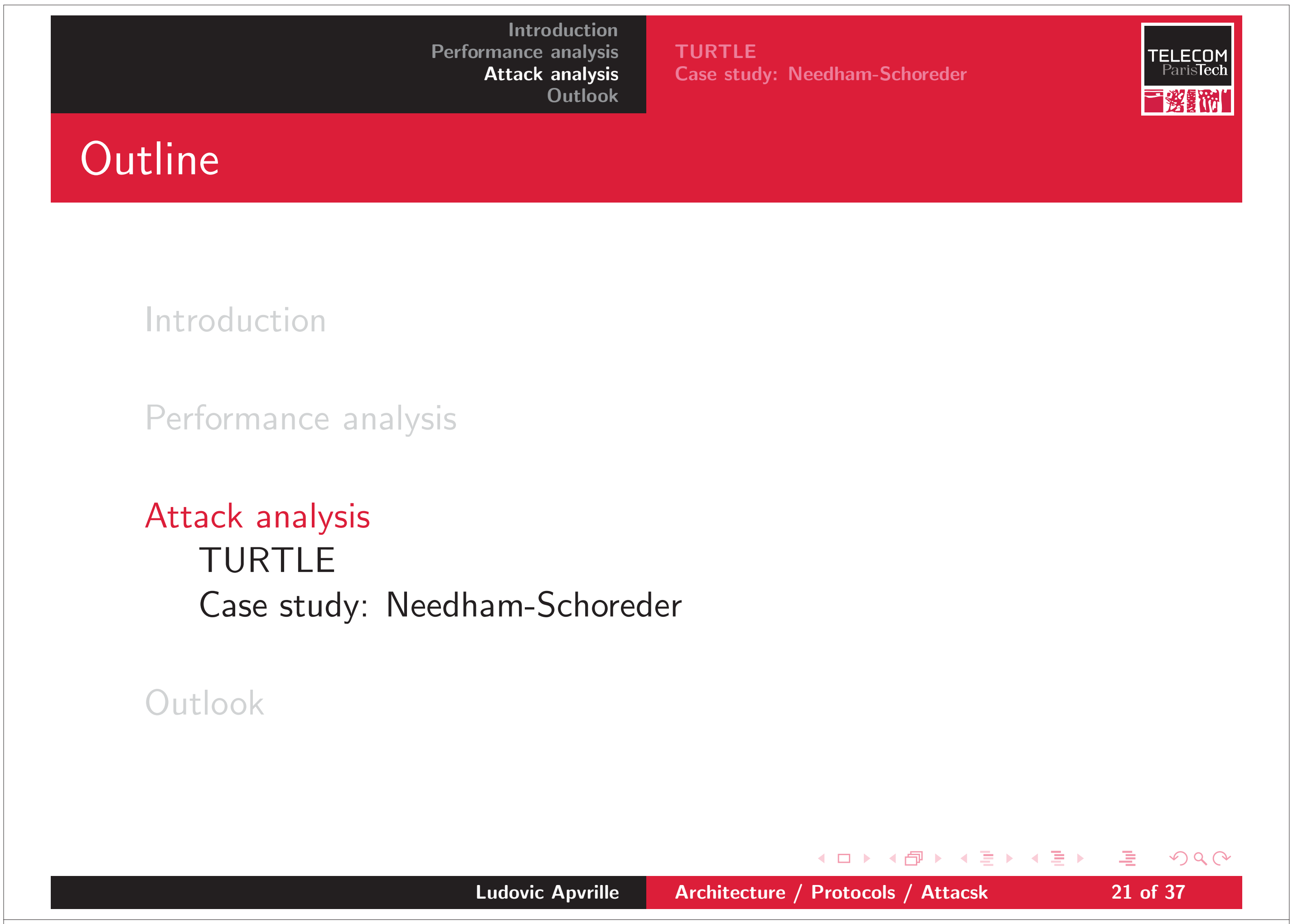
CPUs and Hardware Accelerators

CPU	Load	Contention delay
Load_Emulation	0.15711	29973
CPU_CU	0.11244	0
HSM_CU	0.11939	0
CPU_BCU	0.00010	6806
HSM_BCU	0.00004	0
CPU_PTC	0.00018	0
CPU_ChassisSensor	0.00035	200000
CPU_EnvSensor	0.01115	5818
HSM_CSC	0.11827	0
...	...	...

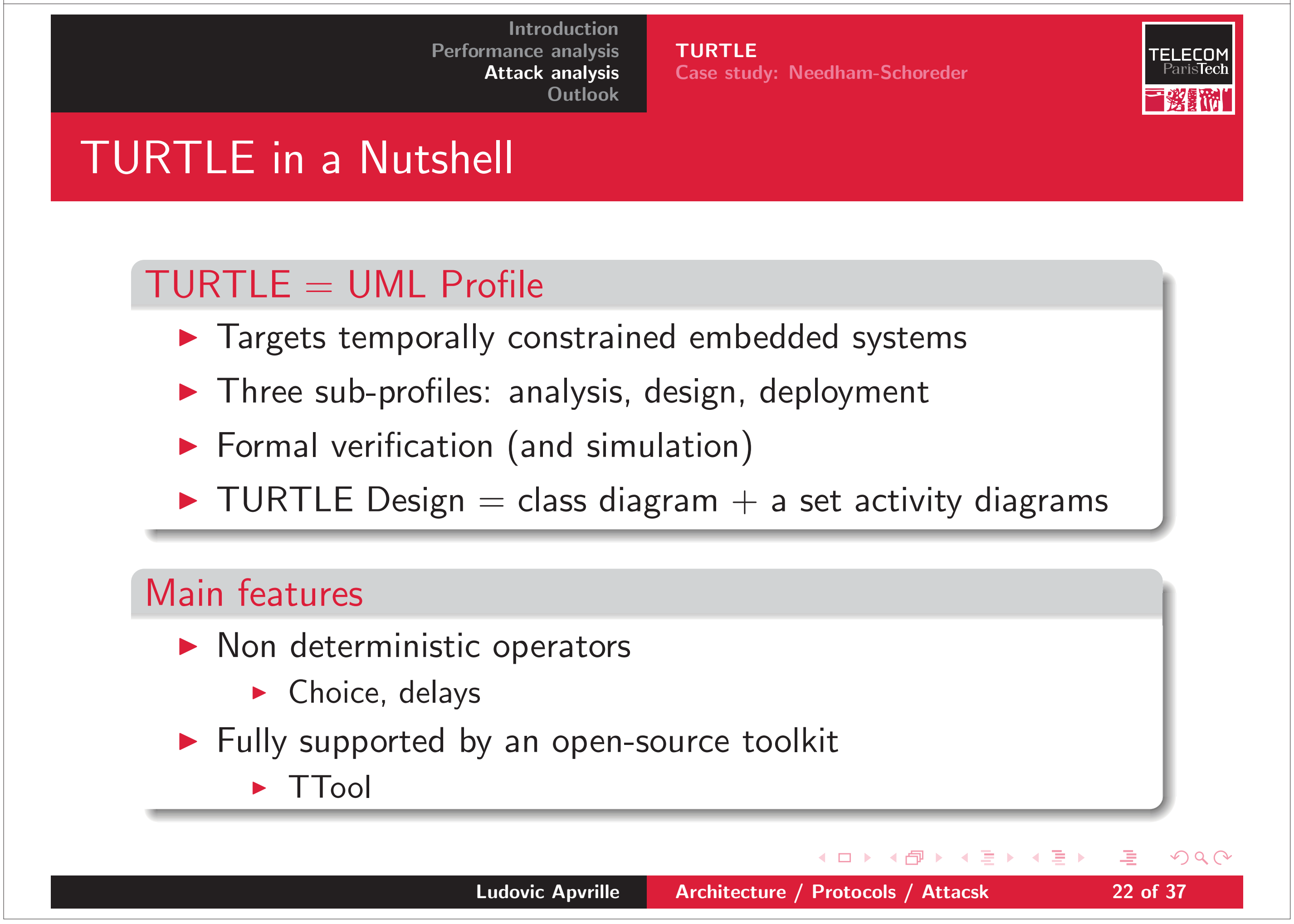
A Few Simulation Results (Cont'd)

Buses

Bus	Load
BCU_local_Bus	0.00017
CSC_local_Bus	0.56926
PTC_local_Bus	0.00026
CU_local_Bus	0.55783
CU_SOC_Bus	0.78811
Main_CAN	0.71469
CSC_SOC_bus	0.74216
...	...



Outline



TURTLE in a Nutshell

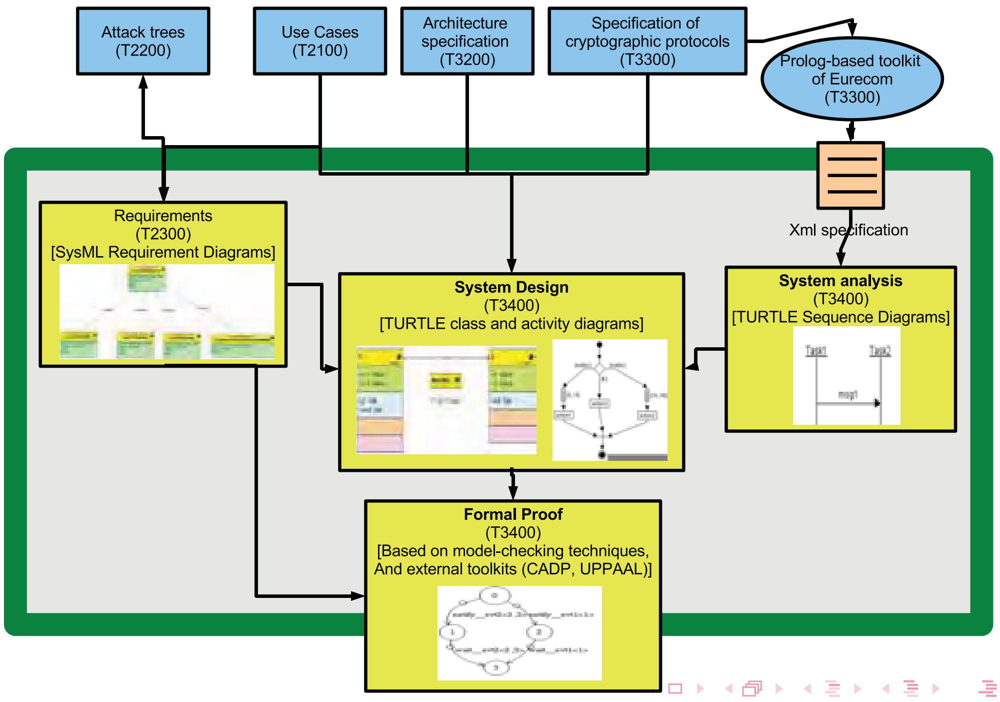
TURTLE = UML Profile

- Targets temporally constrained embedded systems
- Three sub-profiles: analysis, design, deployment
- Formal verification (and simulation)
- TURTLE Design = class diagram + a set activity diagrams

Main features

- Non deterministic operators
  - Choice, delays
- Fully supported by an open-source toolkit
  - TTool

TURTLE: Methodology for Attack Analysis


Attack trees (T2200)
Use Cases (T2100)
Architecture specification (T3200)
Specification of cryptographic protocols (T3300)
Prolog-based toolkit of Eurecom (T3300)
Xml specification
Requirements (T2300) [SysML Requirement Diagrams]
System Design (T3400) [TURTLE class and activity diagrams]
System analysis (T3400) [TURTLE Sequence Diagrams]
Task1
Task2
msg1
Formal Proof (T3400) [Based on model-checking techniques, And external toolkits (CADP, UPPAAL)]

Model: Main Principles

Modeled elements

- Hardware elements in ECUs
  - HSM
  - Communication networks
- Software elements
  - Protocol stack at involved ECUs

Proving security properties

- Observer technique
- Model-checking is used to search for a given action

Description of the Case Study

Why this case study (not directly related to EVITA)?

- Illustrate proofs of security requirements with TURTLE
- A small yet representative system
- Contains all interesting concepts:
  - Entities, network elements, crypto functions and protocols, attacks

Description

- Alice and Bob, who want to exchange a confidential data
- Use the Needham-Schroeder protocol to setup a session key K, using a trusted server
- Then, Bob sends the data to Alice using K

The Needham-Schroeder Protocol

Description

A represents Alice, B Bob, S the Server; R_X is a random number generated by X, and K_{XY} a key used by X and Y to cipher / decipher information with a symmetric encryption algorithm

1. $A \rightarrow S : A, B, R_A$
2. $S \rightarrow A : \{R_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$
3. $A \rightarrow B : \{K_{AB}, A\}_{K_{BS}}$
4. $B \rightarrow A : \{R_B\}_{K_{AB}}$
5. $A \rightarrow B : \{R_B - 1\}_{K_{AB}}$

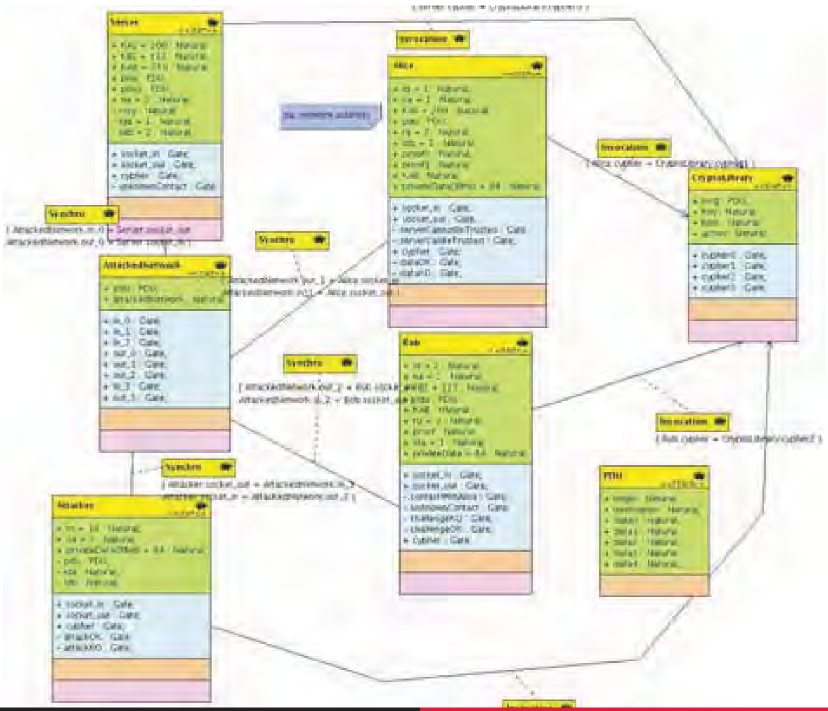
Requirement *req1*

The data sent by Bob to Alice shall be confidential.

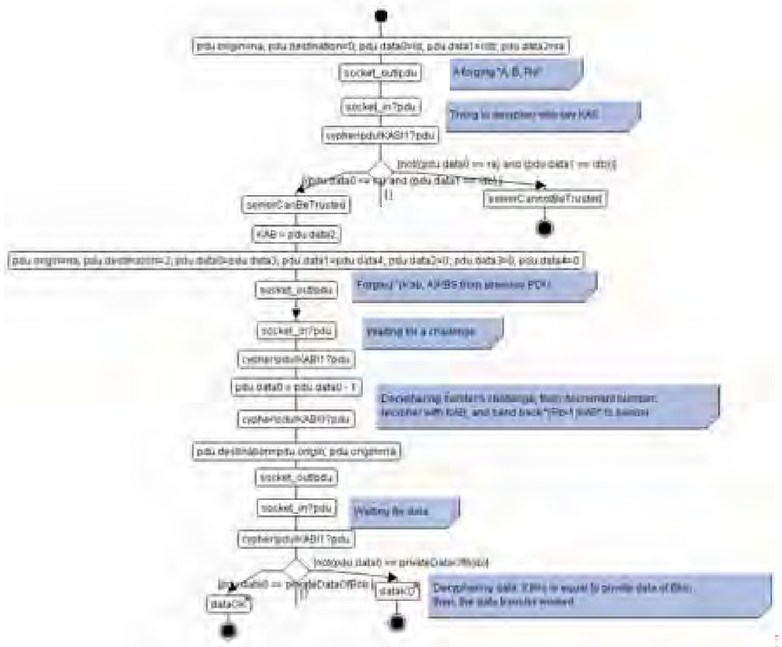
Attacks on the Needham-Schroeder Protocol

- Several known attacks against Needham-Schroeder
- Considered attack: *S. Gurgens et al., "Role based specification and security analysis of cryptographic protocols using asynchronous product automata", Database and Expert Systems Applications, Sept. 2002.*
  (Cx denotes an attacker pretending to be an entity x):
  1. $A \rightarrow C_S : A, B, R_A$
  2. $C_B \rightarrow S : B, A, R_C$
  3. $S \rightarrow C_B : \{R_C, A, K_{BA}, \{K_{BA}, B\}_{K_{AS}}\}_{K_{BS}}$
  4. $C_A \rightarrow B : \{R_C, A, K_{BA}, \{K_{BA}, B\}_{K_{AS}}\}_{K_{BS}}$
  5. $B \rightarrow C_A : \{R_B\}_{R_C}$
  6. $C_A \rightarrow B : \{R_B - 1\}_{R_C}$
- From that attack, *req1* can be proved as non-satisfied.

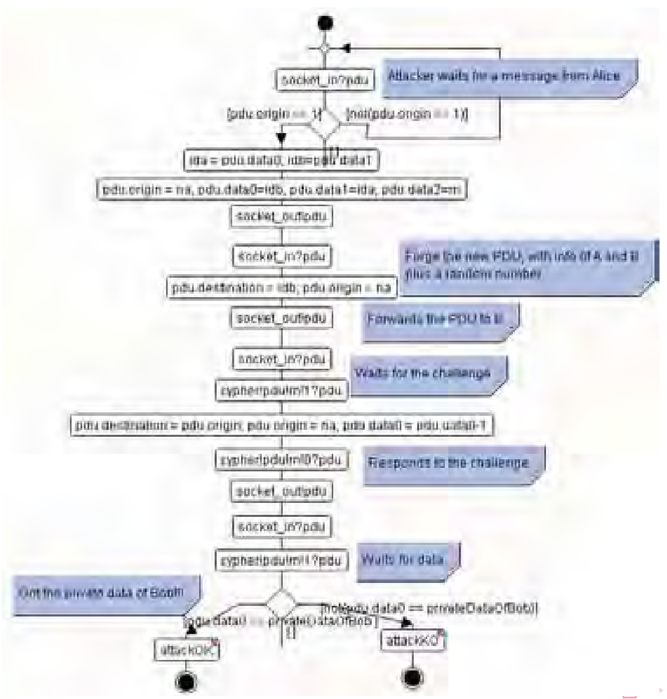
Class Diagram



Activity Diagram of Alice

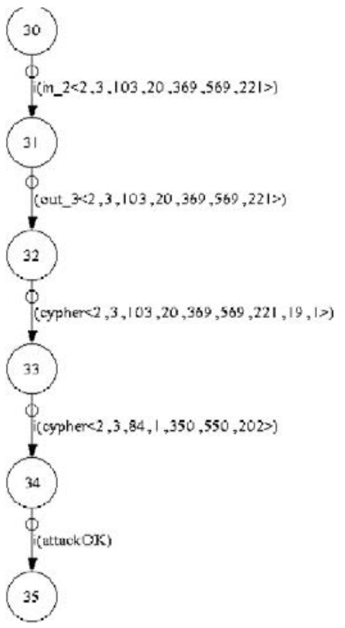


Activity Diagram of Attacker


socket_in?pdu
Attacker waits for a message from Alice
[pdu.origin == 1]
[not(pdu.origin == 1)]
ida = pdu.data0, idb=pdu.data1
pdu.origin = na, pdu.data0=idb, pdu.data1=ida, pdu.data2=ni
socket_out!pdu
socket_in?pdu
Forge the new PDU, with info of A and B plus a random number
pdu.destination = idb; pdu.origin = na
socket_out!pdu
Forwards the PDU to B
socket_in?pdu
Waits for the challenge
cypherpdu!m1?pdu
pdu.destination = pdu.origin; pdu.origin = na, pdu.data0 = pdu.data0-1
cypherpdu!m0?pdu
Responds to the challenge
socket_out!pdu
socket_in?pdu
cypherpdu!m1?pdu
Waits for data
Got the private data of Bob!!
[pdu.data0 == privateDataOfBob]
[not(pdu.data0 == privateDataOfBob)]
[]
attackOK
attackKO

Formal Verification with CADP

Reachability graph

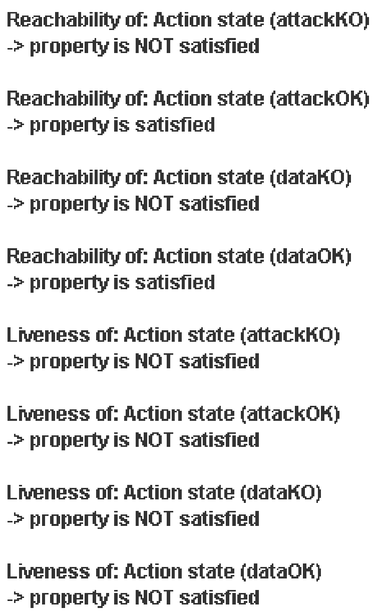

30
i(in_2<2,3,103,20,369,569,221>)
31
(out_3<2,3,103,20,369,569,221>)
32
(cypher<2,3,103,20,369,569,221,19,1>)
33
i(cypher<2,3,84,1,350,550,202>)
34
i(attackOK)
35

Verification approach

- Generate a Reachability Graph using CADP
- Minimize of the reachability graph
- Search for traces containing the *attackOK* and *attackKO* actions

Formal Verification with UPPAAL

Network can be probed


Reachability of: Action state (attackKO)
-> property is NOT satisfied

Reachability of: Action state (attackOK)
-> property is satisfied

Reachability of: Action state (dataKO)
-> property is NOT satisfied

Reachability of: Action state (dataOK)
-> property is satisfied

Liveness of: Action state (attackKO)
-> property is NOT satisfied

Liveness of: Action state (attackOK)
-> property is NOT satisfied

Liveness of: Action state (dataKO)
-> property is NOT satisfied

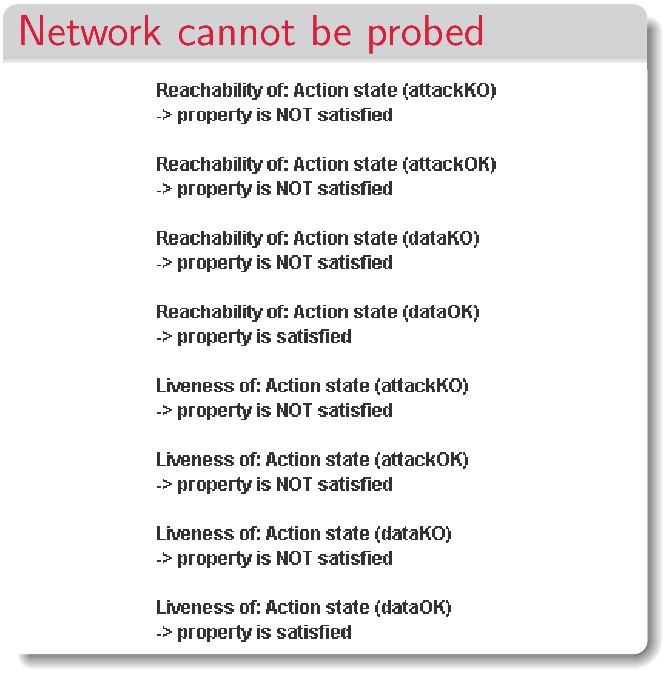
Liveness of: Action state (dataOK)
-> property is NOT satisfied

Verification approach

- Select actions of interest on the UML model
- Automatically invoke UPPAAL
- Search the accessibility and liveness of selected actions

Formal Verification with UPPAAL (Cont.)

Network cannot be probed


Reachability of: Action state (attackKO)
-> property is NOT satisfied

Reachability of: Action state (attackOK)
-> property is NOT satisfied

Reachability of: Action state (dataKO)
-> property is NOT satisfied

Reachability of: Action state (dataOK)
-> property is satisfied

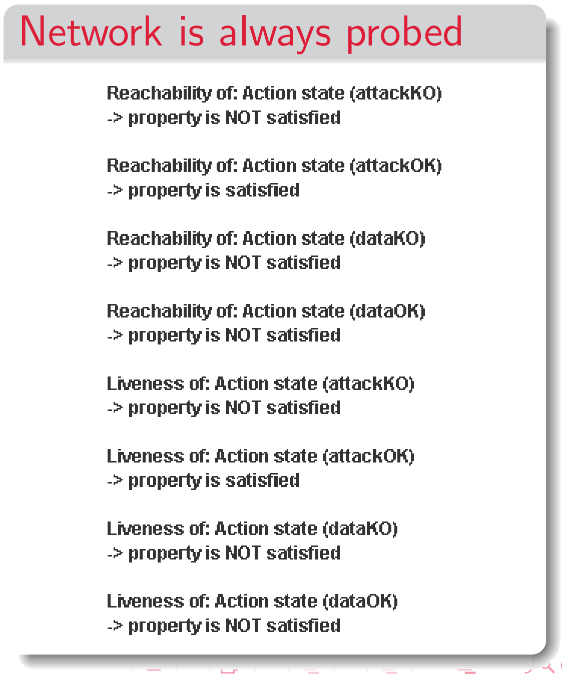
Liveness of: Action state (attackKO)
-> property is NOT satisfied

Liveness of: Action state (attackOK)
-> property is NOT satisfied

Liveness of: Action state (dataKO)
-> property is NOT satisfied

Liveness of: Action state (dataOK)
-> property is satisfied

Network is always probed


Reachability of: Action state (attackKO)
-> property is NOT satisfied

Reachability of: Action state (attackOK)
-> property is satisfied

Reachability of: Action state (dataKO)
-> property is NOT satisfied

Reachability of: Action state (dataOK)
-> property is NOT satisfied

Liveness of: Action state (attackKO)
-> property is NOT satisfied

Liveness of: Action state (attackOK)
-> property is satisfied

Liveness of: Action state (dataKO)
-> property is NOT satisfied

Liveness of: Action state (dataOK)
-> property is NOT satisfied

Outline

Results

Fully integrated environment for the design and verification of embedded systems

- Based on UML / SysML, open-source toolkit (TTool)
- Formal proof can address
  - Safety and security properties
    - Proofs achieved on authenticity, confidentiality, freshness
  - Functional and non functional properties

Recall on methodological stages

- Requirement capture (SysML, DIPLODOCUS)
  - Attack trees, definition and organization of requirements
- Performance analysis (DIPLODOCUS)
- Attack analysis, magnified view approach (TURTLE)

A Few Industrial Case Studies with TTool

- MPEG coders and decoders (Texas Instruments)
- LTE (Freescale)
- Partitioning in vehicle embedded systems, formal proof of security properties (EVITA project)
- Many other systems!

To Go Further ...

TTool

- Type *TTool UML* under google
- And click on the *I am lucky* button!

DIPLODOCUS, TURTLE

- *DIPLODOCUS UML*
- *TURTLE UML*

Towards Model-Driven Security Engineering

Andreas Fuchs

Fraunhofer Institute SIT, Darmstadt

Zusammenfassung

Cooperating systems typically base decisions on information from their own components as well as on input from other systems. Saf ety critical decisions based on cooperative reasoning, such as automatic emergency braking of vehicles, raise severe concerns to security issues.
This talk ad dresses the problem of designing secure syste ms starting from an abstract set o f requirements towards a concrete implementati on and distribution among several entities. The presented approach that originates from the se curity engineering of the project EVITA utilize s the possibilities of formal security proofs and combines them with methodologies from model driven engineering. Th e presented work has b y now been adapted in other proje cts su ch as TERESA an d will be fur ther elaborated on in future works, attempting to establish a security engineering approach that is supported by formal methods.

CV

Andreas Fuchs stud ied computer science at th e Technica l University of Darmstadt, German y and the University of Massachu setts, USA and re ceived his Diplom in 2008 at the former. His research focuses on the topics of Trusted Computing and Tr usted Platforms as well as Formal Methods in Security En gineering. I n the past, he conduct ed research on scalability issues in TPM Re mote Attestatio ns and wa s involved with the de velopment of a library of se curity solutions for AmI en vironments in the IST project SERENIT Y. His current interests are focused on the deve lopment an d applicatio n of formal security analysis methods to the model-driven engineering process within the FP7 projects EVITA and TERESA.

Kontakt

Andreas Fuchs
Fraunhofer Institute for Secure Information Technology
Rheinstr. 75
64295 Darmstadt
Germany
Email: andreas.fuchs@sit.fraunhofer.de

Literatur

[1] C. Jouvray, A. Kung, M. Sall, A. Fuchs, S. Gürgens, and R. Rieke. Security and trust model. Deliverable D3.1 of EVITA, 2010.

[2] A. Fuchs, S.Gürgens and C. Rudolph. A Formal Notion of Trust – Enabling Reasoning about Security Properties. In Proceedings of the 4th IFIP WG 11.11 International Conference, IFIPTM 2010, Morioka, Japan, Springer, June 2010.

[3] D.C. Schmidt. Model-Driven Engineering. IEEE Computer 39 (2), February 2006.

[4] Object Modelling Group. Model Driven Architecture Specification. http://www.omg.org/mda

[5] Project Teresa. http://www.teresa-project.org

Towards Model-Driven Security Engineering

Andreas Fuchs
Fraunhofer Institute for Secure Information Technology
Rheinstraße 75
64295 Darmstadt, Germany

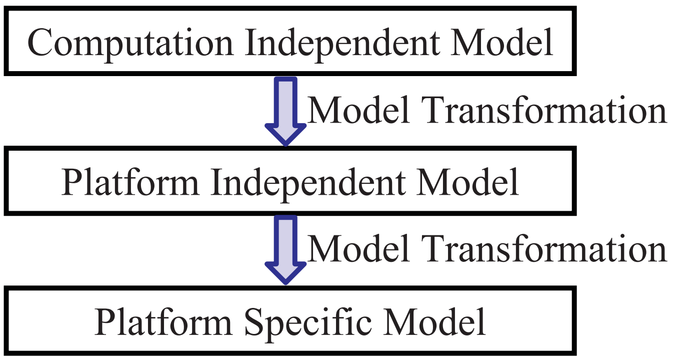
CAST-Workshop
“Mobile Security for Intelligent Cars”
Darmstadt, July 1st 2010

Overview

- What is Model-Driven Engineering ?
- Model-Driven Engineering in the Context of Intelligent Cars
- Formal Methods in Security Engineering
- Consolidation and Integration of Approaches
- Evita’s Security Engineering Process
- Future Work

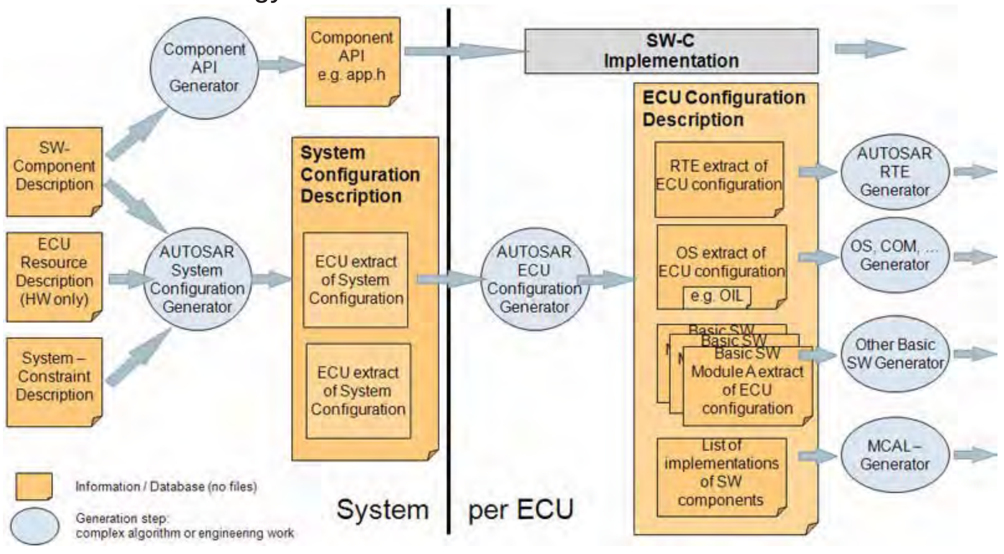
What is Model-Driven Engineering ?

- Software development methodology with focus on creating models, or abstractions, w.r.t. particular domain concepts
- Most known: Model-Driven Architecture by Object Modeling Group (UML-based)
- Refinement of Models from Abstract to Concrete



Model-Driven Engineering in the Context of Intelligent Cars

- Autosar Methodology


Component API Generator
Component API e.g. app.h
SW-C Implementation
SW-Component Description
ECU Resource Description (HW only)
System – Constraint Description
AUTOSAR System Configuration Generator
System Configuration Description
ECU extract of System Configuration
ECU extract of System Configuration
AUTOSAR ECU Configuration Generator
ECU Configuration Description
RTE extract of ECU configuration
OS extract of ECU configuration
e.g. OIL
Basic SW
Basic SW
Basic SW Module A extract of ECU configuration
List of implementations of SW components
AUTOSAR RTE Generator
OS, COM, ... Generator
Other Basic SW Generator
MCAL – Generator
Information / Database (no files)
Generation step: complex algorithm or engineering work
System
per ECU

(Source: Autosar Homepage)

Formal Methods in Security Engineering I

- Known e.g. from formal model checking, a technique of security verification.
- Attempt to provide formal definitions for security properties.
- Allows for reasoning about security properties without the problem of misinterpretation.

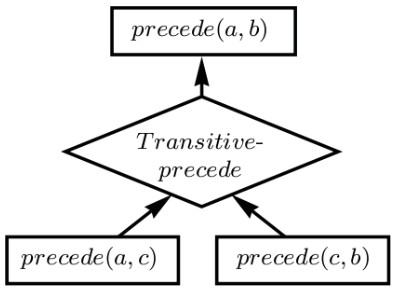
- Security Engineering not that well developed.
  (see e.g. Serenity‘s Security Engineering Manifesto)
- Attempt to establish security through toolboxes and refinements.

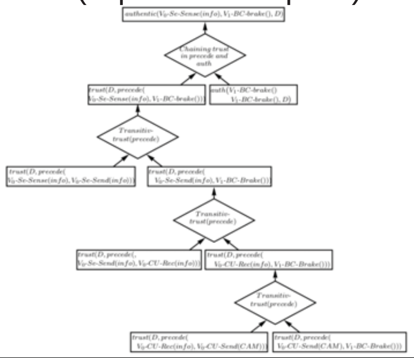
Formal Methods in Security Engineering II

- Language of Formal Methods is rather complex:

  Let S be a system as defined in Definition 10 of [27] which satisfies $precede(x, b)$, let B by the system's behaviour. Then for all $\omega \in B$, $b \in alph(\omega)$ implies $x \in alph(\omega)$. Further, since by assumption $precede(a, x)$ holds in S, for all $\omega \in B$, $x \in alph(\omega)$ implies $a \in alph(\omega)$. Hence we have $b \in alph(\omega)$ implying $a \in alph(\omega)$ which corresponds to $precede(a, b)$ holding for S. $\square$

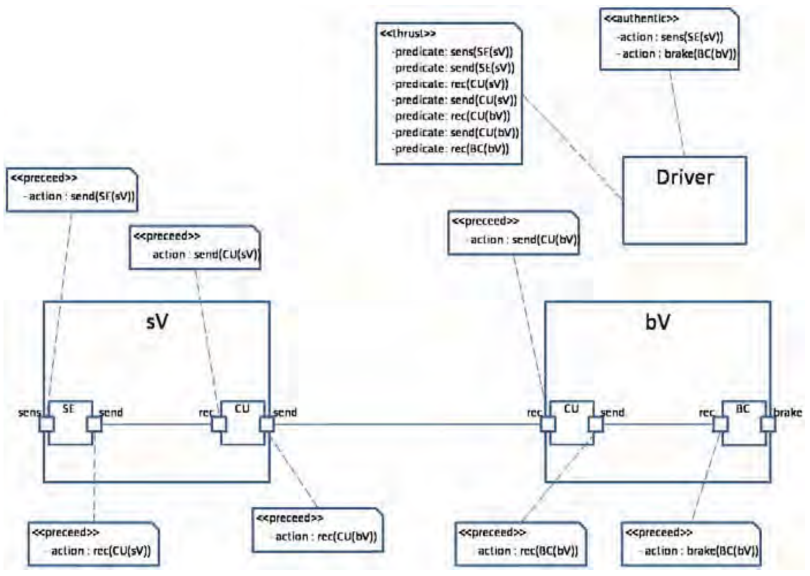
- Graphical Representations easier to comprehend (esp. for non-experts):





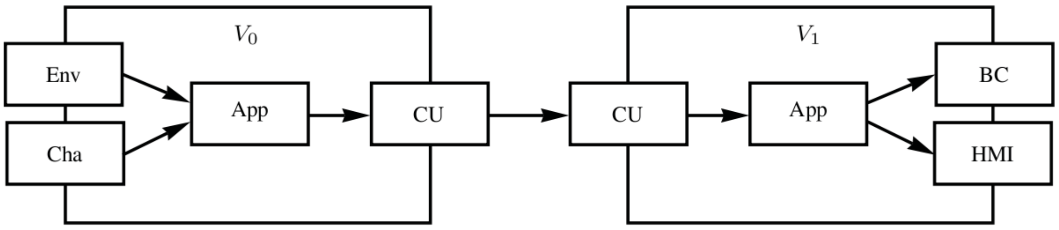
authentic(V_0-Se-Sense(info), V_1-BC-brake(), D)
Chaining trust in precede and auth
trust(D, precede(V_0-Se-Sense(info), V_1-BC-brake()))
auth(V_1-BC-brake() V_1-BC-brake(), D)
Transitiv-trust(precede)
trust(D, precede(V_0-Se-Sense(info), V_0-Se-Send(info)))
trust(D, precede(V_0-Se-Send(info), V_1-BC-Brake()))
Transitiv-trust(precede)
trust(D, precede(, V_0-Se-Send(info), V_0-CU-Rec(info)))
trust(D, precede(V_0-CU-Rec(info), V_1-BC-Brake()))
Transitiv-trust(precede)
trust(D, precede(V_0-CU-Rec(info), V_0-CU-Send(CAM)))
trust(D, precede(V_0-CU-Send(CAM), V_1-BC-Brake()))

Consolidation and Integration of Approaches

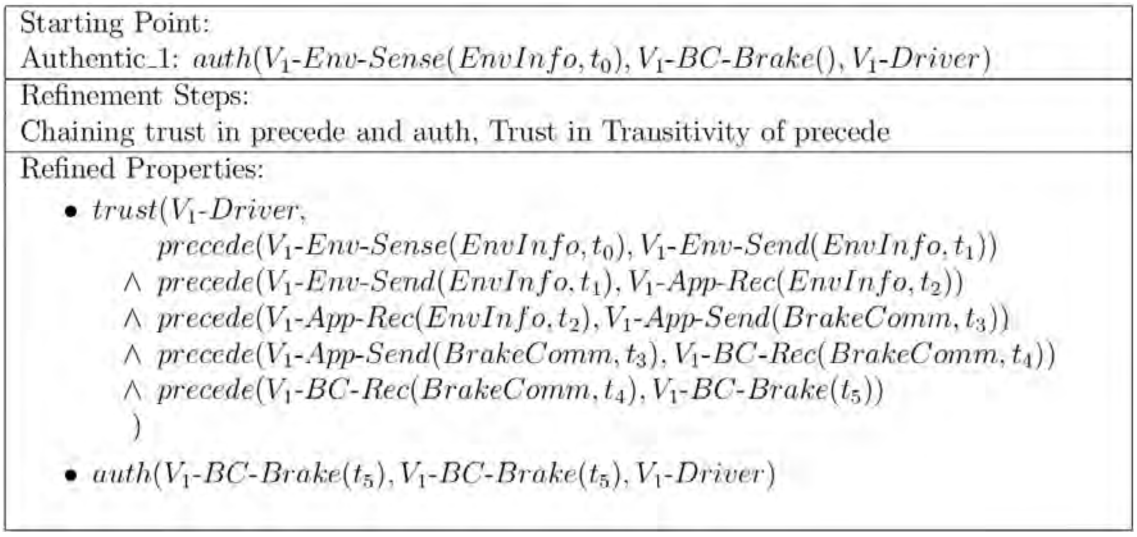


Evita’s Security Engineering Process I

Agent abbreviations:	Data abbreviations:
$V_0 := sensingVehicle$	$Pos := Position\text{-}Information$
$V_1 := brakingVehicle$	$EnvInfo := Environment - Information$
$Env := Environment - Sensor$	$VeDy := Vehicle\text{-}Dynamics$
$Cha := Chassis - Sensor$	$CAM := Car2X - Awareness - Message$
$App := Application\text{-}ECU$	$LDW := Local\text{-}Danger\text{-}Warning\text{-}Message$
$CU := CommunicationUnit$	$Warn := Driver\text{-}Warning\text{-}Message$
$HMI := Human - Machine - Interface$	
$BC := BrakeController$	$D := Driver\ of\ V_1$



Evita’s Security Engineering Process II



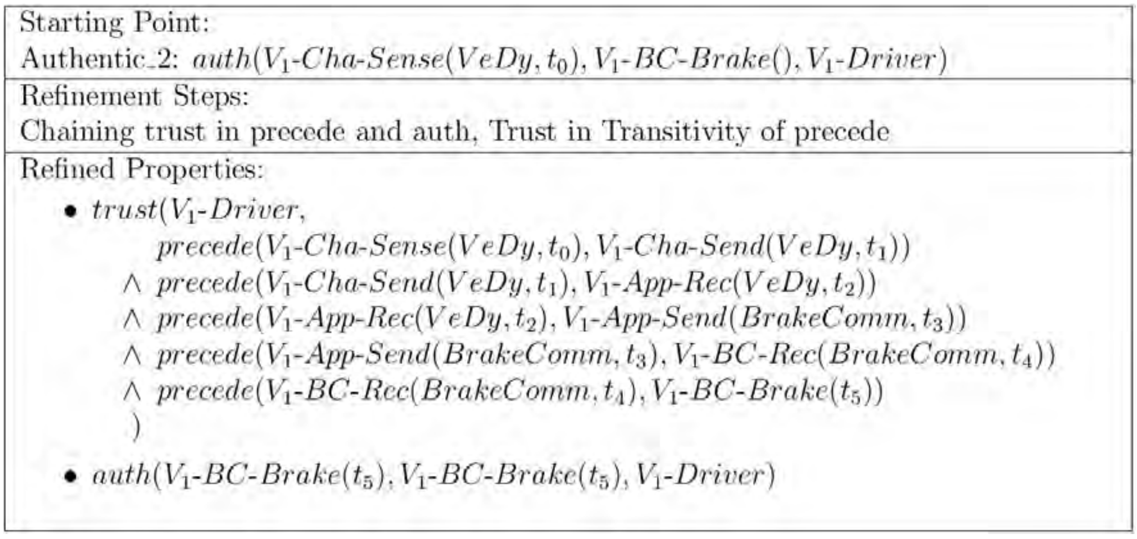
Starting Point:
Authentic_1: $auth(V_1\text{-}Env\text{-}Sense(EnvInfo, t_0), V_1\text{-}BC\text{-}Brake(), V_1\text{-}Driver)$

Refinement Steps:
Chaining trust in precede and auth, Trust in Transitivity of precede

Refined Properties:

- $trust(V_1\text{-}Driver,$
  $precede(V_1\text{-}Env\text{-}Sense(EnvInfo, t_0), V_1\text{-}Env\text{-}Send(EnvInfo, t_1))$
  $\wedge\ precede(V_1\text{-}Env\text{-}Send(EnvInfo, t_1), V_1\text{-}App\text{-}Rec(EnvInfo, t_2))$
  $\wedge\ precede(V_1\text{-}App\text{-}Rec(EnvInfo, t_2), V_1\text{-}App\text{-}Send(BrakeComm, t_3))$
  $\wedge\ precede(V_1\text{-}App\text{-}Send(BrakeComm, t_3), V_1\text{-}BC\text{-}Rec(BrakeComm, t_4))$
  $\wedge\ precede(V_1\text{-}BC\text{-}Rec(BrakeComm, t_4), V_1\text{-}BC\text{-}Brake(t_5))$
  $)$
- $auth(V_1\text{-}BC\text{-}Brake(t_5), V_1\text{-}BC\text{-}Brake(t_5), V_1\text{-}Driver)$

Evita’s Security Engineering Process III



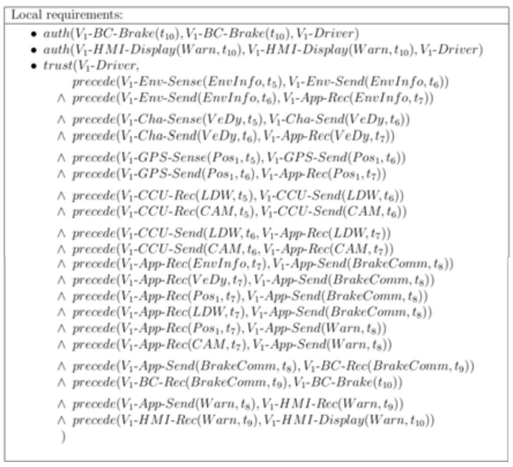
Starting Point:
Authentic_2: $auth(V_1\text{-}Cha\text{-}Sense(VeDy, t_0), V_1\text{-}BC\text{-}Brake(), V_1\text{-}Driver)$

Refinement Steps:
Chaining trust in precede and auth, Trust in Transitivity of precede

Refined Properties:

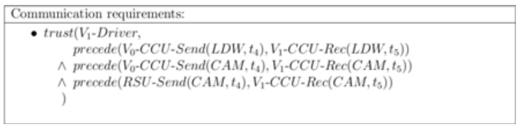
- $trust(V_1\text{-}Driver,$
  $precede(V_1\text{-}Cha\text{-}Sense(VeDy, t_0), V_1\text{-}Cha\text{-}Send(VeDy, t_1))$
  $\wedge\ precede(V_1\text{-}Cha\text{-}Send(VeDy, t_1), V_1\text{-}App\text{-}Rec(VeDy, t_2))$
  $\wedge\ precede(V_1\text{-}App\text{-}Rec(VeDy, t_2), V_1\text{-}App\text{-}Send(BrakeComm, t_3))$
  $\wedge\ precede(V_1\text{-}App\text{-}Send(BrakeComm, t_3), V_1\text{-}BC\text{-}Rec(BrakeComm, t_4))$
  $\wedge\ precede(V_1\text{-}BC\text{-}Rec(BrakeComm, t_4), V_1\text{-}BC\text{-}Brake(t_5))$
  $)$
- $auth(V_1\text{-}BC\text{-}Brake(t_5), V_1\text{-}BC\text{-}Brake(t_5), V_1\text{-}Driver)$

Evita’s Security Engineering Process IV



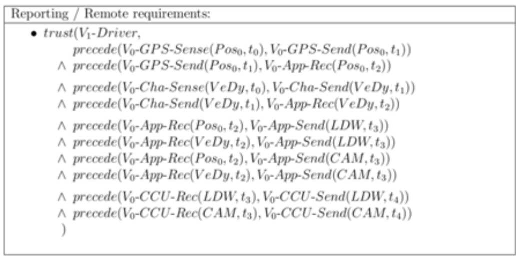
Local requirements:

- $auth(V_1\text{-}BC\text{-}Brake(t_{10}), V_1\text{-}BC\text{-}Brake(t_{10}), V_1\text{-}Driver)$
- $auth(V_1\text{-}HMI\text{-}Display(Warn, t_{10}), V_1\text{-}HMI\text{-}Display(Warn, t_{10}), V_1\text{-}Driver)$
- $trust(V_1\text{-}Driver,$
  - $precede(V_1\text{-}Env\text{-}Sense(EnvInfo, t_5), V_1\text{-}Env\text{-}Send(EnvInfo, t_6))$
  - $\wedge\ precede(V_1\text{-}Env\text{-}Send(EnvInfo, t_6), V_1\text{-}App\text{-}Rec(EnvInfo, t_7))$
  - $\wedge\ precede(V_1\text{-}Cha\text{-}Sense(VeDy, t_5), V_1\text{-}Cha\text{-}Send(VeDy, t_6))$
  - $\wedge\ precede(V_1\text{-}Cha\text{-}Send(VeDy, t_6), V_1\text{-}App\text{-}Rec(VeDy, t_7))$
  - $\wedge\ precede(V_1\text{-}GPS\text{-}Sense(Pos_1, t_5), V_1\text{-}GPS\text{-}Send(Pos_1, t_6))$
  - $\wedge\ precede(V_1\text{-}GPS\text{-}Send(Pos_1, t_6), V_1\text{-}App\text{-}Rec(Pos_1, t_7))$
  - $\wedge\ precede(V_1\text{-}CCU\text{-}Rec(LDW, t_5), V_1\text{-}CCU\text{-}Send(LDW, t_6))$
  - $\wedge\ precede(V_1\text{-}CCU\text{-}Rec(CAM, t_5), V_1\text{-}CCU\text{-}Send(CAM, t_6))$
  - $\wedge\ precede(V_1\text{-}CCU\text{-}Send(LDW, t_6, V_1\text{-}App\text{-}Rec(LDW, t_7))$
  - $\wedge\ precede(V_1\text{-}CCU\text{-}Send(CAM, t_6, V_1\text{-}App\text{-}Rec(CAM, t_7))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Rec(EnvInfo, t_7), V_1\text{-}App\text{-}Send(BrakeComm, t_8))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Rec(VeDy, t_7), V_1\text{-}App\text{-}Send(BrakeComm, t_8))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Rec(Pos_1, t_7), V_1\text{-}App\text{-}Send(BrakeComm, t_8))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Rec(LDW, t_7), V_1\text{-}App\text{-}Send(BrakeComm, t_8))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Rec(Pos_1, t_7), V_1\text{-}App\text{-}Send(Warn, t_8))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Rec(CAM, t_7), V_1\text{-}App\text{-}Send(Warn, t_8))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Send(BrakeComm, t_8), V_1\text{-}BC\text{-}Rec(BrakeComm, t_9))$
  - $\wedge\ precede(V_1\text{-}BC\text{-}Rec(BrakeComm, t_9), V_1\text{-}BC\text{-}Brake(t_{10}))$
  - $\wedge\ precede(V_1\text{-}App\text{-}Send(Warn, t_8), V_1\text{-}HMI\text{-}Rec(Warn, t_9))$
  - $\wedge\ precede(V_1\text{-}HMI\text{-}Rec(Warn, t_9), V_1\text{-}HMI\text{-}Display(Warn, t_{10}))$
  -)



Communication requirements:

- $trust(V_1\text{-}Driver,$
  - $precede(V_0\text{-}CCU\text{-}Send(LDW, t_4), V_1\text{-}CCU\text{-}Rec(LDW, t_5))$
  - $\wedge\ precede(V_0\text{-}CCU\text{-}Send(CAM, t_4), V_1\text{-}CCU\text{-}Rec(CAM, t_5))$
  - $\wedge\ precede(RSU\text{-}Send(CAM, t_4), V_1\text{-}CCU\text{-}Rec(CAM, t_5))$
  -)



Reporting / Remote requirements:

- $trust(V_1\text{-}Driver,$
  - $precede(V_0\text{-}GPS\text{-}Sense(Pos_0, t_0), V_0\text{-}GPS\text{-}Send(Pos_0, t_1))$
  - $\wedge\ precede(V_0\text{-}GPS\text{-}Send(Pos_0, t_1), V_0\text{-}App\text{-}Rec(Pos_0, t_2))$
  - $\wedge\ precede(V_0\text{-}Cha\text{-}Sense(VeDy, t_0), V_0\text{-}Cha\text{-}Send(VeDy, t_1))$
  - $\wedge\ precede(V_0\text{-}Cha\text{-}Send(VeDy, t_1), V_0\text{-}App\text{-}Rec(VeDy, t_2))$
  - $\wedge\ precede(V_0\text{-}App\text{-}Rec(Pos_0, t_2), V_0\text{-}App\text{-}Send(LDW, t_3))$
  - $\wedge\ precede(V_0\text{-}App\text{-}Rec(VeDy, t_2), V_0\text{-}App\text{-}Send(LDW, t_3))$
  - $\wedge\ precede(V_0\text{-}App\text{-}Rec(Pos_0, t_2), V_0\text{-}App\text{-}Send(CAM, t_3))$
  - $\wedge\ precede(V_0\text{-}App\text{-}Rec(VeDy, t_2), V_0\text{-}App\text{-}Send(CAM, t_3))$
  - $\wedge\ precede(V_0\text{-}CCU\text{-}Rec(LDW, t_3), V_0\text{-}CCU\text{-}Send(LDW, t_4))$
  - $\wedge\ precede(V_0\text{-}CCU\text{-}Rec(CAM, t_3), V_0\text{-}CCU\text{-}Send(CAM, t_4))$
  -)

Future Work

- Work on SeBB-based Security Engineering ongoing:
  - e.g. Paper at IFIPTM2010
  - Further publication pending
- Work on the topic of Security Engineering needs focus and good research:
  - e.g. Serenity Security Engineering Manifesto
  - Ontology-based approaches, Formal based approaches, UML-based approaches…
  - Security Engineering process; Grundschutzhandbuch, SQUARE, SREP, …
- Work on the topic of Pattern-based Security Engineering for embedded systems:
  - FP7-Project TERESA: Trusted Computing Engineering for Resource Constrained Embedded Systems Applications
    http://www.teresa-project.org