Secure Vehicle Communication

Deliverable 2.1
Security Architecture and Mechanisms for V2V/V2I

Project:	SeVeCom
Project Number:	IST-027795
Deliverable:	D2.1
Title:	Security Architecture and Mechanisms for V2V/V2I
Version:	v3.0
Confidentiality:	PP
Author:	Antonio Kung, editor (Trialog)
Date:	15 February 2008

Part of the Sixth Framework Programme
Funded by the EC - DG INFSO

Table of Contents

List of Figures

1 Document History

Version	Status	Date
v0.0	Initial draft (L. Bousis, Philips)	22/09/2006
v0.1	Modification of table of content on activities (L. Bousis, Philips)	27/09/2006
v0.2	Include security concepts (F.Kargl, U.Ulm) Relation with Frame (S.Cosenza, CRF) GST security architecture (A. Kung, Trialog)	25/10/06
v0.3	COMeSafety architecture (S.Cosenza, CRF) Identity and key management (P.Papadimitratos, EPFL) Secure communication (F.Kargl, U.Ulm) Tamper proof devices (T.Holczer, BUTE) Privacy (P.Papadimitratos, EPFL) Intrusion detection (F.Kargl, U.Ulm) Data consistency (T.Holczer, BUTE) Secure positioning (P.Papadimitratos, EPFL) Secure user interface (F.Kargl, U.Ulm).	1/12/06
V1.0	Document Integration (A.Kung, Trialog),	10/12/06
V1.3	Presentation of baseline architecture (A.Kung, Trialog),	7/6/07
V1.5	Restructuring of documents for V2.0 (A.Kung, Trialog)	10/7/07
V1.7	Complete abstract architecture. Update on implementation. (Contribution from all partners),	8/8/07
V1.8	Review before submission (Contribution from all partners)	15/8/07
V2.0	Final version for 2.0 (A.Kung, Trialog)	31/8/07
V2.1	Update on platform and baseline specification	15/12/07

Approval		
	Name	Date
Prepared	Antonio Kung	1 February 2008
Reviewed	All Project Partners	1 February 2008
Authorized	Antonio Kung	15 February 2008
Circulation		
Recipient	Date of submission	
Project partners	1 February 2008	
European Commission	15 February 2008	

2 Introduction

2.1 Intended Audience

This deliverable is an intermediate version of the final security architecture deliverable that will be made public as material for dissemination (D6.1). This intermediate version is intended for use within SeVeCom as well as for ITS projects and working groups (e.g. C2C consortium) with which SeVeCom has liaison activities. It reflects the current status of work concerning architecture.

2.2 Abbreviations and Conventions

CA:	Certificate Authority
CALM:	Continuous Air interface for Long and Medium distance
CRL:	Certificate Revocation List
DSRC:	Digital Short Range Communication
DMV:	Department of Motor Vehicles
ECU:	Electronic Control Unit
GPS:	Global Positioning System
IVC:	Inter-Vehicular communication (equal to V2V + V2I)
ITS:	Intelligent Transport System
PKI:	Public Key Infrastructure
OBU:	Onboard Unit
QoS:	Quality of Service
RSI:	Roadside Infrastructure
RSU:	Roadside Unit
R2V:	Roadside to Vehicle
TOC:	Transportation Operation Centre
TCU:	Telematics Control Unit
TTL:	Time To Live
TESM:	Tamper Evident Security Module
VANET:	Vehicle Adhoc Network
V2V:	Vehicle to Vehicle communication
V2I:	Vehicle to Infrastructure communication
VC:	Vehicular Communication
VIN:	Vehicle Identification Number
VSCC:	Vehicle Safety Communication Consortium

2.3 Scope and Objectives of SeVeCom

SeVeCom addresses security of future vehicle communication networks, including both the security and privacy of inter-vehicular and vehicle-infrastructure communication. Its objective is to define the security architecture of such networks, as well as to propose a roadmap for progressive deployment of security functions in these networks.

Vehicle to Vehicle communication (V2V) and Vehicle to Infrastructure communication (V2I) bring the promise of improved road safety and optimised road traffic through co-operative systems applications. To this end a number of initiatives have been launched, such as the Car-2-Car consortium in Europe, and the DSRC in North America. A prerequisite for the successful deployment of vehicular communications is to make them secure. For example, it is essential to make sure that life-critical information cannot be modified by an attacker; it should also protect as far as possible the privacy of the drivers and passengers. The specific operational environment (moving vehicles, sporadic connectivity, etc.) makes the problem very novel and challenging.

Because of the challenges, a research and development roadmap is needed. We consider SeVeCom to be the first phase of a longer term undertaking. In this first phase, we aim to define a consistent and future-proof solution to the problem of V2V/V2I security.

SeVeCom will focus on communications specific to road traffic. This includes messages related to traffic information, anonymous safety-related messages, and liability-related messages. The following research and innovation work is foreseen:

- Identification of the variety of threats: attacker's model and potential vulnerabilities; in particular, study of attacks against the radio channel and transferred data, but also against the vehicle itself through internal attacks, e.g., against TCU (Telematics Control Unit), ECU (Electronic Control Unit) and the internal control bus.

- Specification of architecture and of security mechanisms which provide the right level of protection. It will address issues such as the apparent contradiction between liability and privacy, or the extent to which a vehicle can check the consistency of claims made by other vehicles. The following topics will be fully addressed: key and identity management, secure communication protocols (including secure routing), tamper proof device and decision on crypto-system, and privacy. The following topics will be investigated in preparation of further work: in-vehicle intrusion detection, malfunction detection and data consistency, secure positioning, and secure user interface.

The definition of cryptographic primitives will take into account the specific operational environment. The challenge is to address (1) the variety of threats, (2) the sporadic connectivity created by moving vehicles and the resulting real-time constraints, and (3) the low-cost requirements of embedded systems in vehicles. These primitives will be adaptations of existing cryptosystems to the V2V/V2I environment.

The overall approach is the following:

- Take into account existing results available from on-going eSafety projects in terms of threat analysis and security architecture.
- Work in close liaison with new IST eSafety projects which will focus on C2C application and road network infrastructures. Common workshops will be held in order to reach a consensus on the security threats and the proposed mechanisms.
- Take into account on-going standardisation work at the level of security such as ISO15764 - Extended Data Link Security, or ISO/CD20828 - Security Certificate Management, or at the level of communication ISO2121x series on CALM - Continuous Air interface for Long and Medium distance.
- Integrate SeVeCom mechanisms into a use case development which is based on the V2V/V2I infrastructure used by eSafety projects.
- Investigate the necessary conditions for deployment. This includes the provision guidelines for security evaluation and certification, as well as the definition of a roadmap. This will include discussion on organisational issues (e.g. key and certificate management).

  The project will work in close liaison with the Car-2-Car Communication Consortium; it will also establish strong connections with related efforts elsewhere in the world, notably USA (DSRC, IEEE P1609) and Japan.

  SeVeCom covers a number of research topics. The table below lists them along with the expected achievement.

	Topic	Scope of work	Academic Partners (first name is leader)
A1	Key and identity management	Fully addressed in SeVeCom	EPFL, BUTE
A2	Secure communication protocols (including secure routing)	Fully addressed in SeVeCom	U.Ulm, BUTE
A3	Tamper proof device and decision on cryptosystems	Fully addressed in SeVeCom	BUTE, KUL
A4	In-Vehicle Intrusion Detection	Investigation work	DC, Bosch
A5	Malfunction Detection and Data consistency	Investigation work	BUTE, U.Ulm
A6	Privacy	Fully addressed in SeVeCom	EPFL, U.Ulm, BUTE, KUL
A7	Secure positioning	Investigation work	EPFL
A8	Secure user interface	Investigation work	U.Ulm

2.4 Scope and Objectives of Document

This document specifies the Inter Vehicular Communication (IVC) security architecture in a technology independent manner. This specification takes into account the specific aspects of an IVC environment,

the limitations on available resources, and the concerns (e.g., privacy and cost) of the future users of this technology. Additionally, an overview of existing in-vehicle protection mechanisms will be carried out and an assessment of their maturity will be made.

The following versions of D2.1 are intended as described in SeVeCom technical annex.

- D2.1v1.0 (December 2006)
  This document includes
  - an initial architecture covering A1 to A8
  - an initial analysis of mechanisms covering A1, A2, A3, A6.
- D2.1v2.0 (June 2007)
  This document includes
  - a final architecture covering A1 to A8
  - a final analysis of mechanisms covering A1, A2, A3, A6
  - an initial specification of mechanisms covering A1, A2, A3, A6.
- D2.1v3.0 (December 2007)
  The document includes
  - a final architecture covering A1 to A8
  - a final analysis of mechanisms covering A1, A2, A3, A6
  - a final specification of mechanisms covering A1, A2, A3, A6
  - an initial investigation of topics A4, A5, A7, A8
- D2.1v4.0 (June 2008)
  The document includes
  - a final architecture covering A1 to A8
  - a final analysis of mechanisms covering A1, A2, A3, A6
  - a final specification of mechanisms covering A1, A2, A3, A6
  - a final investigation of topics A4, A5, A7, A8

D2.1v1.0 included

- an architecture analysis part explaining
  - the SeVeCom specification approach,
  - the relationship with Frame,
  - the relationship with COMeSafety,
  - the relationship with GST SEC security architecture
  - an initial analysis of mechanisms covering priority research area
  - an initial analysis of mechanisms covering longer term priority research area

D2.1v2.0 includes

- a deliverable context part (section 3) which explains
  - the SeVeCom analysis approach
  - the relationship with other initiatives (Frame, COMeSafety, GST SEC)
- An analysis section integrating a solution analysis and the individual analysis carried out for each priority research area (section 4.3)
  - Analysis of principles for a solution design
  - Analysis of security mechanisms for a solution design
  - A1 Key and identity management
  - A2 Secure communication protocols
  - A3 Tamper proof device and decision on cryptosystem
  - A6 Privacy
- An analysis section of mechanisms covering longer term priority research area (section 4.4)
  - A4 In-vehicle Intrusion detection
  - A5 Malfunction Detection and Data consistency
  - A7 Secure positioning
  - A8 Secure user interface
- The specification of the SeVeCom baseline architecture (section 5)
  - SeVeCom architecture principles
  - SeVeCom abstract architecture from a conceptual and deployment view, as well as partial integration and administration view
- Current directions concerning SeVeCom architecture proof-of-concept implementation (section 6).

3 Context of this Deliverable

3.1 SeVeCom Architecture Specification Approach

In the Deliverable D1.1 "VANETS Security Requirements" the SeVeCom members identified requirements that are relevant for securing a multitude of applications in the area of vehicle communications. Based on this analysis, we were able to determine 23 security mechanisms that are both suitable and necessary to secure IVC and conquer the identified threats and attacks.

In this section we will briefly summarise this process and review the security mechanisms, as they will provide the building functional blocks of our architecture. The complete details can be found in the SeVeCom Deliverable 1.1.

3.1.1 Requirements

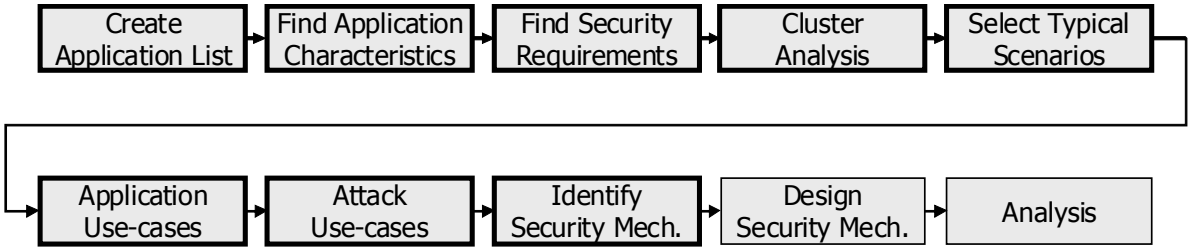

Figure 3-1: Requirements Analysis Process

Figure 3-1 gives an overview over the process we used to finally identify the necessary security mechanisms. First we compiled a list of applications that are relevant in the IVC context. This list was compiled from various sources to ensure that the view of related projects is covered well.

Step two was necessary to get a more detailed understanding of these applications. As existing sources most often gave only a name and a short application description or as different sources described applications to be realized in different ways – e.g. with or without infrastructure involved – we needed a consistent viewpoint on basic characteristics of the applications. This included properties like addressing (uni-, multi-, geocast), vehicle-to-vehicle vs. vehicle-to-infrastructure communication or typical maximum latencies required by the applications.

As the application list was too long to perform a detailed threat and security requirements analysis for all the applications, we decided to select applications with "typical" characteristics and security requirements for further detailed analysis. Therefore, we did a rough analysis of relevant security needs of the applications, determining e.g. whether the application will need different forms authentication, privacy requirements, etc.

Both application characteristics and security requirements were expressed in numerical values so that a statistical cluster analysis was then able to identify clusters of applications with similar characteristics and security requirements. We identified 8 such cluster and selected the following 10 representative applications:

- *SOS services*
- *Stolen vehicles tracking*
- *Map download/update*
- *Intersection collision warning*
- *Vehicle-based road condition warning*
- *Electronic license plate*
- *Road surface conditions to TOC*
- *Software update/flashing*
- *Emergency vehicle signal pre-emption*
- *Work zone warning*

In a next step, these applications were described in use cases in enough details to later do a substantial threat analysis. The use cases contain detailed descriptions of the application scenario and operation. The whole description was, however, done without any security or protection mechanisms.

The attack use cases in the following step now identify various ways how these applications may be attacked in order to e.g. prevent the proper operation of the application, to invade the privacy of users involved, or to alter the system for the own profit. With about 3 attack use cases per application, we identified a representative set of nearly 30 attacks in total.

Finally, based on the gained knowledge regarding the applications and potential attacks, we derived 23 different security mechanisms that when properly designed and deployed will prevent all of the attacks identified.

For now, these mechanisms are pure concepts and it will be the task of our further research to find suitable solutions how to implement them. In the next section, we will first describe the different concepts to highlight what mechanisms our security architecture must include.

3.1.2 Security Mechanisms/Concepts

Identification & Authentication Concepts

The first block of security concepts deals with identity authentication. In contrast to the classical understanding, where authentication means entity authentication, in VANETs we require different forms of authentication, e.g. authentication of vehicle positions or various other attributes. The individual mechanisms are

- *Identification* provides vehicles with unique and unforgeable identities.
- *Authentication of sender* allows the receiver in a communication process to reliably verify the identity of a sender.
- *Authentication of receiver* allows the sender in a communication process to reliably verify the identity of a receiver before actually sending the data.
- *Attribute authentication* allows the sender or receiver in a communication process to reliably verify certain properties of a communication partner without necessarily revealing its identity.
- *Authentication of intermediate nodes* allows two communication partners to reliably verify the identity or certain properties of intermediate nodes in a routing process.

Privacy Concepts

In the requirements engineering process we clearly identified a strong need for privacy-preserving mechanisms. Without that, location tracking and other forms of privacy invasions may seriously damage the adoption of IVC systems. Therefore, an IVC architecture must support the following mechanisms

- *Total anonymity* where a participant in an IVC system remains completely anonymous, i.e. no information that could identify that participant can be gained by other parties.
- *Resolvable anonymity* is the same as total anonymity with the exception that under certain, well-defined circumstances others may be able to identify the otherwise anonymous entity.
- *Location obfuscation* gives an entity the opportunity to report its location with an adjustable precision. Depending on the application and privacy requirements, vehicles may e.g. introduce random inaccuracies in the reported positions. This impedes location profiling.

Integrity Concepts

Ensuring the integrity of communicated information is of vital importance e.g. for all eSafety applications. Modification attacks may render warning information useless or even lead to accidents provoked by the attacker. As communication happens ad-hoc between arbitrary communication partners, not all information can be protected by the classical cryptographic means. Therefore we provide various different means of integrity protection.

- *Integrity protection* ensures integrity in the traditional way, e.g. by using digital signatures or message authentication codes.
- *Encryption* primarily ensures confidentiality. Data is encrypted using either symmetric or asymmetric cryptography.
- *Detection of protocol violation* is based on a model of the communication protocol. Any deviation from this protocol can be seen as an intrusion attempt or an integrity violation.
- *Consistency/context checking* uses sensors or heuristics to conduct consistency checks on received data or align received data with the environment/context. If received information fails this test, the data may be dropped altogether or be rated with lower reliability.
- *Attestation of sensor data* is based on the idea of having tamper-resistant and trustworthy sensors that certify the credibility of the transferred data.
- *Location verification* provides a way to analyze the correctness of position information transmitted by other vehicles.

- *Tamper-resistant communication system* introduces tamper-evident hardware and software in the architecture to protect the integrity of key material or other vital information.
- *Digital Rights Management (DRM)* provides ways of ensuring the integrity of digital data, especially software downloaded into the car. Furthermore this may also be used to manage usage rights for certain software, map data, etc., so it also serves as an access control mechanism.
- *Replay protection* ensures the freshness of communicated data and prevents replay attacks.
- *Jamming/DoS protection* prevents jamming attacks by making the radio interface more reliable and attack-resistant. Different other forms of Overload DoS attacks should also be prevented.

Access Control/Authorization Concepts

Controlling who can participate in various aspects of IVC and who can access different parts of the in-vehicle systems will provide additional security.

- *Access control* controls, which nodes in a VANET can participate in different forms of applications.
- *Closed user groups* is a mechanism that supports the creation of closed user groups, e.g. for inter-vehicle messaging. Doing that, outsider attacks are generally prevented.
- *Firewall/Checkpoint* is a component that controls, what part of the in-vehicle systems are accessible by foreign nodes. This creates a single point of control, where parts of a security policy can be enforced.
- *Sandbox* is a component where software from unreliable sources can be executed, monitored and controlled to prevent malware from overtaking or compromising the system.
- *Filtering* e.g. at intermediate nodes can prevent illegitimate information from flooding the network leading to a denial of service attack.

3.1.3 Architecture specification approach

This section explains the design process of the SeVeCom security architecture, which is a continuation of the threats analysis and security requirements process. Here, we describe the structure of the architecture design process and how it relates to the earlier work done in SeVeCom.

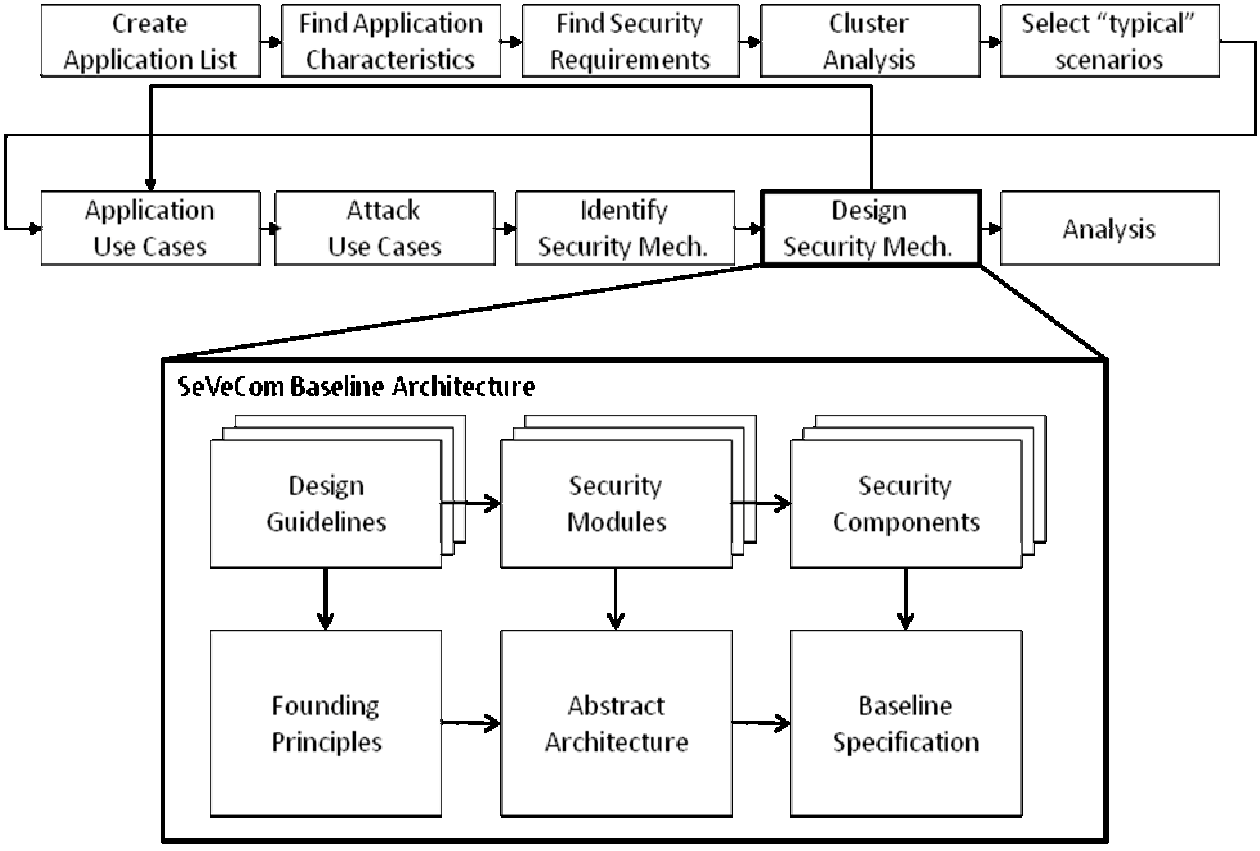

Figure 3-2: Architecture specification approach

As shown in Figure 3-2, the requirements engineering process is followed by the compilation of security mechanisms. SeVeCom contains three different kinds of architectures. The process starts with a list of generic ***Design Guidelines*** that are directly derived from the previous analysis process. These include fundamental decisions like the applications of pseudonyms for privacy protection or the application of PKI-based structures for trust management. These guidelines will be summarized by the ***Founding Principles***, which represent the basis of all further architecture and design work.

Drawn from the generic design guidelines the SeVeCom architecture consists of several ***Security Modules*** that each addresses a specific area of security. The current version of the architecture contains the following modules:

- *Identification & Trust Management Module*
- *Privacy Management Module*
- *Secure Communication Module*
- *Tamper-Evident Security Module*
- *In-car Security Module*

As one can see, the individual modules correspond to the different research activities within SeVeCom. Together with the generic framework these modules form the ***Abstract Architecture*** that is described in detail in this document. Additionally, the abstract architecture contains mechanisms for organization and configuration of modules and for attachment of the SeVeCom security system to the other in-vehicle or in-RSU systems.

When it comes to concrete instantiation of the SeVeCom security architecture, each module is again split into distinct ***Security Components***. These components partially correspond to and implement the security mechanisms identified during the requirements analysis process. "Partially" because sometimes multiple mechanisms are combined in one component in order to prevent unnecessary fragmentation of the architecture. Examples of security components include *Identification Management, Trust Management, Secure Beaconing, Secure Geocast, Secure Georouting, Pseudonym Management and Application, Key/Certificate Storage, Secure Time Base, Intrusion Detection,* etc..

Altogether the first instantiation of the abstract architecture and their components form the ***Baseline Specification***. As these components may be implemented in many different ways, providing different levels of assurance and security, generating different levels of overhead, etc., they are not part of the abstract architecture anymore but represent a specific instance of a SeVeCom security system. While conforming to the architecture, this gives the necessary flexibility to adapt the abstract architecture to the specific system and market requirements of implementing or standardization bodies that will build on the SeVeCom results.

The rest of this document describes both the founding principles as well as the abstract architecture and gives a brief overview on the initial baseline specification.

3.2 Relationship with other IST Initiatives

3.2.1 Frame

Frame (FRamework Architecture Made for Europe) purpose was to create a stable Framework for the development and deployment of working and workable ITS within the European Union. The result is a "tool-box" from which other ITS Architectures and/or systems specifications can be developed. While Frame is a completed project, a forum has been created to promote and coordinate the use of the "tool-box". More can be found on Frame on the following web site:

http://www.Frame-online.net

This section compares Frame architecture work viewpoint to SeVeCom architecture work.

3.2.1.1 Frame Architecture Process

The Frame architecture process is depicted in Figure 3-3 (note that the part of the process which is not explained is covered by a cloud). It involves the following elements:

- *User needs*, which are expressed as services or functions. User needs are categorized according categorized into groups (General, Management Activities, Policing/Enforcing, Financial Transaction, Emergency Services, Travel Information, Traffic Management, In-Vehicle Systems, Freight and Fleet Operations, Public Transport, etc.).
- *Models* These represent "real World" situations and in some cases may represent scenarios of how the functionality to develop should be used. The combination of the *Models*, real life situations, and the *user needs* list allows for the identification of the effective functions of an application.

- The *System Context.* It defines how the outside World will relate to the Systems. This relationship is defined through the *terminators*. These components describe what data the Systems will expect the outside world to provide and what it must do with the data coming from the Systems. A *terminator* may be a human entity, a system or a physical entity from which the data can be obtained (road surface, atmosphere). Here is a list of examples : bridge/tunnel, weather system, traveller, traffic, emergency system, road related system, driver, vehicle, law enforcement agency, operator, freight equipment.

All the elements described above, the *terminators* and the *user needs* especially, are then used to develop the ***functional architecture***.

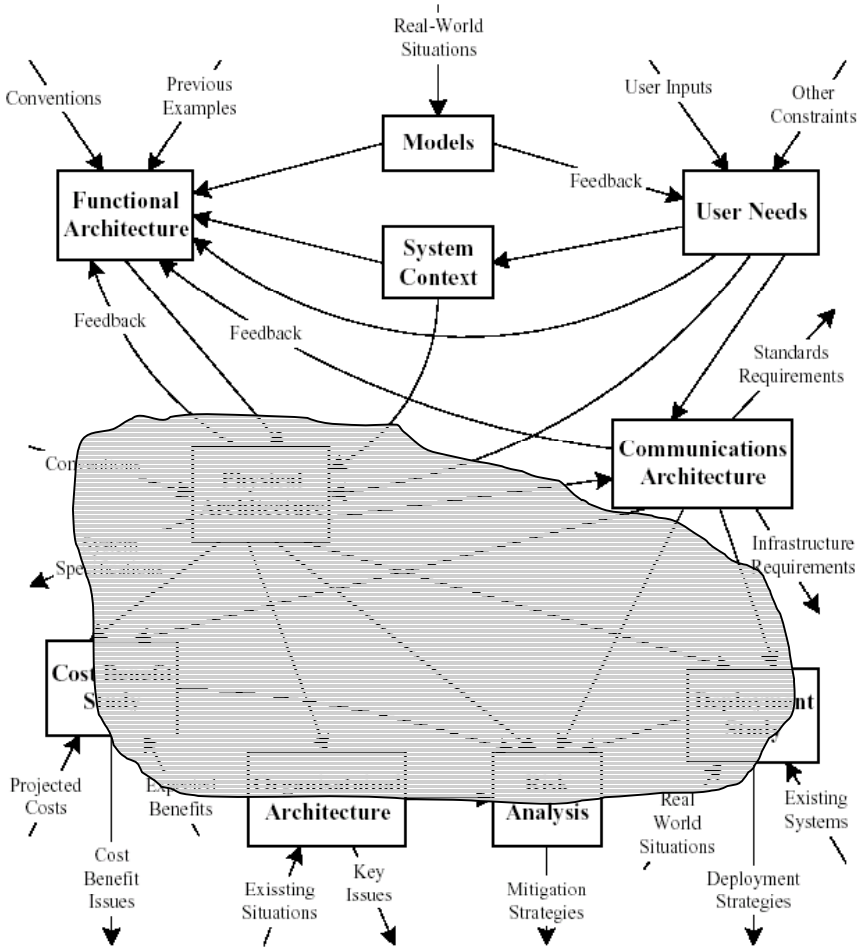

Figure 3-3: Frame Process

3.2.1.2 Discussion

Both SeVeCom and Frame have defined an architecture design process. Frame architecture design process is user-need driven, and appropriate to allow for the definition of a functional architecture. SeVeCom architecture design process is driven by security needs and appropriate for the definition of a technical architecture for security of vehicular communication.

The link between the two architectures is as follows:

- For each category of ITS application, Frame considers a list of general performance, quality requirements, and constraints. Security is part of this list. Therefore SeVeCom contributes to enrich Frame specification.
- SeVeCom addresses specific functional aspects such as privacy and policing/enforcing which are part of the functional approach.

3.2.2 GST SEC Security Architecture

This section explains the relationship of SeVeCom security architecture with GST Sec security architecture, detailed in the following document:

> GST SEC Architecture and Interface Specifications for the SEC sub-project in GST, DEL_SEC_3_1, September 2005. Stephan Eichler (Technical University of Munich), Jérôme Billion (Trialog).
>
> http://www.gstproject.org/sec/docs/DEL_SEC_3_1_Architecture_and_interface_specifications_v1.0.pdf

GST SEC is a subproject of GST the objective of which was to specify and demonstrate an overall open solution for telematics applications. The GST architecture work is consequently focusing on the distributed entities making up an eSafety infrastructure (e.g. service centre, control centre, vehicle). Within GST, the GST SEC architecture focused on the security part of it, involving security interactions (i.e. called collaborations in GST architecture terminology) such as *Distributed authorisations, Authentication services*, *Secure communication* and *Trusted execution platforms.*

3.2.2.1 GST SEC Secure Communication Architecture

GST SEC defines a secure communication architecture based on the combination of *security modules* and *secure communication tunnels*. The *security module* is the component that makes a node security compliant. In terms of implementation, a security module may consist of hardware and/or software elements, or a combination of hardware and software. The tamper evident component of the Sevecom architecture can be considered as an instantiation of the security module for secure communication to take place between two nodes, a *secure communication tunnel* is established between the security module of one node, and the security module of the other node.

The following diagram shows a client node that sets up a secure communications tunnel with a server node. The security module of the respective nodes deal with the confidentiality and authentication of the information sent through the communications tunnel.

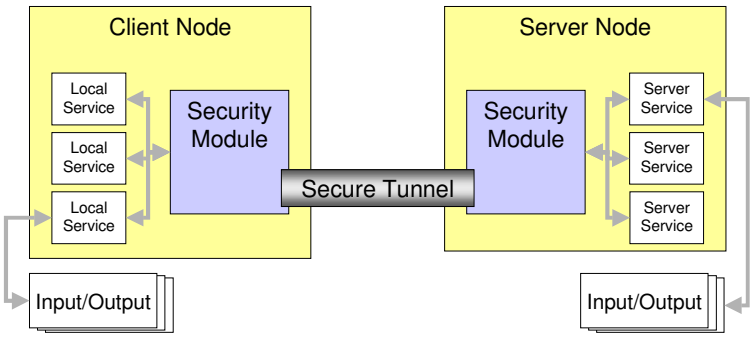

Figure 3-4: Secure Tunnel

Establishing the communications tunnel is based on an abstract secure communication protocol based on (1) a key agreement protocol and (2) secure communication. The key agreement protocol allows two entities A and B to anonymously establish a shared key with mutual entity authentication and mutual explicit key authentication. Assuming that two nodes *A* and *B* need to establish a security context, the key agreement protocol is illustrated below. The Ping and Pong messages implement the key agreement protocol.

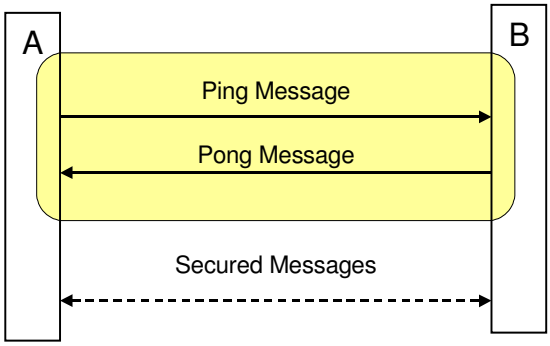

Figure 3-5: The basic key agreement messages exchange

Having securely established a shared secret, entities may exchange secured messages according to the security settings established during the key agreement part: *Insecure, Authenticated, Confidential,* and *Secure*.

3.2.2.2 Discussion

The GST SEC architecture in terms of secure communication is in the scope of SeVeCom. The key agreement protocol used in the GST SEC architecture is optimized to minimize the security overhead and considers all underlying cryptographic mechanisms as building blocks. This means that the selection of the exact cryptographic algorithm does not influence the protocol used, e.g., when instantiated in a car-to-car communication, or in a car-to-infrastructure communication, the first instantiation could be based on an Elliptic Curve Cryptosystem, where the second could rely on more traditional systems such as the RSA or Diffie-Hellman. Note, however, that GST SEC did not explicitely focus on car-to-car communication, nor on privacy, the GST SEC architecture (i.e. abstract secure communication) therefore the GST-SEC secure communications protocol needs to be instantiated in a privacy-focusing variant to cope with the communication patterns of SeVeCom .This instantiation does not change the basic protocol, as the underlying key agreement protocol (which is the Station-to-Station protocol) supports a mode in which the secret keys are agreed in an anonymous manner so that one of the communicating parties identifies itself to the other, or they mutually identify each other, without a third party being able to identify either party.

3.2.3 COMeSafety

COMeSafety is an on-going project to coordinate on-going eSafety projects. One of the coordination points is a common communication architecture. COMeSafety analysis is as follows

- A communication architecture involves a description of two main elements : (1) the communication systems functions integrated in entities of an eSafety infrastructure (e.g. control centre, vehicle, …), and (2) a description of the communication links in terms of used communication technologies as well as supported communication and message formats.
- A common such architecture involves agreement on a terminology and architecture design patterns. Agreeing on architecture design patterns involves an architecture process.

COMeSafety proposes to rely on approaches elaborated in Frame ITS and IEEE Std 1471 mainly. The Frame ITS approach ensures a common overall functional vision. IEEE1471 provides guidelines to specify architectural description (AD). Figure 3-6 sketches its conceptual approach.

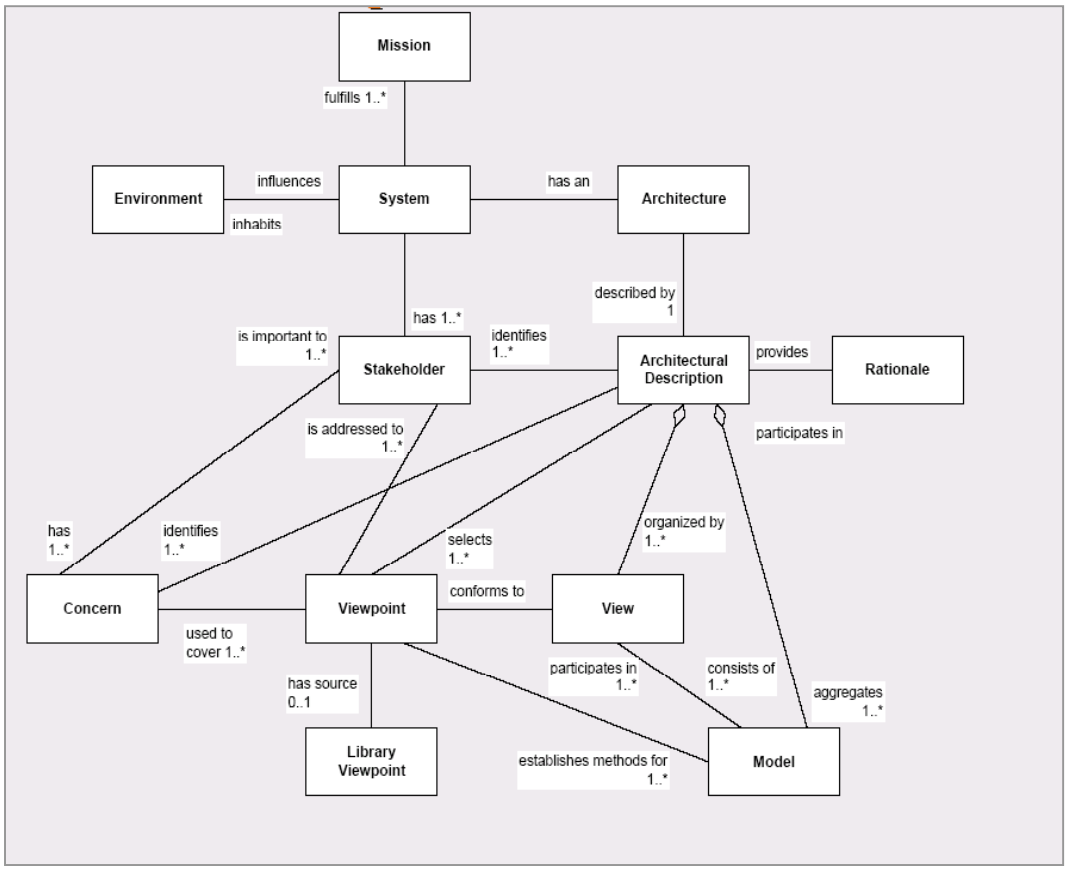

Figure 3-6: IEEE 1471 Conceptual Framework

3.2.3.1 Discussion

- COMeSafety is co-ordinating work related to communication architecture. It has considered the FRAME architecture design process, as well as the communication architecture resulting from existing eSafety projects (such as GST).
- COMeSafety is considering FRAME as well as IEEE 1471 to define an overarching common architecture design process that would lead to a communication architecture that is "compatible" with FRAME and common to the eSafety community

SeVeCom intends to contribute and liaise with COMeSafety architecture work from the security point of view.

4 Analysis

4.1 Principles for a Solution Design

This section provides a set of principles that will guide the definition of a SeVeCom future proof solution. These principles are also defined and explained in [37]. Identifying these principles was motivated by the state of the art literature available at the beginning of this project. A small number of works were concerned with different aspects of security and privacy of vehicular networks, either outlining challenges, describing particular attacks or more general attack overviews, offering general suggestions towards solutions, or proposing mechanisms. The identification of a set of design principles provides a solid basis for the development of future vehicular security schemes. As VC is a technology in the making, our investigation draws from the current understanding and projections from both the academic and industry worlds. At the same time, we point out the unique or novel aspects due to VC salient characteristics.

Moreover, many proposals to address security for vehicular communications are expected to be devised. Nevertheless, security mechanisms and protocols to safeguard the system operation and thwart adversarial behavior will in general differ in functionality. Based on our investigation on operational assumptions, the system, communication, and adversary models in [37], as well as experience from other related networking paradigms and the to-date development of the vehicular communications technology, the set of design principles outlined next can serve the development of future security solutions.

Principle	Description
Default Network Access	Messages, especially those that are broadcast (e.g., safety, driver assistance) are by default accessible to all nodes that can receive them. Similarly, nodes are by default assumed to assist multi-hop communication. Furthermore, possession of keys and valid credentials is the basic prerequisite for nodes to transmit messages.
Locality and Timeliness as Privileges	As vehicular networks and the supported applications are context-aware, only the vicinity of a node to a location or an action to a point in time may enable specific action. Examples are the generation or validation of a specific message or a credential, the request and access to a service, or the participation to a distributed protocol execution.
Visibility of Events	An attestation of an event by an individual node requires that the event be visible to the attesting node; either the node is the sole responsible for generating the event (e.g., alarm), or it had the locality and timeliness privilege (e.g., reception of the message within δ seconds from its generation) to provide the attestation. More generally, events that trigger joint computations or actions by multiple nodes should have been visible to all nodes participating in the distributed protocol. The definition of visibility, with respect to locality and timeliness, as well as other node attributes, such types of on-board sensory inputs, is application specific. Note also that relaying event reports originating from other vehicles does not imply visibility of the reported event by the relaying node, but rather compliance with the data dissemination protocol. Furthermore, the definition of what constitutes an event, with the appropriate granularity and expressiveness, that is, with environment, network, or application events, can reflect a wide range of settings. Consider for example a collection of multiple messages/reports of an environmental event (e.g., traffic jam) by a node; should this result to adequate trust level on the onset of a jam as evaluated by the node itself, the node can report an application event related to the aforementioned inference even if the effects of the traffic jam are not (environmentally) visible to it (e.g., via measurements of own velocity and that of nearby vehicles, density of vehicles, etc.)
Mandated (non-circumventable) Mediation	All actions that change the security state of the network (e.g., assignment of identities or distribution of cryptographic keys and

Principle	Description
	credentials), are mediated by a network authority. Network authority actions cannot be circumvented by any of the network nodes. For example, no coalition of nodes can substitute an authority and issue new or revoke valid credentials
Accountability	All messages or protocol executions that affect the network operation, or at least a critical subset of the operation, and in particular the participation of other nodes in the network, are auditable by the authorities. For example, alarm messages that notify of system malfunctions must be auditable
Vehicle Autonomy	Node actions and operations that do not require mediation are autonomous with respect to those of other nodes. For example, all nodes are able to reject messages from other nodes, or utilize solely their local input for any computation. We note that autonomy does not however mean freedom in participating in the protocol execution, e.g., avoid relaying messages
Separation of Privilege	Reliance on multiple authorities, as well as the separation of the roles of authorities and infrastructure, and thus the distribution of trust, can provide increased security, privacy, and fault-tolerance
Non-Frameability	Non-frameability implies that a trusted entity (node) cannot perform actions or more generally prove that a node x performed the action, if x never did so. This principle is based on our operational assumption of authorities that nonetheless should not be all-powerful. For example, use of a cryptosystem that allows the authority to sign on-behalf of the registered entity (vehicle) would not satisfy this principle
Privacy Conservation	Vehicular communications should not become a weak link in terms of privacy, providing users at least with the same level of privacy that is currently afforded without vehicular networks. The privacy of a driver should be protected against private citizens and law enforcement agencies. Conservation is meaningful in the latter case as well, whereas privacy should be conditional to specific scenarios (liability).
Usability	Usability calls for ease of the user to access and utilize the vehicular communication system, as well as to utilize the information it provides. Psychological acceptability is one important aspect. Simplicity and reliability of management operations (e.g., maintenance, or refreshing of credentials) is also crucial; consider, for example, the clearly unwanted situation that a vehicle computer would not allow its engine to start because a necessary 'fresh' credential cannot be obtained.
Staged Response to Faulty Behavior	This principle calls for a system design with a multilevel, escalating response to faults. More specifically, a low assurance detection can be followed by a warning, then self-constrained participation, a probation period, and local containment following a report, and finally eviction from the system. At all stages, reinstatement should be possible. The cause behind such a principle is the difficulty in distinguishing among benign and malicious faults due to the network volatility, and the frequent non-critical nature of the majority of faults. The bounded adversary presence is related, as it renders the staged response a reasonable approach. Which can also be assisted by the inherent redundancies in the protocol mechanisms. Moreover, actions at different stages can be in accord with the principle of vehicle autonomy. Finally, eviction clearly assists towards bounded adversarial presence.
Reconfigurability	Reconfigurability is a principle that applies to the on-board software and firmware (e.g., automatic download of patches), as well as the policies that describe what services each node provide to its peers. More generally, reconfigurability of services provided by the infrastructure nodes (registration/deregistration), as well as flexible service discovery (and delivery). Open interfaces to the network and security services. Beyond its obvious practical aspects, reconfigurability can be viewed as the means to ensure the bounded adversarial presence. In that sense, it is related to the response to faulty behavior, and be viewed as an additional means to ensure bounded adversarial presence. For example,

Principle	Description
	applying software patches and updating virus definition files can prevent the exploit of software vulnerabilities and the spread of malicious software across the VC system.

4.2 Security Mechanisms in a Solution Design

This section reviews each security mechanisms determined in section 3.1.2 from a solution design point of view.

Identification

Identification is a very basic security concept forming a base for almost all other security mechanisms. It is discussed in detail in section 4.3.1. Communication protocols are involved in so far as cars have to exchange credentials in the course of the identification process and have to contact or rely on trusted third parties / certification authorities in order to manage their credentials. This exchange will have to be secured, i.e. integrity and/or confidentiality protected, protected from replay attacks, etc. There are however separate mechanisms for dealing with these issues.

Authentication of data origin (i.e., original sender)

Authentication of data packets will play a vital role in IVC. For many applications, data sources must be authenticated. However, there are some major problems coming from the application characteristics in IVC which prevent some standard solutions from being applicable in all cases:

1. *Communication usually is one-way*: many data transmissions will not have an answer being sent back to the originator. So any kind of bi-directional authentication protocol will not work for these applications. Instead, the authentication of the sender must be possible directly when receiving a packet. Note, however, that this usually implies that it is very hard to protect against replaying recorded data transmissions.
2. *Communication is time critical*: many eSafety applications require periodic transmission of packets at a high-rate. So classic public-key cryptosystems like RSA might be too heavy-weight to generate signatures at that rate for sender authentication. Faster and more light-weight solutions like Elliptic Curve-based cryptosystems (ECC) may need to be considered, as ECC systems typically meet these requirements: extremely fast signature generation, and short signature sizes (compared to RSA or DSA signatures).
3. *Privacy is relevant*: as discussed in Sections 4.3.1 and 4.3.4[FK1], privacy is of great importance. This will prevent to use standard entity authentication mechanisms directly, as this would potentially reveal the identity of vehicles or their drivers, or even passengers. Instead, pseudonyms will have to be used which might have an influence on the authentication protocol.
4. *No online connection*: in vehicular ad-hoc networks one usually cannot assume an online connection to any kind of central trusted third party, certification authority, or whatever. For this reason, there is a clear need for a solution that supports sufficient guarantees in an offline situation.
5. *No direct packet forwarding*: Some applications do not simply forward packets in a multi-hop fashion, but instead will use data dissemination, where data is broadcast, received by neighbors, processed, and finally new data packets are broadcast again. It needs to be considered if a way can be found to authenticate the parties that contributed to the transmitted information in any way.
6. *Packet size*: some signature schemes like RSA will significantly increase the packet size. As it is expected that in particular eSafety applications will emit a high number of small packets, this creates a significant problem, which may also be solved using ECC-based solutions.

These problems need to be respected when designing one or multiple appropriate authentication protocol(s). Of course, the authentication will have to based be based on the identity management system described in section 4.3.1.

Authentication of receiver

Although our requirements analysis has shown that the authentication of a receiver is not as relevant as sender authentication, it is necessary in some cases. Basically, the same problems apply as discussed in the previous section (with the exception of problem 5, *No direct packet forwarding*). One-way receiver authentication may be achieved by encrypting the data with an appropriate key known

only to the legitimate receiver or group of receivers. Again, the authentication will have to be based on the identity management system described in section 4.3.1.

Attribute authentication

Like with the previous two mechanisms, attribute authentication requires a suitable authentication protocol that considers the mentioned problems. However, it is not the goal of attribute authentication to identify certain entities but instead to ensure certain attributes are valid, with a communication partner (usually the sender of data) , or that an entity possesses certain credentials. The *No direct packet forwarding* Problem in the previous section might play a special role here, as one might want e.g. to ensure, that only real vehicles have contributed to a distributed traffic density calculation and not attackers with notebooks.

Authentication of intermediate nodes

When multi-hop position-based routing is used in some applications, authentication of intermediate nodes may prevent some attacks (e.g., Denial of Service). However, we think that this mechanism is only of reduced importance and might be considered only as an option in the communication protocol.

Total anonymity

This and the following two concepts are mostly related to Section 4.3.4 and will be again discussed in full detail there. For an initial analysis, we can first constitute that communication is the main cause of privacy problems in IVC. By communicating, vehicles report their existence plus potentially other data, like their position, speed, but maybe even license plate or owner information. Privacy enhancing mechanisms need to be designed and used in a way that they try to reveal this information from eavesdroppers. Whereas some information may be hidden from eavesdroppers by means of encryption, at least legitimate recipients of data need the ability to decrypt the data. Even worse, the pure existence and identification of a vehicle cannot easily be concealed and will be reported in clear. Therefore, it is envisioned that cars will change their identification regularly by using pseudonyms. Changing pseudonyms may however create problems for applications that need a session semantics lasting longer than the interval between pseudonym changes. Here some session management will be necessary. The pseudonyms can then be used with the authentication mechanisms instead of the regular identifiers.

Resolvable anonymity

In addition to what has been said in the previous section, resolvable anonymity does not create any additional problems for the communication system, as long as this resolvable anonymity mechanisms do not require additional bidirectional communication acts during authentication. This has to be prevented, as again e.g. applications using one-way communication may suffer.

Location obfuscation

In some cases, location obfuscation may solve some privacy problems, as the quality of location profiles can be reduced. However, the accuracy and thus value of position-based routing and especially of applications relying on exact position information is reduced at the same time. Therefore, this is not a real option for general IVC.

Integrity protection

Integrity protection is of huge relevance to practically all IVC applications as modified data may damage the operation of every application. Therefore, communicated data needs some integrity protection which can be achieved either Message Authentication Codes (MACs) when using symmetric cryptography or by digital signatures with asymmetric cryptography. Problems are that MACs required the exchange of a shared key between sender and receiver and digital signatures may create a huge overhead in terms of bandwidth and computing power.

Encryption

Encryption of communicated data is the standard way of ensuring confidentiality in unicast communication and will also be used in VANETs. However, it might be a problem to negotiate session keys in one-way communication. Relying only on asymmetric cryptography on the other hand creates a significant overhead. Group communication and message dissemination create additional problems; however the analysis has shown that applications using such communication patterns usually need no confidentiality and thus no encryption of data. When the encrypted data also includes checksums and

is carefully combined with authentication mechanisms, encryption may also provide data integrity properties. MACs reach a similar goal with symmetric cryptography.

Detection of protocol violation

Certain communication procedures, like routing or authentication, but also application protocols may be formalized so that an intrusion detection system (see Section 4.4.2) can detect a protocol violation which also in some way affects the "integrity" of the overall system. However, this poses the requirement to somehow model the communication procedures to allow automatic detection of violations. Furthermore the question of reaction arises, especially when offending vehicles can simply change their pseudonym and remain undetected.

Consistency/context checking

This mechanism will mainly be addressed in Section 4.4.2. Like in the previous section, the system needs some formal model of the exterior world and check whether the data read by sensors and communicated by wireless communication is consistent with this model.

Attestation of sensor data

By having this form of attestation – where the solution is beyond the scope of communication – sensor data could be communicated to other vehicles which could then check the credibility of this data. A question arising in the security domain is how this attestation could be preserved when data is processed and aggregated in vehicles.

Location verification

This is a special form of consistency/context checking. Vehicles will periodically announce their current position in beacon packets. The goal of this mechanism – which is in more detail discussed in Section 4.4.3 – is to check the consistency of these position announcements. Besides the fact that positions transmitted need to be authenticated and integrity protected, communication is not directly touched here.

Tamper-resistant communication system

Protecting security-relevant data like key material inside cars plays an important role. Details are discussed in Section 4.3.3. If it can be assumed that key material is stored securely and cannot even be accessed by vehicle owners, symmetric cryptography may be used to a larger extend and the need for online PKIs is lessened.

Digital Rights Management (DRM)

The SeVeCom architecture will utilize DRM mechanisms mostly in the context of secure and controlled distribution of software to vehicles.

Replay protection

Replay protection is an important mechanism in many security protocols. Standard solutions include timestamps, random challenges, or sequence numbers in order to limit the validity of messages so a replay can be detected and ignored. Timestamps require synchronized clocks whereas sequence numbers are useful mainly if multiple packets from one sender are received by a receiver. Usually the later ones are more practicable, but in vehicle communication we have a special situation. Timestamps require synchronized clocks which is normally a problem in distributed systems. But as vehicles are assumed to be equipped with satellite navigation systems which provide clocks with high accuracy, this is not a problem. On the other hand, communication between vehicles may be extremely short lived, so often any particular vehicle may receive only one single packet from a specific other vehicle. In this case, sequence numbers are of no use, which is why random challenges provide a solution to replay attacks for short-lived communications sessions.

Jamming/DoS protection

Protecting the communication system from jamming attacks is closely related to the security of the physical layer which is not in the direct scope of SeVeCom. If taken a little bit more general, of course mechanisms that address different forms of overloading DoS attacks at various levels are a concern that must be addressed.

Access control

Generally speaking, access control is the combination of authentication and authorization. Authorization describes various mechanisms that control what (authenticated) subjects are allowed to request which operations on given objects. The entity checking whether access is granted or denied is called a reference monitor. This procedure happens at various occasions in the security architecture, especially in the in-board system which is not the scope of this section. The communication system cannot really prevent any vehicle to send a message, however the system can check incoming messages and authorize their interpretation or simply drop the message, based on the identity or other properties of the sender.

Closed user groups

Based on a check of allowed senders like described in the previous section the security system can easily implement closed user groups.

Firewall/Checkpoint

This mechanism is also related to the *Access Control* mechanism described above. It is similar to the closed user groups, but more generic. Based on arbitrary rules, incoming or outgoing packets can be analyzed and forwarded or dropped. The specific challenge in IVC is the fast changing network. Static rules based on neighbour addresses will probably not be helpful, as these neighbours change very quickly. Instead attributes should be based on attributes, relative locations, etc.

Sandbox

This is a in-vehicle protection mechanism that can prevent downloaded software from taking over the system. See Section 4.4.1 for details.

Filtering

This is a mechanism that is mostly identical to firewall. Based on rules, data packets are considered or dropped.

4.3 Priority Research Areas

This section provides analysis work carried out in SeVeCom priority research area.

4.3.1 Key And Identity Management

The identity of the entities that make part of a vehicular communication (VC) system is data that uniquely characterize them. In general, an identity can be context-specific. Independently of VC, vehicles and transportation systems have been in place for many years, and so have administrative processes, including management of the identities of the involved entities. In this section, we first briefly describe what has been the status quo before the advent of VC systems. Then, we pose the question and consider what can and will constitute an identity in the context of VC.

Vehicles have a predominant role in VC, while a tight coupling between vehicles and users, especially drivers, will usually exist. In general, the driver-vehicle relation is many-to-many, as a driver can operate many vehicles, and similarly, many users may be entitled to operate a particular vehicle. Even though the two types of entities are clearly distinct, as it will become clear later, they may be bound to each other.

Currently, the identities of vehicles and (their) users are managed by a variety of organizations. The identity of the users is in general established by states (e.g., identity cards, passports), and in the context of transportation by specific organizations, such as the Department of Motor Vehicles (DMV), which grant drivers licenses and attest to the ability of users to operate a vehicle.

The DMV is responsible for the identification of vehicles as well. On the one hand, the registration process, which is repeated periodically, has basically a two-fold output. First, a license plate that uniquely identifies the vehicle, determining the issuing authority, perhaps a division within the area covered by or corresponding to the authority, and an identifying string. Second, a binding between the plate, the vehicle, and the owner of the vehicle.

Nonetheless, identification is not done by the authority (e.g., DMV) alone, but can involve manufacturers. The vehicle itself is, on the other hand, identified by a presumed unique and assigned by the manufacturer vehicle identification number (VIN), as well as technical details such as manufacturer, date of production, model and color.

All these 'brick-and-mortar' attributes are expected to be part of digital identities, which are to be defined in VC systems. Nonetheless, an electronic-world identity of a vehicle can be significantly broader, or multiple identities may exist and used alternately as needed. The reason is that a large variety of applications will emerge, mixing not only attributes as those mentioned above, but also including new ones that convey access control privileges to on-line data and services. The variety of applications will be commensurate with the multiplicity of identities that will be used by vehicles and users.

At the same time, the VC systems will necessitate, beyond the application context, a within-the-network identification of nodes. This will transcend the entire networking protocol stack: network addresses at the data link and network layers (e.g., NIC and IP address respectively), end node identifiers (e.g., TCP port), and user-friendly names. All these identifiers, seemingly independent according to the layering concept, as well as other context specific data, such as geographical coordinates, can be critical in terms of privacy as discussed in Section 4.3.4.

A trusted third party or authority manages not only the identities but also the credentials and cryptographic keys of all network nodes. For the rest of the discussion we denote this as the certification authority (CA). This approach is deemed appropriate, instead of an ad-hoc or web-of-trust (PGP-like) method. Rigid identity and credential management processes for vehicles and drivers have been long in place, accountability and attribution of liability will continue to be crucial, and mechanisms for access control will be necessary. This is clearly reflected on the intent of the US DoT Intelligent Transportation Systems (ITS) initiative to base its security solutions on a Public Key Infrastructure (PKI), as stated by the IEEE 1609 family of standards for Wireless Access in Vehicular Environments (WAVE) [54], [55].

Basically, without an appropriate certificate, network nodes should be essentially unable to participate in the system operation. Nevertheless, the certificate(s) cannot be valid for unlimited periods of time after their generation. Moreover, the CA reserves the right to revoke the certificate(s) of any node, and essentially evict the node from the network.

4.3.1.1 Problem Analysis

A vehicular communications system comprises a number of interacting entities that we classify broadly as: (i) Users, (ii) Network nodes, and (iii) Authorities. Our focus in SeVeCom is on the network operation and the communication of the computing devices, i.e., largely, the network nodes that we define more precisely below. Nevertheless, users, that is, individuals operating vehicles, are instrumental in determining the vehicle behavior and the overall transportation system operation, and thus warrant a distinction.

Network nodes are processes running on computing platforms capable of wireless communication; they are mounted on vehicles and road-side units (RSUs). We denote the RSUs collectively as the road-side infrastructure (RSI). The complexity of the nodes can vary from relatively powerful devices (e.g., on-board vehicle computers) to relatively simple ones (e.g., alert beacons on the road-side).

The authorities are public agencies or corporations with administrative powers in a specific field; for example, city or state transportation authorities. They are responsible for instantiating procedures, as those currently in place for vehicle registration and driver license issuance, as well as vehicular network entities that act on behalf of the authorities and provide services. For the rest of the discussion, we refer to authorities only as network entities, unless noted otherwise. A detailed discussion of the VC system operational assumptions follows.

Authorities

Authorities are trusted entities responsible for the issuance and management of identities and credentials for parties involved in the vehicular network operation. In general, authorities can be multiple and distinct in their roles and the subset of network parties in their jurisdiction.

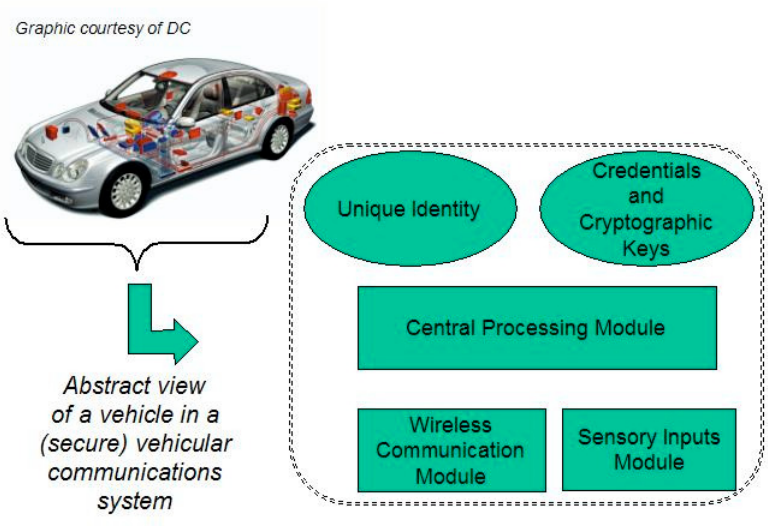

Figure 4-1: Model of a network node

We denote the set of system entities, S_X, registered with an authority X determined by geographical, administrative, or other criteria, as the domain of X. All parties in S_X trust X by default.

The presence of on-line authorities is not required, as connectivity and communication, especially over the wireless medium, with an authority may be intermittent. Nodes can in general establish two-way communication with the authorities, even though one-way communication, from an authority towards the nodes can be meaningful as well.

Vehicle Identification and Credentials

Each vehicle has a unique identity V, and a pair of secret or private and public cryptographic keys, k_V and K_V respectively. The binding of K_V to V, and the binding of K_V to other data or attributes pertinent to V are achieved by an identity certificate or an attribute certificate respectively. We denote a certificate on K_V issued by an authority X as $Cert_X\{K_V, A_V\}$, with A_V being a possibly void attribute list. The addition of a lifetime field to the vehicle certificate is possible, as detailed in Section 5.3.3.

The vehicle identity, V, denotes the on-board central processing and communication module. Other on-board sensing, actuating, and processing units are identifiable locally, with V having full control (access/read/write) over those resources. In other words, we abstract away the complexity of the on-board equipment, which could essentially be viewed as a wired network of its own, as shown by the illustration of an in-car system by Daimler-Chrysler in Figure 4-1. Thus, we consider a network node to comprise:

- A unique identity V
- A public/private key pair K_V, k_V
- A module implementing the networking and the overlying application protocols
- A module providing communication across a wireless network interface.
- A module providing the sensory inputs from all on-board sensor equipment.

This abstraction, illustrated in Fig.4-1 for a car, is applicable to vehicles and infrastructure nodes alike.

Infrastructure Identification and Credentials

Each infrastructure node has a unique identity, I, and k_I and K_I private and public keys. $Cert_Z\{K_I, A_I\}$ [FK2]is a certificate issued by an authority Z for the infrastructure unit I with attribute list A_I. Similarly to the vehicle certificates, a lifetime can be specified.

A subset of the infrastructure nodes serves as a gateway to the authorities, or inversely, from the point of view of the authorities, a gateway to the mobile wireless part of the vehicular communications system. Infrastructure nodes, or a subset of those, can be considered as more trustworthy than other nodes, with respect to specific functionality or attributes. For example, infrastructure nodes can be assigned the role to transmit specific (safety or not) messages whose content is trusted as correct or given precedence over other messages. RSUs can be, for example, assumed to have absolute and relative locations that are in most cases fixed and thus often known or straightforward to infer.

Public Vehicles

Vehicles are distinguished in two categories, public and private. The former can include public-safety (highway assistance, fire-fighting) or police vehicles, aerial vehicles (e.g., police helicopters), or even public transportation vehicles (buses, trams). Public vehicles, similarly to infrastructure nodes, are considered more trustworthy, and they can be used to assist security related operations.

User Identification and Credentials

Each user of the vehicular communications system has a unique identity, U, and a pair of private and public cryptographic keys, k_U and K_U respectively. $Cert_Y\{K_U, A_U\}$, again, possibly with a lifetime field. is a certificate issued by an authority Y for a user with an A_U attribute list. The user can be bound to its credentials and secrets through some token she/he uniquely knows or possesses (e.g., pass-phrase, biometric data).

User and Vehicle Association

The user can be the owner and/or the driver of the vehicle, or in general any passenger. The association of vehicles and users is in general many-to-many, however, at each point in time only one user can operate a vehicle. For the rest of the discussion, we make the simplifying assumption that the user is the individual that operates the vehicle, i.e., the driver. User access to the vehicle relies on the possession of a type of credential (e.g., physical key, PIN, biometric).

Trusted Components

Nodes are equipped with trusted components further called tamper evident security modules (TESMs), i.e., built-in hardware and firmware with two types of functionality: (i) cryptographic operations, and (ii) storage. The role of TESMs is two-fold: to protect the vehicle's cryptographic material and their use, and to safeguard data usable for liability identification.[FK3]

The TESMs enforce a policy on the interaction with the on-board software, including the access and use of the securely stored keys, credentials, and secrets. Access (read or write) to any information stored in the TESMs and modification of their functionality is possible only through the interface provided by the TESMs. For example, the protected information cannot be exposed through the execution of any sequence of the commands provided by the interface. Similarly, the policies enforced by the TESMs specify the authorized entities to modify information and functionality.

Cryptographic operations, with signature generations and verifications expected to be the more frequent ones, are performed without the TESM revealing the cryptographic material to the potentially compromised or faulty computing unit. On-the-fly data and outcomes of computations are also stored, with those corresponding to a recent interval $[t_0, t]$ maintained if the TESM is triggered by a specific event at time t, to provide protected audit trails.

The TESMs have to resist some tampering, in order to provide enhanced protection of the cryptographic material and other data. Tamper-resistance can also imply that keys, credentials, and other secret data are physically bound to the on-board platform. It is however possible to minimize or waive the requirement for tamper-resistance; for example, the points of the audit trail can be signed or encrypted data.

Assuming on-board TESMs is in accord with the current state and developments in vehicle equipment, which already includes hardware components and firmware that regulate or record information on the vehicle operation and on its users' inputs. Examples are speed limiters, tachographs and event data recorders (EDRs [65]). These may not be necessarily tamper-resistant but they are tamper-evident,

and it is commonly accepted, among manufacturers (more so in the US) and legislators, that TESMs will be routinely present.

4.3.1.2 Research Contribution

Revocation

The advantages of having an authority that manages cryptographic keys and identities for VC are accompanied by some challenging problems, notably certificate revocation. For example, the certificate(s) of a detected attacker or malfunctioning device have to be revoked, i.e., it should not be able to use its keys or if it still does, vehicles verifying them should be made aware of their invalidity.

The most common way to revoke certificates is the distribution of CRLs (Certificate Revocation Lists) that contain the most recently revoked certificates; CRLs are provided when infrastructure is available. In addition, using short-lived certificates automatically revokes keys. These are the methods proposed in the IEEE P1609.2 standard [55]. But there are several drawbacks to this approach. First, CRLs can be very long due to the enormous number of vehicles and their high mobility (meaning that a vehicle can encounter a high number of vehicles when traveling, especially over long distances). Second, the short lifetime of certificates still creates a vulnerability window. Last but not least, the availability of an infrastructure will not be pervasive, especially in the first years of deployment.

To avoid the revocation shortcomings above, a number of revocation protocols have been designed, namely RTPD (Revocation Protocol of the Tamper-Proof Device), RCCRL (Revocation protocol using Compressed Certificate Revocation Lists), and DRP (Distributed Revocation Protocol). We present them briefly (this section is taken from [41]).

In RTPD, once the CA has decided to revoke the key(s) of a given vehicle M, it sends to it a revocation message encrypted with the vehicle's public key. After the message is received and decrypted by the TPD of the vehicle, the TPD erases all the keys and stops signing safety messages. Then it sends an ACK to the CA. All the communications between the CA and the vehicle take place in this case via base stations. In fact, the CA has to know the vehicle's location in order to select the base station through which it will send the revocation message. If it does not know the exact location, it retrieves the most recent location of the vehicle from a location database and defines a paging area with base stations covering these locations. Then it multicasts the revocation message to all these base stations. In the case when there are no recent location entries or the ACK is not received after a timeout, the CA broadcasts the revocation message, for example, via the low-speed FM radio on a nationwide scale or via satellite.

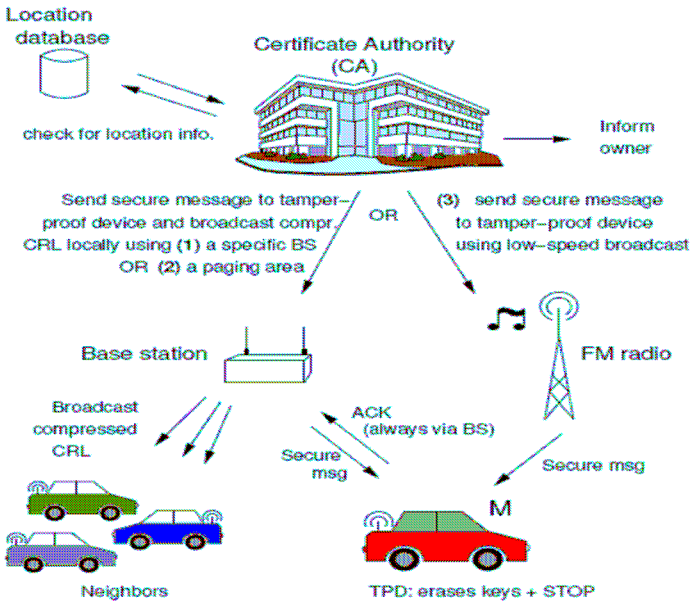

Figure 4-2: Illustration of revocation information dissemination mechanisms

The RCCRL protocol is used when the CA wants to revoke only a subset of a vehicle's keys or when the TPD of the target vehicle is unreachable (e.g., by jamming or by tampering of the device). Given the expected large size of CRLs in VANETs, the key idea in RCCRL is to use Bloom filters - a probabilistic data structure used to test whether an element is a member of a set. Thus, the size of a compressed CRL or CCRL will be only a few KB. RCCRL also relies on the availability of infrastructure that broadcasts the CCRLs once every 10 minutes. Compared to RTPD, RCCRL has the special feature of warning the neighbours of a revoked vehicle as they also receive the CCRLs.

The DRP protocol is used in the pure ad hoc mode whereby vehicles accumulate accusations against misbehaving vehicles, evaluate them using a reputation system and, in case misbehaviour is detected, report them to the CA once a connection is available. Unlike RTPD and RCCRL, the revocation in DRP is triggered by the neighbours of a vehicle upon the detection of misbehaviour. Mechanisms for the detection of malicious data can be leveraged to spot vehicles generating these data (since all messages are signed).The above protocols, with DRP renamed to LEAVE, as parts of a broader framework for eviction of misbehaving nodes from the VC system, are described in the more recent [42].

4.3.2 Secure Communication Protocols

As already mentioned, basic questions about secure communication regard which and how security mechanisms can be used to secure communication protocols, and how these security mechanisms can be integrated with the actual functional components, like the routing or medium access.

Regarding the security mechanisms to apply, we first have to identify which communication protocols will finally be used. As other projects currently also do not have a specification ready and most likely will use different variants of protocols, we extrapolate basic "**communication patterns**" from our application analysis, which also served as a basis for the definition of requirements. The usage of communication patterns instead of concrete protocols also has the advantages that we stay independent of the implementation details and security mechanisms can easily be adapted to similar communication protocols.

The identified communication patterns are the following:

- Beaconing
  Periodic, single-hop broadcasts, containing e.g. a vehicle's location, heading etc.
- Flooding/Geocast
  Multi-hop broadcast over a certain number of hops (restricted by TTL or by specified geographic destination region)
- Geographic unicast routing
  Multi-hop, hop-by-hop forwarding of packets, either for unicast end-to-end connections, for anycast requests or for subsequent flooding/geocast in a remote destination region.

Because the communication patterns differ substantially in their mode of operation, they also partly require different mechanisms to thwart security and privacy infringements, as already described previously. A subset of problems and existing work on the communication patterns is given in the following.

Secure Beaconing

Beacons are packets that are sent periodically via broadcast over a single hop, which means that they are not relayed by receiving nodes. This kind of communication is useful for instance for all cooperative awareness applications.

From the security point of view, a reasonable level of security of beaconing can be achieved with some basic functions. As a generally important building block, beacon packets should be signed, which in turn can ensure authentication of the sender, attribute authentication and integrity protection. Moreover, beacons should carry a timestamp that prevents replaying them at a later time.

For typical beacon packet content like the current location of the sender, receivers may apply location verification mechanisms to detect falsified location information.

However, due to their high frequency, a number of challenges on the application of crypto mechanisms arise:

- Because of their frequency, beacons can cause a substantial part of the overall channel load. This situation is aggravated if every packet has to carry a complete set of security data like signature and certificate. Therefore, it would be desirable to reduce the channel load by more sophisticated security solutions. At the same time, each packet should be self-contained, i.e.

authentication and integrity checks should be achievable without the context of other packets to allow for fast evaluation of time-critical packets.

- A similar problem due to high frequency of beacons originates from the computational requirements of asymmetric crypto operations. It is well known that creation and verification of asymmetric signatures can consume considerable amount of time. For example, if we assume beacons to be sent with frequency ***f*** and the current vehicle density is ***d***, then ***f*** signature operations and ***f*d*** signature verifications have to be performed per second. Moreover, as some applications need time-critical communication to some extent, the sum of both the time for creation and verification plays a role.

Secure Restricted Flooding/Geocast

The Flooding/Geocast communication pattern provides a distribution of a message over multiple hops either as long as the time-to-live (TTL) counter is larger then zero, or as long as receivers are currently located within the specified geographic destination region. To thwart impersonation, freshness and manipulation of the packets, it is also sufficient to attach a timestamp and to have the original sender sign packets, at least the parts that are not modified in transit. For simple geocast, no fields need to be changed by intermediate nodes, as the destination region is specified by the sender and it never needs to be altered afterwards. However, implementations might want to include the last forwarding hop into the packet e.g. for consistency checking. Then things get more difficult and we might need additional hop-by-hop signatures. For TTL-based flooding, the Time-to-live must be decreased at every forwarding vehicle, which opens up the opportunity for an attacker to increase the TTL and thus to cause more network load. Yet, this attack can be prevented by applying a hash chain mechanism. The use of such a mechanism for this purpose, to constrain the propagation of a message across a limited number of hops, was proposed in [35]. Then, a TTL can only be decreased by an intermediate node, but not increased. As simple flooding usually distributes packets with considerable redundancy because every node relays a message, decreasing the TTL by a single attacker has only a negligible or even no effect, because other nodes will forward the packet regularly.

Another particular problem about flooding and geocast is that these mechanisms generate a considerable amount of network load. For example, if a message is forwarded to a very large area, where every vehicle rebroadcasts the message once, this leads to ***n*** broadcasts, if ***n*** vehicles reside within the area. If the attacker forges bogus messages with large destination areas frequently, the channel will soon get overloaded. For this case, we intend to include a rate control mechanism and a maximum size of TTL or geographic region, dependent on the type of originating vehicle. This is also proposed in [19].

Though these basic measures already can help against attackers, there are still open problems to be addressed:

- As soon as the notion of node location plays role, there is always an attack opportunity against the positioning system that provides nodes with the current position. Thus, secure positioning could help for all position-related packet types in the network. If not all vehicles in a certain area are tricked in parallel (e.g. by a fake GPS satellite), also a heuristic approach to position verification can be helpful.
- Simple flooding and geocast mechanisms typically use broadcasts to send packets to all neighbours at once. Therefore, packets are not acknowledged by the receiver, which allows an attacker to selectively destroy packets on the data link layer. For a receiving node, the attack would just look like a collision which happens regularly in wireless ad hoc networks. Because flooding and geocast both generate a lot of redundancy if every intermediate node rebroadcasts a packet, this is not a problem in a large area where multiple paths exist and where an attacker only has a local impact. But, on highways, the radius of the transmission range is often enough to block all packets of one message from further forwarding. As there is no retransmission, these packets will get lost and the flooding/geocast ends there.

Secure Geographic Routing

With geographic routing, we denote multi-hop single-path forwarding according to the principle of greedy geographic routing, like it is used e.g. in GPSR [24] or CGGC [30]. The messages' destination is a geographic coordinate rather than a node address. The basic concept of geographic forwarding is to pass messages always to a neighbor node, which is geographically closer to the destination than the current node. To be able to select such a next hop for a packet, every node needs to know his one-hop neighbors and their current positions. For that, greedy geographic routing requires a periodic beaconing service to get the described neighbor information. More advanced mechanisms can work without beaconing [16], however these mechanisms also have drawbacks, and as we need beaconing in VANETs anyway, we refer to the original form here.

The reason why this type of routing was favored over topological routing protocols for ad hoc networks like AODV or DSR is that is has significant advantages in ad hoc networks with very high dynamics like it is the case in inter-vehicle networks [15].

To secure geographic routing, there are several aspects to be considered. Like in the previously described patterns, packets must be integrity protected and it is helpful to guarantee that packets can only be generated by legitimate participants of the network, like registered vehicles or RSUs. This can be achieved by signing packets.

The more difficult aspects concern one of the building blocks of geographic routing, the beaconing. Apart from the general security considerations of beaconing (see beaconing pattern), there are more problems to be solved with geographic routing. Such issues are investigated in [19].

Falsified position claims

Because the greedy routing metric selects the next hop neighbour to forward the message according to their given position, an attacker may give a forged position in his beacons to trick other vehicles. In [26], we have shown that this is a serious problem that can e.g. lead to loops and may decrease the performance of the network considerable. Even worse, on highways, an attacker may utilize forged position claims to reroute the whole traffic along the road to himself. An example of that problem is given in Figure 4-3.

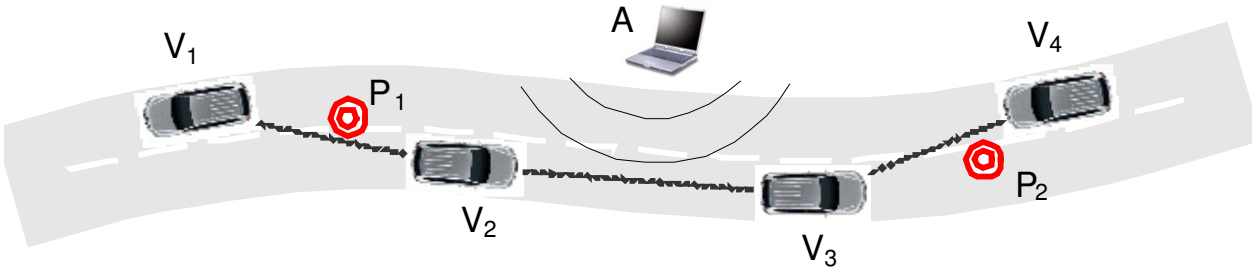

Figure 4-3: Attacker A is able to reroute all data traffic along the road by forging two identities at positions P_1 and P_1

Using a Sybil attack, A sends faked beacons with only two different identities claiming to be at positions P_1 and P_2. These strategic positions allow the attacker to control all the data traffic in both directions. In the example, when messages are forwarded from left to right, they eventually reach V_2. This vehicle would then determine one of his neighbours as next hop which is closest to the destination. As data is forwarded to the right, the rightmost neighbour will be selected. Without attack, this would be V_3, but when the attacker claims to be at P_2, V_2 will pass the packets to A instead, because P_2 is closer to the right [28].

To thwart this rerouting attack we propose plausibility checks which enable us to detect a large amount of such kind of falsified position claims. Details on this system can be found in [29].

Privacy – Efficiency tradeoff

Another problem relates to the proposal to preserve privacy by changing a vehicle's network identifier regularly. As described before, geographic routing relies on neighbour awareness, which is done by beaconing. However, if the neighbour vehicles' identifiers (temporary pseudonyms) change from time to time without announcement (which actually would render pseudonyms useless), this leads to frequent forwards to addressed of neighbours which actually are not reachable any more at this address. This effect in conjunction with high frequency of pseudonym changes can lead to considerable decrease in efficiency of geographic routing.

As privacy is a very important requirement for private vehicles in VANETs, we cannot afford to do regulations on the pseudonym change algorithms as a solution to the problem. However, there is a mechanism available in the network stack that can be used to get around timeouts in the neighbour table: acknowledgements on data link layer. Because unicasts from one hop to the next are acknowledged on data link layer, the routing could be informed about the lost link by a cross layer notification. If every packet is saved for some time, or if the MAC layer is also able to return the packet to the routing in such a case, packets can be retransmitted to another forwarding node and will not be lost. Still, this causes considerable waste of channel bandwidth, but at least the performance of the routing can be improved by such a mechanism. Details on the analysis of the problem and the proposed solution can be found in [43].

Other communication patterns

Beyond the described communication patterns, inter-vehicle communication may use more types, partly also for information and business applications.

End-to-end connections

In contrast to most safety applications, where broadcast and information of all other vehicles is predominant, some use case ideas also set up bidirectional end-to-end connections like they are typical for today's communication networks. Examples of such applications are e.g. a vehicle safety inspection by a police car, a connection to a roadside unit or a connection to a background network service. In all these cases, confidentiality as a requirement is much more relevant than for eSafety applications.

Information Dissemination

Another evolving type of communication is mainly intended to distribute information within the network and can be used for a variety of information and warning functions. Examples include the widespread dissemination of information on road and weather conditions, traffic jams or accidents. Information dissemination can be seen as a more intelligent way of flooding; however it may also include data processing and aggregation, which is clearly beyond packet flooding. Sometimes the mechanism also focuses on holding the information available in the network as long as it is relevant which is known under the term stored geocast.

4.3.3 Tamper Proof Device and Decision on CryptoSystem

Implementing security services for vehicular communications require the vehicles to store sensitive data, such as cryptographic keys (secret keys, private keys), event logs, etc. It must be assumed that potentially malicious parties, such as maintenance service providers or even the vehicle owner, can have unsupervised access to the vehicle for extended periods of time. In addition, these potentially malicious parties may have incentives to compromise the sensitive data stored by the vehicles. For these reasons, the sensitive data needs to be protected from unauthorized access by physical means. In other words, the sensitive data must be stored in a device that is hard to tamper with. If implemented in hardware, such devices are referred to as tamper evident security modules (TESM), as successful tampering of the hardware will always leave physical traces.

In the SeVeCom Project, we do not intend to carry out research in the field of designing TESMs. However, we do want to specify the basic requirements on TESMs intended for vehicular communication purposes. For this reason, we need an understanding of how TESMs work, what level of protection they can provide, and how they can be attacked. Therefore, in the first part of this section, we give an overview on these issues.

The second part of this section is concerned with the selection of an appropriate cryptosystem (digital signature scheme) for vehicular communication purposes.

4.3.3.1 Analysis of Tamper Evident Security Modules

In general, the job of a TESM is to securely generate and/or store long term secrets for use in cryptography and to physically protect the access to those secrets over time. As such, the TESM consists of a piece of hardware that is designed to resist physical intrusions aiming at opening the device and getting direct access to its components (e.g., to the memory module where the secrets are stored). In addition, the TESM has some firmware and software components that implement its logical services. These services can be accessed through an Application Programming Interface (API), which is the highest level software component of the TESM. Typically, the API offers the possibility to invoke various lower level functions that together implement the services of the TESM. Most TESMs are equipped with special cryptographic hardware that accelerate complex cryptographic operations (e.g., large integer arithmetics).

Attacks against TESMs

Attacks against TESMs can be **physical or logical attacks**. Logical attacks aim at exploiting weaknesses in the operating system of the TESM, in the software implementations of the cryptographic algorithms running on the device, or in the API that provides access to the functions implemented on the TESM. Often, carrying out logical attacks does not even need physical access to the TESM itself. Physical attacks on the other hand need physical access to the TESM, and they can be categorized as follows:

- **Invasive attacks:** In these attacks, the attacker tries to access directly the inner parts of the device. This usually needs opening the housing or packaging of the TESM, and often results in damaging the device. An example of this attack is when the attacker uses micro-probing needles to directly access the bus inside an IC.

- **Semi-invasive attacks:** In this kind of attack, the attacker tries to access the inner parts of the TESM without actually opening and damaging the packaging. For example, the attacker can use a laser beam to influence the state of a flip-flop inside an IC.
- **Non-invasive attacks:** In a non-invasive attack, the attacker observes the operation of the TESM, and she can manipulate the environment of the TESM (e.g., the attacker can insert clock glitches into the clock signal). Side-channel attacks such as power analysis and timing attacks belong to this category.

Based on their knowledge and resources, we can classify attackers as follows [1][FK5]:

- **Clever outsiders:** They are often very intelligent but may have insufficient knowledge of the system. They may have access to only moderately sophisticated equipment. They often try to take advantage of an existing weakness in the system.
- **Knowledgeable insiders:** They have substantial specialized technical education and experience. They have varying degrees of understanding of parts of the system, but potentially they have access to most of system. They often have highly sophisticated tools and instruments for analysis.
- **Funded organisations:** They are able to assemble teams of specialists with related and complementary skills backed by great funding resources. They are capable of in-depth analysis of the system, designing sophisticated attacks, and using the most advanced analysis tools. They may use knowledgeable insiders as part of the attack team.

Examples of TESMs used in practice

We briefly describe the two extremes of the spectrum of TESMs: smart cards and the IBM 4758 high-end cryptographic co-processor. In addition, we also give a brief overview on the Trusted Platform Module (TPM) which can be positioned somewhere in the middle of the spectrum. Our objective with these overviews is to understand what tamper resistance means, what TESMs are capable for, and what the trade-off is between the features and the cost of the device.

Smart cards

A smart card is microcomputer embedded in a plastic card (credit card size or smaller). It has a CPU, some memory, and an input/output interface. In a smart card, the input/output interface is not connected to a system bus, but it is connected directly to the CPU. Hence, the CPU can control all the traffic between the card and the card reader. The main difference between a smart card and an ordinary computer is that in a smart card, all the parts are realized in one chip. The advantage of this is that a single chip can be protected from physical attacks easier than a complex device. Many smart cards support cryptographic operations by featuring some custom hardware for DES and modular arithmetic.

Smart cards are used in a wide range of applications, such as bank cards, GSM SIM cards, electronic tickets for public transport systems, payTV applications, access control to buildings, electronic ID cards, and e-passports. In all these applications, smart cards store sensitive data, such as cryptographic keys (even system master keys), access codes, account balance, etc. Smart cards are intended to protect these sensitive data in hostile environments, but we must note that in many applications, smart cards are complemented with other security measures (e.g., video surveillance, transaction log analysis and blacklisting). When such additional security measures are not applied, smart cards often become less effective and fraud occurs (see e.g., payTV systems).

Smart cards prevent unauthorized access to the information that they store at two levels. First of all, the smart card has a single interface, through which it can communicate with the external world. Access to sensitive data through this interface is controlled by the operating system of the smart card. Granting access is based on entering and checking PIN codes. An additional feature is that after a certain number of unsuccessful attempts to enter the PIN code, the card usually blocks itself and does not allow any further access attempts.

The second level of defense is that, by their construction, smart cards provide a certain level of physical security (i.e., tamper resistance). However, the physical security of smart cards is only moderately strong, and they usually do not resist against the attacks of a determined attacker with substantial knowledge and special physical hacking equipments. According to the FIPS 140 criteria, smart cards are evaluated at level 2 or 3.

Attacks against smart cards can be classified into two categories: non-invasive and invasive attacks. Non-invasive attacks do not destroy the card. They include side-channel attacks, such as timing attacks and power analysis attacks. In general, side channel attacks rely on careful observation of the interaction of the card with its environment during critical operations. This often reveals some information about the sensitive data stored in the card. In addition, unusual operating conditions may have undocumented effects on the operation of the card, which may also be exploited in an attack.

For instance, unusual temperatures or voltages can affect EEPROM write operations, and power and clock glitches may affect the execution of individual instructions.

The invasive attacks destroy the smart card. They require the removal of the chip from the plastic cover and also the removal of the passivation layer from the chip. Once this is done, the chip becomes visible, and micro-probing needles or electron beam testers can be used to access on-chip signals and extract data from the chip.

The IBM 4758 cryptographic coprocessor

The IBM 4758 high-end cryptographic coprocessor is a programmable PCI board with custom hardware to support cryptography and tamper resistant packaging. Its main features include the following:

- pipelined DES encryption engine
- pipelined SHA-1 hash engine
- 1024-bit and 2048-bit modular math hardware to support RSA and DSA
- hardware noise source to seed random number generation
- pseudo-random number generator support for RSA key pair generation, encryption, and decryption
- support for key management (DES based, RSA based, key diversification, PIN generation)
- secure on-board clock
- support for PKCS#11 and the IBM Common Cryptographic Architecture (CCA)
- battery backed RAM (BBRAM) to store secrets persistently
- steel house with tamper detecting sensors and circuitry to erase the sensitive memory.

The IBM 4758 also has a hardware state controller that controls access to portions of the BBRAM and Flash memory. Essentially, it functions as a hardware lock separated from the CPU, and it denies unauthorized access to secrets even if the CPU runs a malicious application.

The physical security of the IBM 4758 is evaluated at level 4 (the highest level) according to the FIPS 140. The physical defensive measures include the wrapping of the module in a grid of conductors, which is monitored by a circuit that can detect changes in the properties of these conductors. The conductors are non-metallic and resemble the material in which they are embedded. Moreover, the grid is arranged in several layers, and the entire package is enclosed in a grounded shield to reduce detectable electromagnetic emanations. Inside the packaging, the IBM 4758 has additional tamper detection sensors, including sensors for measuring the temperature, humidity, pressure, ionizing radiation, and the changes in supply voltage and clock frequency. If any of these sensors raises an alarm, then the content of the BBRAM (the sensitive secret data) is erased and the whole device is reset.

The firmware, operating system, and the application code are organized into so called *code layers*. The device is shipped with code for booting. This code is divided into two parts called miniboot 0 and miniboot 1, which constitute code layer 0 and code layer 1, respectively. Miniboot 0 resides in ROM, while miniboot 1 is stored in flash memory. The OS and applications are loaded into the flash memory by miniboot 1, and they constitute code layer 2 and code layer 3, respectively.

Each layer has its own page in the BBRAM, where it can store its own secrets. The device has a mater private key, which is stored in page 1. The state controller ensures that code running at a given layer cannot access pages that belong to lower layers. This is achieved through the following operation: Each time a new layer is loaded and started, the state controller is stepped forward. Therefore, its state corresponds to the level of the currently running code. The state controller continuously monitors the requests issued to the BBRAM, and if it detects that the currently running code tries to access a memory page that belongs to a lower code layer, then it denies the access. In addition, the state controller prevents writing into the flash memory by the OS and the application. This ensures that a malicious OS or application cannot remove the integrity checks and the instruction to advance the state controller from the code of lower code layers (i.e., the miniboot 0 and 1).

Loading new software into the device at a given layer is authorized by code authorities. Code authorities are organized into a tree. Each authority has a key pair, which is certified by its parent authority. The root of the hierarchy is the miniboot 1 authority. Its public key is stored at layer 1 in the device, and hence, the miniboot 1 code can verify certificate chains starting from the miniboot 1 authorities public key. Code to be loaded into the device is signed by the corresponding code authority, and the certificate chain from the root to this authority is attached for verification purposes. Miniboot 1 verifies the signature on the code and the attached list of certificates, and if the verification is successful, then it loads the code into the flash memory.

Note that when an application running at code layer 3 wants to sign a message, it cannot use the device master private key, since that key resides at layer 1, and therefore the state controller denies access to this key for the application. For this reason, the applications and the OS have their own public/private key pairs, and their public keys are signed by the layer below. More specifically, the application keys are signed by the OS, and the OS key is signed by miniboot 1 with the device master private key. Finally, the master public key of the device is signed by the device manufacturer (i.e., IBM). When signing a message, the application uses its own private key, and attaches a chain of certificates that starts from the manufacturers certificate (containing the device master public key) and ends with the application's certificate (issued by the OS).

The Trusted Platform Module (TPM)

Trusted hardware and/or software devices are key for a secure communications system, especially in a vehicle that has a long life-expectancy. Therefore, security modules that use or contain cryptographic keys to identify a vehicle, or to manage its pseudonyms, should preferably be implemented in hardware. Such hardware can provide tamper evidence which results in illegal physical access to the module's content to not remain unnoticed. Security modules usually offer both data storage and cryptographic primitives execution facilities. A security module typically consists of:

- Specific hardware components that are used to support high-bandwidth communications;
- Cryptographic coprocessors to speed up the cryptographic operations that use private or secret keys (encryption, decryption, signature generation, key agreement);
- Secure storage to store the cryptographic private and secret keys;
- Software key store where cryptography-related data is stored in permanent memory and operated from non-secure micro-controllers.

A Trusted platform module (TPM) consists of a combination of the middle components: a cryptographic coprocessor with secure storage to store a few cryptographic secret and private keys.

A TPM is built by default in many new laptops and desktops to authenticate the device, and to link content to that device, e.g., for DRM purposes.

Today's TPMs lack features necessary for SeVeCom like authentication of users and application data on the one hand, and like high-bandwidth support on the other.

The ideal security module abstracts the possibilities of smartcards and TPMs by combining their functionalities to result in a trusted component, of which the feature set includes basic cryptographic operations like encryption, plus hashing and signing for authentication purposes. It covers also the generation of cryptographic keys and random seeds, execution of key agreement protocols, storage of data and certificates in a secure area, and provides security functions like secure logging, security mechanisms conversion or time checking. Given these primitives, the module can issue and manage the pseudonyms necessary to meet SeVeCom's privacy-specific requirements.

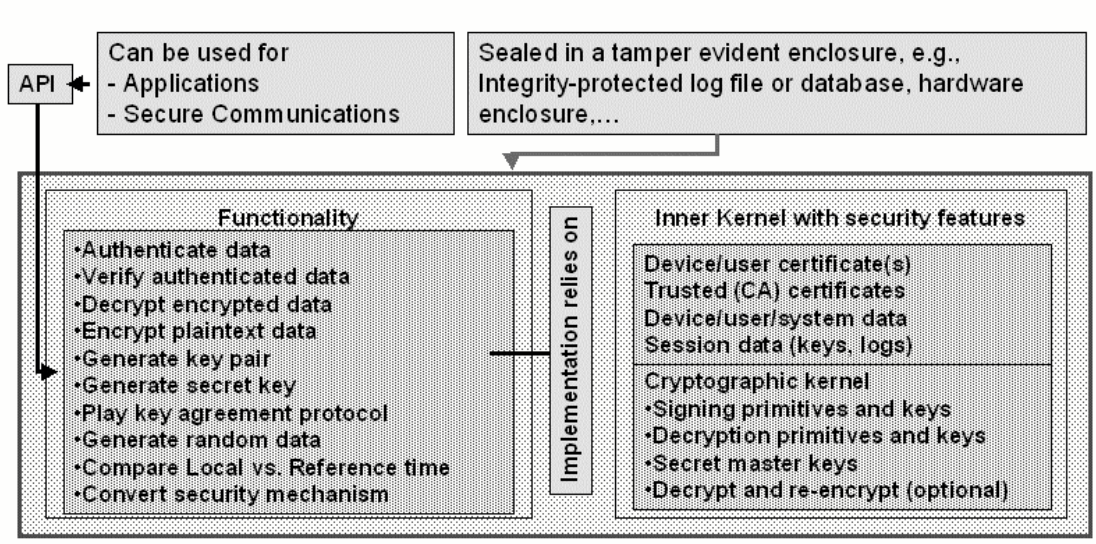

Figure 4-4: Abstract view on the functionality of a security module.

This last figure represents this security module concept, as the association of security features in a tamper evident enclosure to realise the actual physical security requirements of SeVeCom.

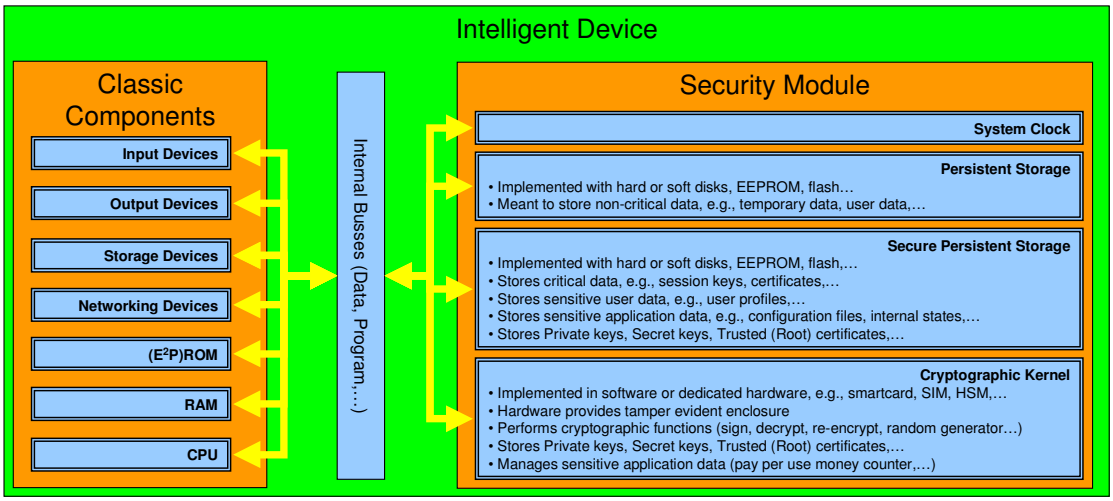

Figure 4-5: Interaction with a Security Module.

Depending on the level of protection and functionality needed, the actual implementation of this security module can be done using various secure hardware/software combinations. The authentication of information that will be broadcast, or the calculation of new pseudonyms are for instance the applications for which the security module has to be used. Note, however, that strict access control must be implemented to use the functionality of such security module, as it really manages the credentials of the device in which it is embedded: if a fraudster were able to request signatures from the security module, she can replay this information at a later stage to impersonate the original device. Preventing unauthorized access to the functionality of such module is a hard problem, certainly if the module is deployed in a vehicle which actually operates in an unmonitored environment.

Discussion

The main advantage of trusted modules such as smart cards (and low-end TESMs in general) is their low cost. In addition, smart card technology has already been proved to be useful in large scale applications (e.g., GSM SIM cards). The main disadvantage of low-end TESMs is their limited physical protection. Moreover, low-end devices often need an external power supply, which also implies that they cannot have an on-board secure clock. It may also be difficult to integrate additional custom hardware into them (e.g., a GPS receiver circuit cannot be built into a smart card). Finally, the performance of low-end TESMs (in particular smart cards) in terms of processing speed may not be sufficient in vehicular applications.

The main advantages of high-end TESMs (such as the IBM 4758) are their very high level of physical security, their high performance, and their flexibility in terms of code providers, and code update. Indeed, the layered trust model and code authorities of the IBM 4758 (or similar devices) may be conveniently mapped to a typical vehicular scenario (e.g., miniboot officer → security module manufacturer, OS officers → car manufacturers, application officers → car manufacturers, national authorities, and third party software providers). However, the main disadvantage of high-end devices is their very high price (e.g., the IBM 4758 costs in the order of 4000 USD). Another disadvantage is the limited battery lifetime and typically weak robustness (e.g., the operating temperature of the IBM 4758 must be between 10 and 40 Celsius, otherwise the device erases its BBRAM).[FK6]

4.3.3.2 Analysis of cryptosystems

In this part, we compare the existing digital signature algorithms, and examine their suitability for vehicular communications purposes. First, we select some candidate algorithms from the list of available algorithms, and then we compare them based on their message overhead and performance benchmarks. Finally, we give recommendation regarding the most suitable candidate.

Available digital signature schemes

In the scientific literature, a number of digital signature algorithms have been proposed. The most important ones are the following:

- RSA
- DSA (Digital Signature Algorithm) / ElGamal
- ECDSA (Elliptic Curve DSA) and variants (ECGDSA, ECKCDSA)

- LUC
- ID Based / Group based Signature Schemes
- XTR / ECSTR (Efficient and Compact Subgroup Trace Representation)
- ECNR (Elliptic Curve Nyberg-Reuppel)
- Cramer-Shoup cryptosystem
- Schnorr signature algorithm
- Rabin signature algorithm
- Pointcheval-Stern signature algorithm
- Paillier cryptosystem

Although the theoretical security of these signature algorithms has been proven in research papers – either in the standard or in the random oracle model – most of them cannot be used directly for practical applications. The reason for this is that it takes a lot of time and extensive research, before a cryptographic algorithm becomes trusted and widely accepted. One who applies a cryptographic scheme should be aware of all its possible vulnerabilities – either algorithmic or implementation – and must know the exact level of security it provides. Generally, this information is only available for well studied / standardized algorithms so it is safer to restrict our selection to the following three signature algorithms that are approved in FIPS 186-2:

- **RSA:** It was first published in 1977. It is widely used in electronic commerce. The details of the algorithm are defined in PKCS #1 (RFC 3447).
- **DSA:** Proposed by NIST in 1991. It uses the ElGamal signature scheme.
- **ECDSA:** Elliptic curve version of DSA. It was proposed in 1997.

Performance analysis

The most important properties of digital signatures with respect to vehicular communications are the following:

- Signature Size
- Public Key Size
- Size of domain parameters (which depends on the Public Key Size)
- Signature Generation Time
- Signature Verification Time

In the literature several benchmarks can be found that compare the signature generation/ verification time of the selected three algorithms (see for example [64], [20], [13], [25], [12]). In the following table, we combine the most important results of those works. As benchmark results differ in the execution platform, in the optimization level of the implementation, and also in the used public/private parameters, our only goal is to set up a relative performance order. Therefore, in each case, we take the time of an RSA-1024 signature verification to be one unit, and normalize the results with this.

	Signature Generation	**Signature Verification**
RSA – 1024	15.00 – 18.14 – 19.44 – 37.5 – 49.49 – 54.11 – 154.54	1
DSA – 1024	8.07 – 80	9.9 – 97.72
ECDSA – 160	0.9 – 0.959 – 1.325 – 2.45 – 6.94	1.382 – 2.29 – 4.162 – 4.46 – 14.01

Timing comparison of RSA, DSA, ECDSA

The large variance in the computational cost is due to the different – and sometimes really small – public exponents used in the measured RSA system. As FIPS 186-3 – which currently exists as a draft version – does not recommend the usage of exponents smaller than 65537, the highest three values can be eliminated from the field corresponding to RSA signature generation.

	Signature Size	**Public Key Size**
RSA - 1024	128 Byte	128 Byte
DSA - 1024	40 Byte	128 Byte
ECDSA - 160	40 Byte	20 Byte

Key size comparison of RSA, DSA, ECDSA

From the two tables, one thing is clearly visible: ECDSA outperforms DSA in every aspect.

Scalability

As the SeVeCom Project considers long term security, we might consider increasing the key size of the above discussed algorithms. For this reason, it is important to examine, how the performance of these cryptosystems changes as their key size increases.

In general, RSA and DSA have the same behavior in terms of scaling. For both of these algorithms, doubling the key size (i.e. using 2048 bit keys) results in about 4-8 times slower operation [45], [18]. At the same time, ECDSA-224 – which provides the same level of security as RSA-2048 – requires less than twice the amount of computational effort of ECDSA-160. In general, the computational cost is proportional to the key size, however, in case of elliptic curve algorithms, the key size grows slower with the provided level of security.

Conclusions

ECDSA is superior to DSA in all aspects. The signature verification of RSA is slightly faster than that of ECDSA – even for larger key sizes, – but the signature generation is one order of magnitude slower. In addition, the communication overhead of ECDSA is significantly smaller, making it a better choice for our needs. Moreover, for higher security levels, ECDSA is clearly better than RSA, as it scales significantly better. Thus, in SeVeCom, we have selected ECDSA as the digital signature scheme used in the proof of concept implementation of our baseline architecture.[FK7]

4.3.4 Privacy

Digital identities and their attributes should be designed and managed within VC systems. These tasks will be undertaken by multiple organizations or authorities, which will be responsible for generating and granting credentials for the VC system entities.

Personal or sensitive data warrant special protection or limited disclosure. Yet, as vehicular networks are systems in the making, decisions by involved parties are necessary to specify both precise requirements and processes for privacy protection. Especially because privacy is a rather broad notion.

One approach, generally applied beyond the VC context, is the use of pseudonyms[FK8]. These identifiers do not carry information about the identity of the system entities, in a way that any two or more pseudonyms cannot be correlated with the same identity and thus entity. An equally general approach is to equip the system with fine-grained control of the entities on the sought level of privacy. Furthermore, to ensure the minimum amount of identity information is disclosed for a specific context and transaction.

At the same time, access control and accountability are indispensable security attributes for VC systems. This means that the above-discussed objective of anonymity, that is, concealing one's identity and avoiding linkability (with respect to a set of observers) of one's actions to its own identity, is not straightforward to achieve. In fact, the two aspects are seemingly contradicting.

Consequently, it appears that full and unconditional anonymity will not be acceptable. This is implied by the current status quo, with strong identification processes for vehicles and users in place. More specific requirements, such as anonymity revocation ('de-anonymization') globally or locally, are also relevant to VC, as it is almost certain that multiple administrative authorities will co-exist.

The system should enable different entities to obtain multiple credentials, perhaps from different organizations, to support the wide range of envisioned VANET functionality. Yet, the system should prevent users from sharing their credentials, either by passing them among themselves or 'presenting' them so that a third party is misled that the credentials belong to the same entity.

A number of existing or under development techniques for privacy protection can contribute towards the above objectives. In other words, the problem at hand for privacy enhancing technologies in VC has similarities, in terms of requirements and characteristics, with existing or under-development,

beyond the VC context, technologies. However, what is more interesting are the 'differences' due to considerations that are specific, if not unique, to VC systems. At first, clearly, VC systems will not be merely another wireless technology to access the Internet (even though this will be supported as well), but a much more complex system that enables applications specific to the VC mission.

VC systems are not necessarily user-centric. Rather, non-human entities, vehicles, and most important, vehicles owned or operated by private parties, will be multiply identifiable and play a central role. One could view the vehicle as the user, yet what remains as a difference is the level of automation that the VC systems will require.

The significance of the vehicle role is mostly due to established administrative processes we discussed in Section 4.3.1.1. Furthermore, not only the vehicle itself but the operational condition of any of its individual subsystems (sensory or mechanical) may be of interest and necessary to be identifiable.

Robustness, as well as and liability and accountability are important in our context.

The VANET communication pattern is an important distinguishing factor: frequent, if not continuous, vehicle to vehicle communication. Communication in VANET will often, if not mostly, be not of transactional nature. Nodes will transmit data that are not addressed to a particular node, or in other words, communication will not be unicast, with two-party protocols. Instead, messages will be mostly 'floating' across the network, i.e., broad- multi- or any-casted, with destinations defined in terms of context- or node-specific attributes (e.g., location, or node characteristics). More 'traditional' types of communication are surely possible and expected; the actual fraction of the overall traffic can only be determined once a set of applications are at the (pre-)deployment phase.

Such VC-specific communication, nonetheless, is at a relatively high rate; some representative widely accepted value: at least one message generated per node every 200 or 300 milliseconds. Depending on the density of the network, and the area across which each such message propagates, a multiple number of messages will need to be validated at each node. What is important is the network overhead due to the cryptographic mechanisms, especially if anonymity is supported. Moreover, the processing overhead can be a significant issue.

To illustrate this, we consider for the sake of an example, the Idemix system: the showing of a credential with all optimizations mentioned by the authors, not implemented at the time of [7], for the system running at a Pentium III, needs a running time of 2.5 seconds. This is roughly a period of time during which at least 12 messages should be transmitted. Of course, further application-specific optimizations may be possible, or somewhat more powerful on-board platforms may be used. This back of the envelope calculation should not be perceived as any sort of criticism, but it solely points out the importance of processing overhead in the context of VC communications (which can be often time-critical).

Finally, regarding communication, VC mandates that a significant fraction of the total traffic, namely safety messages, is frequent and periodic. However, it is not at the discretion of the user/owner of the device to stop or enable it. Furthermore, in contrast with approaches for ubiquitous computing, the user will not elect but will by default engage in context-rich (including, for example, the sender's/receivers' coordinates) communication.

What is most important is that vehicle-to-vehicle communications will call for anonymity as well as security (e.g., authentication). Moreover, anonymity appears as a pre-requisite for the channel communication; in other words, achieving anonymity during a transaction is not meaningful if network communication allows a vehicle to be tracked otherwise.

4.3.4.1 Problem Analysis

Privacy and Anonymity: Vehicular communication systems should not disclose or allow inferences on the personal and private information of their users. This being a very general statement and a requirement within the broader area of information hiding, we state a narrower requirement within the vehicular network context: anonymity. Note that confidentiality is a different requirement.

We require anonymity for the actions (e.g., messages, transactions) of the vehicular network entities, which we denote as nodes, with respect to a set of observers. At minimum, any of the observers should not be able to learn if a node performed or will perform in the future a specific action,

assuming that the node performs the action. Such a definition does not, however, guarantee that it is impossible for the observer to infer, with relatively high probability, the identity of the node that performs the action in question.

To prevent such inferences, stronger anonymity requirements would be necessary: nodes should be almost equally likely to have performed an action, or have strong probabilistic anonymity, with the probabilities, as far an observer is concerned, being equal for any node [34]. Or, without considering probabilities, require full anonymity: an action α performed by a node x could have been performed, as far as the observer is concerned, by any other node in the system.

The definition of anonymity depends on what is the set of the VC system entities. Or, in fact, whether entities are partitioned into a number of subsets, for administrative reasons. This implies that the anonymity requirement needs to be modified accordingly. For example, if two non-overlapping subsets A and B existed, a node x registered with/belonging to A remains anonymous as long as x and any other node y also in A are equally likely to have performed action α. However, it may be trivial to infer that any node z in B did not and will not perform action α.

Anonymity requirements could be refined further, for example, by considering the nature and capabilities of the observers. For example, observers could share information in different manners [44] in an attempt to either learn that a node x performed or is more likely than other nodes to have performed action α. Moreover, it is possible that anonymity is not a requirement with respect to special set of observers, due to a different system requirement we discuss below. Similarly, anonymity may not be a reasonable requirement for all entities of the vehicular communications system.

Liability Identification: Users of vehicles are liable for their deliberate or accidental actions that disrupt the operation of other nodes, or the transportation system. The vehicular network should provide information that identifies or assists the attribution of liability.

This is a requirement that largely follows from the current practice in transportation systems. However, liability identification implies that anonymity would need to be paired with the option to learn or essentially recover the node's identity if necessary. Specifying the type of observer (e.g., a public authority) vested with the power to do so depends on the actual scheme.

Accountability, and eventually liability, of the vehicles and their drivers is required. Vehicular communication is envisioned as an excellent opportunity to obtain hard-to-refute data that can assist legal investigations (e.g., in the case of accidents). This implies that, to begin with, unambiguous identification of the vehicles as sources of messages should be possible. Moreover, context-specific information, such as coordinates, time intervals, and associated vehicles, should be possible to extract or reconstruct. But such requirements raise even stronger privacy concerns. This is even more so when drivers' biometrics are considered: Biometrics, useful for enhancing vehicle access and control methods, are highly private and unique data cannot be reset or reassigned.

Related Mobile and Wireless Networking Technologies

Considering identity management and privacy protection, it can be useful to look at standardized wireless communication technologies. At first, we take cellular networks and GSM as an example. There are two forms of IDs in GSM; the first one being the International Mobile Subscriber Identity (IMSI), which identifies the subscriber and is stored in the SIM card. The cell-phone providers keep a database, the so called Home Location Register (HLR) where this IMSI is connected to the subscriber data. Second, there is the International Mobile Equipment Identity (IMEI), which uniquely identifies the GSM equipment. Similarly to the HLR a provider keeps an Equipment Identity Register (EIR) where the IMEIs of banned or monitored mobile phones are stored.

All the identity management within a network is completely managed by the provider, including authentication and revocation. In case of roaming between providers, they grant access to their HLR so authentication can take place. In cellular networks, the mobile nodes only attach and authenticate with the base stations of own or foreign providers (in case of roaming). Therefore authentication and especially generation and resolution of pseudonyms are straight-forward, the base station plus core network is considered to be trusted. This is not the case in VANETs, where cars communicate with each other, or with infrastructure provided by multiple organizations that may not all be considered trusted.

In order to protect privacy, there is a form of pseudonyms, the so called Temporary Mobile Subscriber Identity (TMSI). It is assigned to a mobile device as soon as it connects to a Location Area and used thereafter instead of the IMSI. This should prevent tracking of devices. Of course if an attacker manages to eavesdrop on the initial handshake, it will be able to track the device by its TMSI later on. Mechanisms like the IMSI Catcher also show the vulnerabilities and concept failures of the system.

To probe further, we consider the Wireless LANs according to IEEE 802.11. There are no identities in the core WLAN standard itself, perhaps with the exception of the unique MAC addresses used. Instead there is the option of using a shared key for accessing the network.

The IEEE 802.1x/802.11i additional mechanisms, when used, provide a standard authentication mechanism with the access point, using the credentials for authentication. The credentials are typically managed in one or more radius servers that check the authentication credentials. However, these servers are not expected to be available online in VANET scenarios.

There is no real mechanism for privacy protection in WLANs, as MAC addresses are always sent in the clear. However, at least IEEE 802.1x/EAP-TLS secures the authentication dialogue, so the credentials cannot be eavesdropped.

Finally, a number of approaches have been proposed for generic MANET, which are neither standardized nor target necessarily specific applications. For example, the instantiation of certification authorities in a distributed manner, with network nodes acting as CA servers, has been proposed. However, literature on MANET has largely neglected the question of identity management.

The development of VANET, with a more precise application context, not only allows posing specific questions on identity management but also move towards providing answers. Furthermore, requirements on anonymity and privacy protection, which were largely not investigated in the context of MANET, can be set more precisely.

Architecture components

To protect sensitive data, processes and policies for privacy protection should be defined. In particular, minimum private information disclosure on a need-basis only should be a basic guideline. For example, if the sought information is the possession or not of a driver's license, the user should not exhibit anything more than that; home address or date of birth would be irrelevant. Accordingly, in the VC context, if the sought information is whether a vehicle passed a technical check, then this is all that should be disclosed; not the Vehicle Identification Number or the registrant's home address. More general, fine-grained control mechanisms are necessary, to allow system entities to regulate private information disclosure.

As accountability is needed, authentication is also necessary. This means that, assuming asymmetric cryptography, a certificate must be provided to any node verifying the validity of a signature. However, each time a message is signed with the same private key all such messages can be linked to the same certificate. Granted, the certificate may attest to specific attributes, that is, a partial identity. Nonetheless, messages and transactions can be linked to a specific certificate. As hiding or obfuscating the location information that is expected to be carried by a significant fraction of the V2V messages (e.g., safety) is not really an option (many such applications rely on the availability of high-quality location information), hiding the identity of the sender is the choice made here.

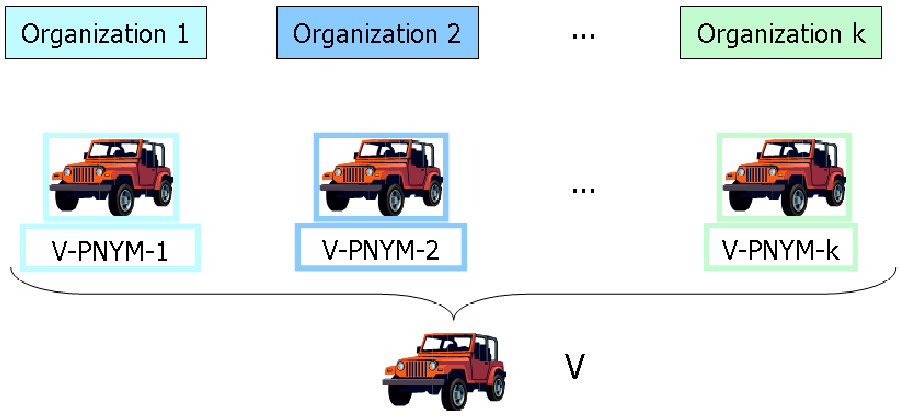

Figure 4-6: Multiple pseudonyms, each relevant to different organizations (verifiers)

Making one step in that direction, all identifying information can be removed from the credentials. This is essentially the *digital pseudonym* concept first introduced by [8]. It can be particularly useful for example when only authentication is necessary, rather than "showing" specific attributes (e.g., generating a signature with the private key corresponding to a attribute certificate). Now, the key and its use convey no identifying information. Nonetheless, again, all uses of the same pseudonym (public key) can still be linked to each other.

A further refinement can be achieved by having distinct pseudonyms. This is achieved by partitioning the identity into multiple partial identities (pseudonyms) each associated with a subset of attributes, or completely devoid of any identifying information. Similarly, a different pseudonym can correspond to a different verifier. For example, each service provider in the context of VC can equip a client-vehicle with its own credential; then, depending on the type of transaction or even the application, vehicles utilize the corresponding pseudonym. The benefit of such an approach is that the same node can be involved in multiple transactions with distinct organizations (verifiers), without an observer being able to link those to the node. This is illustrated in Figure 4-6, where node V has k pseudonyms that cannot be linked to each other.

Multiple uses of the same pseudonym, namely, V_i, could still be linked to each other. This could be more so, if one of the pseudonyms as discussed so far were meant to authenticate messages to be received not by a single entity but all other vehicles. This would be the case for safety messages.

To further enhance the anonymity of nodes, each vehicle can be equipped with a set of pseudonyms PS, each used for a limited period of time. By alternating among the available pseudonyms, actions of the node can be linked to each other only within the period of using a particular pseudonym. Clearly, the pseudonyms should be generated so that they cannot be linked to each other.

Pseudonyms are preloaded by an authority and are periodically renewed after all the keys have been used or their lifetimes have expired. This renewal can be done during the periodic vehicle checkup (for example, yearly) or on-line if the appropriate facility is available. One authority should maintain the ability to resolve pseudonyms to long-term identities. Well-defined policies on the conditions that warrant (anonymity) revocation or pseudonym resolution are necessary.

A vehicle may be tracked despite of regular pseudonym changes because of certain circumstances.

For instance, if the car changes its pseudonym while very few other vehicles are around, linking old and new pseudonym is rather simple by tracing its trajectory using beacons. Similarly, if a car uses changing pseudonyms daily and is parked on the same reserved parking slot each day, the pseudonyms can also be related easily.

Changing the pseudonym on one communication layer does not make sense if protocols on other, non-encrypted layers also use identifiers. In this case, node pseudonyms could be linked by the identifiers of other communication layers. So, changing pseudonyms must be coordinated between layers.

On the one hand, changing the pseudonym only once every night (while the car is parked) has surely no significant influence on communication performance or on-going sessions. On the other hand, changing the pseudonym every 10 milliseconds may increase privacy protection, but it will surely render most communication impractical[FK10][43]: communication towards that specific node will be hard, if not impossible, or at best at a very high communication overhead (assuming the system calls for maintaining end-to-end connectivity or node reachability).

4.3.4.2 Research Contribution

It follows that different methods for changing or alternating among pseudonyms are necessary, in order to reduce the likelihood that pseudonyms used by a node are linked. Next, we discuss two such approaches: (i) change of pseudonyms based on vehicle velocity and eavesdropper placement, and (i) Mix-Zones for VANET.

Velocity-based pseudonym change

In order to preserve the driver's anonymity the key changing algorithm can adapt to the vehicle speed and take into account key correlation by the attacker. In a typical tracking scenario, the attacker controls stationary base stations separated by a distance d_{att} and captures all the received safety messages; he can later use these data (including the public keys) to illegally track vehicles. In addition, the attacker can correlate two keys if the sender moves at a constant speed in the same direction and on the same lane between two observation points (e.g., given the initial position of the target, the attacker can predict its position in the future and confirm this prediction if a message is received at the next observation point with correct predicted speed and position); this is typical of a highway scenario.

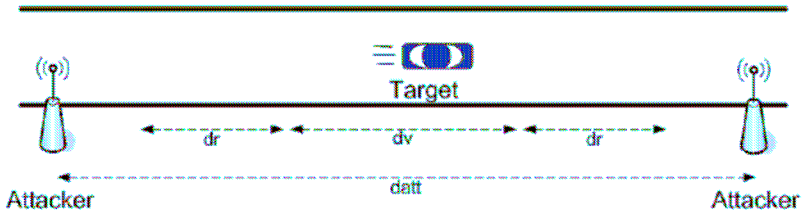

Figure 4-7: To uncover the identity of its targets, the attacker leverages on key correlation and the target's transmission range

Let us assume that the speed of target V is v_t, its transmission range is dr, and dv the distance over which a vehicle does not change its speed and lane (the vulnerability window with respect to the correlation of keys), as in Figure 4-7. The vehicle's anonymity is vulnerable over a distance equal to $d_v + 2d_r$. This means that it is not worth changing the key over smaller distances because an observer can correlate keys with high probability. Thus, the minimum key changing interval T_{key} can be determined as $\min(Tkey) = d_v + 2d_r v_t$ seconds

But if $d_{att} > dv + 2dr$, V can avoid being tracked by changing its key, as long as it does not use the same key for a distance equal to or longer than d_{att}. This in turn defines the upper bound on the key changing interval $\max(Tkey) = d_{att}\ v_t$ seconds. Since V does not know d_{att}, but knows d_r and d_v, it can choose a value of T_{key} that is a little larger than min(Tkey). If we denote by r_m the message rate, one key should be used for at most: $N_{msg} = [r_m\ T_{key}]$ messages.

For example, assume d_{att} = 2 km, r_m = 3.33 msg/s (1 message every 300 ms), d_v = 30

s v_t (i.e., V does not change its lane and speed during 30 s), dr = 10 s v_t (according to

DSRC, the transmission range is equal to the distance travelled in 10 s at the current speed), and v_t = 100 km/h. Then $\min(T_{key})$ = 50 s and $\max(T_{key})$ = 72 s. V can choose T_{key} to be 55 s; as a result, N_{msg} = 184 messages.

Mix-Zones for Vehicular Communications

Besides the expected benefits of vehicular communication, it also have some potential drawbacks. In particular, many envisioned safety related applications require that the vehicles continuously broadcast their current position and speed in so called \emph{heart beat} messages. This allows the vehicles to predict the movement of other nearby vehicles and to warn the drivers if a hazardous situation is about to occur. While this can certainly be advantageous, an undesirable side effect is that it makes it easier to track the physical location of the vehicles just by eavesdropping these heart beat messages.

One approach to solve this problem is that the vehicles broadcast their messages under pseudonyms that they change with some frequency. The change of a pseudonym means that the vehicle changes all of its physical and logical addresses at the same time. Indeed, in most of the applications, the important thing is to let other vehicles know that there is a vehicle at a given position moving with a given speed, but it is not really important which particular vehicle it is. Thus, using pseudonyms is just as good as using real identifiers as far as the functionality of the applications is concerned. Obviously, these pseudonyms must be generated in such a way that a new pseudonym cannot be directly linked to previously used pseudonyms of the same vehicle.

Unfortunately, changing pseudonyms is largely ineffective against a global eavesdropper that can hear all communications in the network. Such an adversary can predict the movement of the vehicles based on the position and speed information in the heart beat messages, and use this prediction to link different pseudonyms of the same vehicle together with high probability.

On the other hand, the assumption that the adversary can eavesdrop all communications in the network is a very strong one. In practice, it is more reasonable to assume that the adversary can monitor the communications only at a limited number of places and only in a limited range.

This problem can be modeled using mix zones (such a location which is not under the control of the adversary, so the vehicles can mix their identities). This analysis better fits to Deliverable 5.2, so there can be found how much private information such a local attacker can gain, and how can this privacy breach be modeled and measured.

4.4 Long Term Research Areas

This section provides analysis work carried out in SeVeCom long term research area.

4.4.1 In-Vehicle Intrusion Detection

Future vehicles will be open systems with multiple networks, different types of devices and with various interfaces (a typical scenario is depicted in Figure 4-8) Therefore the vehicle itself will be the target of multiple types of attacks. A basic requirement for the envisioned applications is the validity of the transmitted information. Therefore it must be ensured that no invalid data is sent to the network. For instance, by tricking vehicle sensors, a warning message could be automatically generated and sent out which is actually not valid. So both the in-vehicle system needs to be able to detect intrusions from the wireless network and the network should be kept free of invalid messages due to malfunction of or attack on in-vehicle systems.

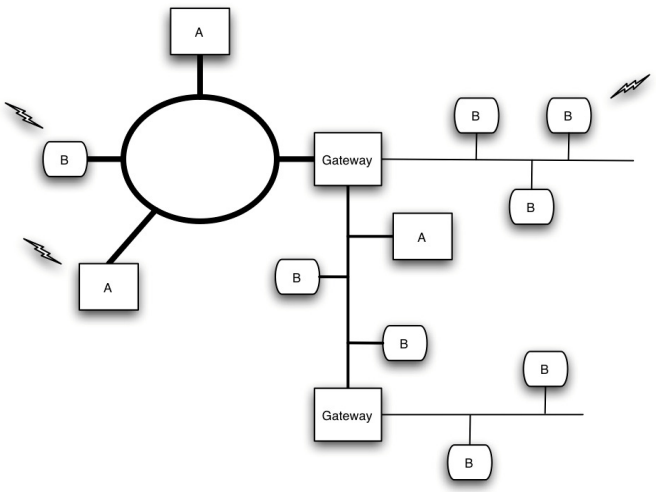

Figure 4-8: Typical in-vehicle System

In addition to the traditional security approaches, intrusion detection systems (IDS) seem to be a promising mechanism to improve the security of the vehicle IT system. Intrusion detection in its most common sense deals with the detection of and the reaction on malicious or fraudulent activities. In contrast to proactive security mechanisms that are intended to prevent misbehaviour in advance, intrusion detection systems react on such activities. Besides the potential to detect malicious behaviour that can not be prevented proactively, intrusion detection mechanisms also offer the possibility to detect attacks that are presently unknown. These reactive mechanisms are especially important in the automotive environment, because of the long lifecycle of a vehicle - many types of attacks will be unknown at the time when the system is designed and produced - and limited connectivity of vehicles - frequent or regular updates (software or/and configuration) - traditional proactive mechanisms are not possible.

The tasks of an in-vehicle IDS are:

- Monitoring

  This includes investigating data flows, calls of system units, watching system state etc.. Monitoring also includes the sending of probes into the network [FK11]to check properties or control system integrity.

- Analysis

  This covers the continuous evaluation as well as the analysis of logged data.

- Reaction

  This covers the logging of data, the alerting if security relevant incidents were detected and the recovery to reset the system in a known secure state. Alerting and recovery are especially important in the in-vehicle environment, because interaction with the driver should be prevented and recovery should be done as autonomously as possible or at least postponed until the next stop.

4.4.1.1 Existing approaches

Traditionally IDS are classified with respect to the observation level:

- host-based: to detect attacks against the host on which the IDS is running
- network-based: to detect attacks against a (sub-) network
- protocol-based: to detect attacks against a given communication protocol
- hybrid: to detect attacks of several types

or with respect to the detection technique

- Knowledge[FK12] based: detection is based on known misuse patterns or attack signatures

  This approach provides for a high accuracy of detection (i.e., low amount of false positives). Disadvantageous is the incapability to detect new types of attacks and the need for regular updates of the knowledge base.

- Anomaly detection: here the IDS has information about the normal behavior of the system

  The advantage of this approach is possibility to detect previously unknown attacks or modifications of existing ones. The disadvantage is the large amount of false positives and problems with the specification of the normal behavior of the system.

- Hybrid approaches: the IDS includes mechanisms of both technologies

Intrusion detection is a rapidly growing field and for each of the classes a large number of systems both Open Source (e.g. Snort [59]) and commercial systems (e.g. Cisco [58], IBM [57], Checkpoint [56]) exist.

Standardisation efforts include:

- ISO Technical Report to IDS [60]
- Common Vulnerabilities and Exposures (CVE) [61] of MITRE
- Common Intrusion Detection Framework Project [62] of DARPA
- Intrusion Detection Working Group of IETF [63]

4.4.1.2 Research Challenges

The situation in a vehicle is characterized by:

- Performance constraints

  Some of the tasks performed by the in-vehicle system (especially the functions related to driving) are safety critical (soft real-time mode). Therefore performance is an important requirement. On the other hand, the in-vehicle network has only a limited number of nodes that can be used as

sensors and their computational power is limited. Additionally it is important that the IDS network traffic does not interfere with the regular traffic.

- Different in-vehicle networks

  The in-vehicle system consists of different networks (e.g., CAN, MOST, Flexray, wireless communication with passenger). The IDS should monitor the entire system and detect attacks that involve several nodes from different networks.

- Embedded devices

  Embedded devices generally do not support for deploying an IDS sensor. Thus monitoring of embedded devices requires special techniques

- Autonomous operation

  In some situations the IDS can not alert the driver (e.g. in a difficult traffic situation), and must solve critical issues by itself, delaying (blocking) critical issues till the right time. This means the IDS must include a decision component that works autonomously or at least semi-autonomously.[FK13]

Compared to a IT network scenario, the in-vehicle environment has additional characteristics, which are not covered by the existing approaches and therefore require either the adaptation of existing IDS or the development of new approaches.

4.4.2 Malfunction Detection and Data Consistency

4.4.2.1 Problem Statement

Previous sections describe various problems and approaches to ensure security of inter-vehicle communication. Traditional network security focuses on data integrity and confidentiality, providing mechanisms like authentication or encryption. However, these traditional mechanisms – that provide a somehow proactive form of security – are only of limited help when VANETs consist of arbitrary vehicles that do not share pre-established trust-relationships or when vehicles inject false data into the network – be it because of malfunctions or maliciously.

Reactive security has a different approach. Instead of preventing attacks, it tries to deal with the effects of false data injected into the network. Using a reactive approach, data consistency is the primary goal and malfunctioning nodes that compromise data consistency should be detected. Following this a reaction can e.g. exclude that node from the network or simply disregard all information sent by that node.

4.4.2.2 Existing approaches to data consistency verification in VANETs

The most relevant paper in the topic of data consistency verification in vehicular ad hoc networks is the paper of Golle et al. [17] In this work the authors classify the attacks, propose a solution and even an analytic framework to analyze the problem. According to the authors, the attacks against VANETs can be classified as follows:

- Attack Nature (attacker lies about itself / attacker lies about other nodes)
- Attack Target (local target / remote target)
- Attack Scope (limited (small subset of nodes is affected) / extended (bigger subset of nodes is affected))
- Attack Impact (undetected / detected / corrected)

The analytical framework presented in the paper is based on models that the nodes maintain. In other words, each communicating node maintains a model about the VANET containing all the knowledge the node has of the VANET. This model consists of a set of rules that are derived from the physical world, e.g., no two nodes can be at the same location in the same time, or nodes usually move slower than 200 km/h. If a node receives messages from other nodes then it can test the validity of these messages by testing the messages against the model. If the message seems to be valid, the node accepts it as trusted and uses it to refine the model. If the message is inconsistent with the model then the node uses heuristics to resolve the conflict. Heuristics can be application specific, and they define a list of possible explanations of the inconsistency. Heuristics also contain precedence relations which can be used to order the possible explanations (e.g., by their probability). For example, an attack with a higher number of malicious nodes is less likely than an attack with only one malicious node. The node usually accepts the most likely explanation for the inconsistency (this principle is called Occam's Razor). This detection and correction mechanism makes the VANET fault tolerant and more robust. This proposed solution can correct even such errors that cannot be detected via cryptographic mechanism. However, it requires a good working model and good heuristics. Regrettably, both of them are hard to define and measure.

In another paper [39], Picconi et al. investigate the problem of validating aggregated data in vehicle-to-vehicle traffic information systems. They assume that every car has a tamper-proof device that is able to carry out secure operations like signing, timestamping and random number generation. Using this tamper-proof device and PKI based authentication, the proposed solution is able to catch an attacker probabilistically. Here, data aggregation only means packet collection, i.e., there is no data aggregation in a mathematical way, but the payload of the messages is collected into a common message, that is, a lot of headers can be thrown away. The problem that they address is that in this scenario an attacker may compromise some elements in the aggregate message.

The main idea of the solution proposed by Picconi et al. relies on the tamper-proof device. This device is responsible to form messages in the following way: the aggregate messages have to inherit a random number and one (or more) replicated part(s) of the aggregate along with their original signatures. Of course, the aggregate messages are signed and timestamped too. When a recipient receives such a message, he can check whether the signature on the message is valid and that the replicated part of the message is equivalent to a part of the aggregate defined by the included random number, and that its signature is valid. This scheme defends against bogus message attacks, where cars lie about other cars in order to generate false traffic scenarios. An attacker can modify some part of the aggregate message but he cannot create the original signature of the replicated messages. Thus, if the random number addresses that part, the attack is detected. Including more replicated parts increases security by increasing the probability of detecting an attack, but is increases the bandwidth needs since this results in a longer message. Thus, there is a trade-off between the security and the bandwidth requirements.

The evaluation of the scheme shows that it overperforms the one base case in terms of *security/bandwidth* where no signatures are deployed and messages travel separately without aggregation. Here, *security* is measured by the probability of detecting an attack. The second base case is when all the messages travel separately but all of them have a signature. This base case is overperformed with the proposed scheme in case of a higher rate of aggregation with moderate number of replicated parts inside the aggregate messages.

Initial analysis performed by SeVeCom partners shows that rather simple detection and reaction mechanisms that focus on the location reported in beacon messages can be extremely effective against position-faking nodes that might otherwise severely affect eSafety applications or position-based routing[FK14] [27].

4.4.2.3 Research challenges

The process of detection and reaction consists of the following components:

1. ***Information gathering***: a vehicle receives information about its context either by communication or by on-board sensors which measure some physical effects in its surrounding.
2. ***Information analysis***: This context information can then either be checked for soundness (comparing if the different information received matches) or it can be verified against some kind of world-model. In any case, we assume some knowledge of the world and compare it with the gathered information. Various methods for information analysis may apply, e.g. self-learning approaches based on neural networks, Markov-models, Bayesian networks, etc.
3. ***Reaction***: Based on the information analysis, other nodes might trigger actions, like ignoring packets or certain information, adapting ratings about other nodes, send out warnings, etc.

The basic research questions arising can be assigned to these components:

1. Information gathering: What kind of information is suitable for efficient and reliable detection in VANETs? How can this information be effectively collected and shared between vehicles without creating too much overhead or delay?
2. Information analysis: Which of the mentioned methods are suitable for information analysis? Are they fast enough in the face of delay-sensitive eSafety applications? Do they require a learning phase in case of encounter of new vehicles?
3. Reaction: How effective are the different reaction mechanisms against attacks? Can they be abused by attackers to e.g. blacklist regular nodes?

These questions need to be analyzed to build effective and secure reactive security mechanisms for VANETs.

4.4.3 Secure Positioning

Any car's location can be determined by using GPS or with the help of on-road infrastructure. Existing positioning and distance estimation techniques assume that vehicles cooperate in determining or

reporting their locations or distances, but some might try to report false distances or locations. Let's look at two solutions for verifying vehicle locations [21].

Each vehicle could have a tamper-proof GPS receiver that registers its location at all times and provides this data to fixed stations or other vehicles in an authentic manner. Fortunately, this doesn't require any additional infrastructure and can be implemented independently in each vehicle. However, one drawback is its availability in urban environments: buildings, bridges, or tunnels often block GPS signals. There are also other well-known weaknesses. The most serious problem with this approach is that GPS-based systems are vulnerable to several different kinds of attack, including blocking, jamming, spoofing, and physical attacks. Moreover, relatively unsophisticated adversaries can successfully execute them. The most dangerous attack involves fooling the GPS receiver with a GPS satellite simulator, which produces fake satellite radio signals that are stronger than legitimate ones. Such simulators are routinely used to test new GPS products.[FK15]

One solution for verifying vehicle location is based on roadside infrastructure and uses distance bounding and multilateration. (Distance bounding guarantees that the distance is no greater than a certain value; multilateration is the same operation in several dimensions.) This approach removes the need for tamper-proof hardware, but requires the installation of a set of base stations controlled by a central authority. The infrastructure covers an area of interest, such as specific roads or city blocks, and can verify vehicle locations in two or three dimensions.

Verifiable multilateration works as follows: Four verifying base stations with known locations perform distance bounding to the vehicle, the results of which give them four upper bounds on distance from the vehicle. If the verifiers can uniquely compute the vehicle's location using these distance bounds, and if this location falls into the triangular pyramid formed between the verifiers, then they conclude that the vehicle's location is correct. Equivalently, only three verifiers are needed to verify the vehicle's location in two dimensions; the verifiers still consider the car's location correct if they can be uniquely computed and if it falls in the triangle formed between them.

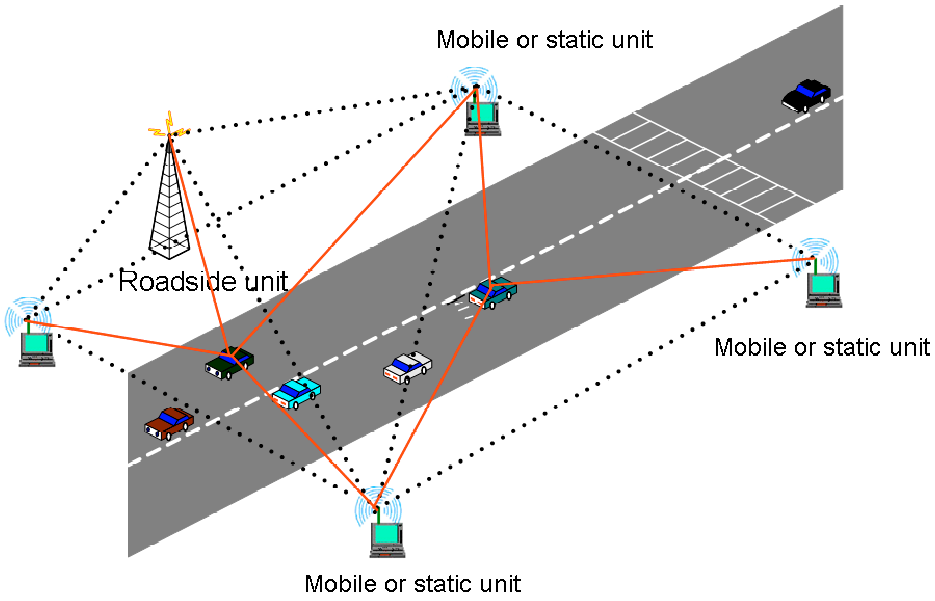

Figure 4-9: Illustration of multi-lateration.

Figure 4-9 shows an example of verifiable multilateration. The intuition behind the technique is that a vehicle might try to cheat about its location. As we mentioned earlier, the vehicle can only pretend that it is further from the verifier than it really is because of the distance bounding [5] property. However, if it increases the measured distance to one of the verifiers, it would need to prove that at least one of these distances is shorter than it actually is, to keep its claimed location consistent with the increased distance. This property holds only if the claimed location is within the triangular pyramid formed by the verifiers: if an object is located within the pyramid and it moves to a different location within the pyramid, it will certainly reduce its distance to at least one of the pyramid vertices. The same holds in two dimensions.

Other approaches analyzed within SeVeCom try to work without additional infrastructure and rely on the use of various simple heuristics to detect and disregard faked position claims by other vehicles[FK16] [27]. These have also shown to be surprisingly effective given the minimum effort required.

4.4.4 Secure User Interface

Another question that will be investigated in SeVeCom will be how an optimal user interface to security mechanisms will look like. There is a simple answer to this question: the optimal user interface for security mechanisms is simply not existing. Security functionality should work autonomously and not distract the user from its original task. This is true for ordinary desktop applications and it is especially true for in-vehicle interfaces. Drivers should not be bothered with decisions on certificate validity or be alerted of dropped packets during driving.

First, driving is a complex task that needs concentration, so distraction should be avoided. Second, many drivers have absolutely no background in computer technology let alone security. So bothering them with security decisions will only annoy them and they usually have no knowledge to make well-founded decisions (e.g. on certificate validity).

However there are a few exceptions to this rule:

- Users may change certain settings of the security system which can safely be done while the car is parked.
- In certain conditions working without any user interaction means that the system can only make conservative decisions or risk attacks. Depending on the driving situation, user interaction might be acceptable.
- In some situations, the security system may rate the trustworthiness of information displayed to the driver. It might be useful to also present this trust information to the driver, so he can decide himself what to do with the presented information.

In all this cases two questions have to be answered:

- When should driver interaction happen and when should it be avoided?
- If the answer to the first question is yes, what is the optimal way to perform this interaction?

SeVeCom will investigate these questions and give some hints and guidelines to user interface designers that usually are not considering security mechanisms in their work.

4.4.4.1 Configuration of the Security Systems

Some components of the security system may need to be configurable from the user interface. One obvious candidate is configuration of privacy settings. People have varying privacy requirements and the right to enforce these requirements in technical systems like IVC. The Article 29 working group in the European Union lately published a statement regarding privacy in the eCall service. They explicitely stated that "the Article 29 Working Party privileges the voluntary approach for the introduction of the eCall service" and "a user-friendly solution taking care of selfdetermination of car users by introducing the technical possibility to switch off/on eCall on a case-to-case basis must be introduced" [46]. So user-empowerment – to be able to determine their privacy settings on their own –seems to be an important issue. However, the user interface needs to be designed in such a way that the drivers are actually able to understand what they are configuring and to easily express their wishes using the restricted IO capabilities in cars. There has been some research regarding user interfaces for privacy in regular systems [11], but the implications of IVC privacy and car user-interfaces have not been addressed so far.

4.4.4.2 Security Decisions and Trust Ratings

Consider the following situation: the Intrusion Detection System determines that a received warning message can be trusted only with a confidentiality of 75.2%. What to do? Display the message or drop it? If you drop a valid message, the driver may miss the warning and an accident may result. If you display faked messages, the driver will get annoyed and ignore warnings altogether. Or what happens if the car receives a Car-to-Car message from another driver, but the certificate has expired or is invalid. Display the message or not?

In all these cases, it might be an option to display the information, but also alert the driver that something is suspicious about this information, e.g. by changing colors, etc.

4.4.4.3 Solution guidelines

As a first result of these observations, some first guidelines on designing security-related user interfaces may be given. However, these guidelines will have to be validated and extended by future research in this area.

Guideline 1: Avoid interaction

Users do not want to get bothered with security. So avoid interaction if possible.

Guideline 2: Design unobtrusive interfaces

Interfaces should not interfere with normal driving operation.

Guideline 3: Design adaptive UIs

Interaction should be adapted to the attention level of the driver, the current driving situation, and the severity of the security event to be communicated.

Guideline 4: Delay interaction to a later time

If interaction with the driver would violate guideline 2, interaction may be delayed to a later point in time, when e.g. the driving situation is more relaxed or the car is parked.

Guideline 5: Give users unobtrusive indications of security status

The system may give information on the security status in unobtrusive ways, e.g. by changing the color of displayed warnings according to the trustworthiness of the information.

4.5 Implementation Analysis

This section addresses the following problems:

- SeVeCom implementation must be integrated with the vehicular communication systems used by eSafety projects
- Security solutions implementations in the future will also need to be flexibly integrated in the deployed vehicular communication systems.

To this end, we need a software architecture which

- is ***modular***, so different security mechanisms can be implemented in separate and independent modules.
- is ***configurable***, so the applications can determine what modules are necessary and how they should be configured.
- can be ***flexibly integrated*** into the overall communication architecture and allows the ***configuration of communication mechanisms*** according to security needs.
- allows the ***inspection***, ***filtering***, and ***manipulation*** of data units.

How can that be achieved in SeVeCom?

For supporting ***modularity*** we propose a plugin architecture, where different security mechanisms are implemented as plugins that can dynamically be loaded or unloaded. Alternatively, this can also be realized as different classes or libraries, if a more static implementation approach is desired.

For ***configurability***, we envision a system where applications have to declare their security requirements to the system.

```
<?xml version="1.0"?>
<SecurityDeclaration xmlns ="http://www.sevecom.org/security-declaration-language/">
  <Application name="Vehicle-based Road Condition Warning">
    <Type>eSafetyApplication"</Type>
    <SecurityRequirement module="AttributeAuthentication">
      <nodeType>Vehicle</nodeType>
    </SecurityRequirement>
    <SecurityRequirement module="Privacy">
      <sv:idPrivacy changeInterval="5s"/>
    </SecurityRequirement>
  </Application>
</SecurityDeclaration>
```

Figure 4-10: Example of Security Declaration File

Such a declaration can be realized in form of XML-based configuration files like shown in Figure 4-10.. The example expresses that the application with the well-known name "Vehicle-based Road Condition Warning" is an eSafety application, requires authentication of the attribute that the communication partner is in fact a vehicle, and for privacy reasons the identifiers should be changed at least every 5 seconds.

Other opportunities of having such a configuration mechanisms includes prioritisation of security requirements in case of conflicts of interest, dynamic adaptation of security behaviour to national legislation e.g. in case a car crosses national boundaries, and the easy support of user-configuration to support user empowerment and individual privacy settings.

Finally the security modules need ***integration*** in the communication architecture. It must be able to control the ***configuration*** of the communication mechanisms as far as security is concerned, e.g. regularly modify identifiers in case of privacy requirements. The security modules must be capable of ***inspection***, ***filtering***, and ***manipulation*** of data units. There are several options how to implement the security mechanisms:

1. *Library Approach*: Modules are implemented as libraries that offer the necessary functions. Functions are called from the communication system as appropriate. This would penetrate the code of the communication system with calls to the security mechanism and render it rather unreadable. Additionally, members of both the communication project (e.g. CVIS) and the SeVeCom projects would need to closely interact to correctly instrument the communication system code with security calls. This requires a mutual understanding of what the other's code is actually doing.
2. *Aspect Orient Programming*: in this paradigm, side-functionalities (so called *System-Level-Concerns)* like security are taken out of the main program code and collected in so called aspects. As this is however still a matter of research in Software Engineering we do not consider this to be mature enough to be used in IVC.
3. *Hook-based approach*: based on a common definition of APIs and hooks, the communication system would allow interested software components to register callback functions at defined positions of the communication flow. Additionally an API is required that allows the configuration of the communication mechanisms.

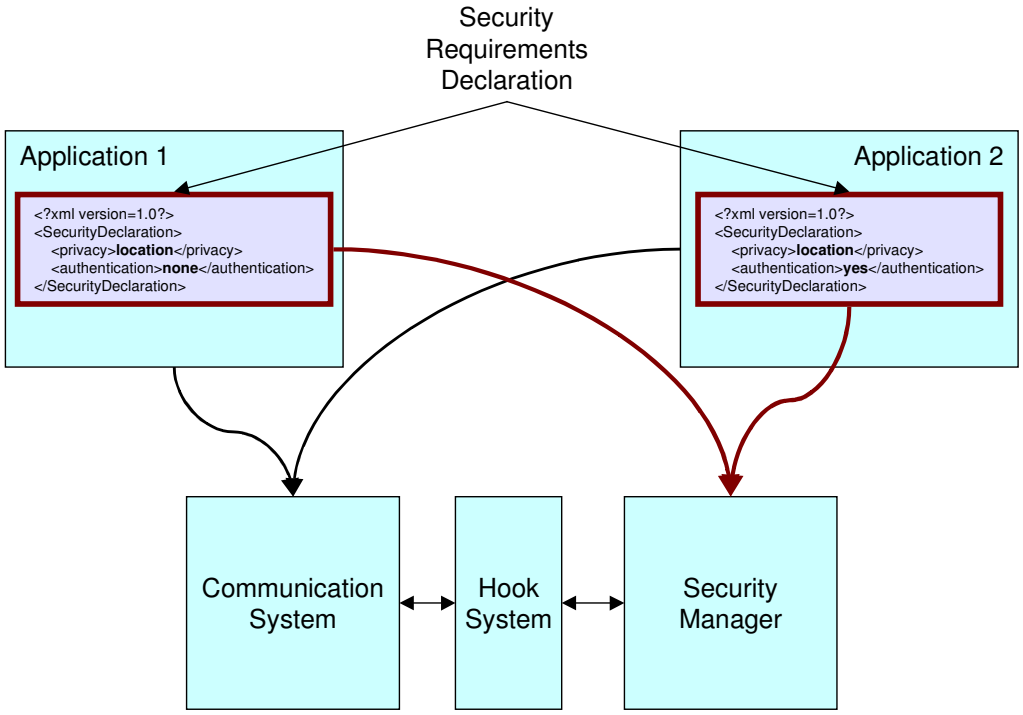

Figure 4-11: Secure Communication Architecture

Option 3 has obvious advantages, so we decided to choose this alternative. Figure 4-11[FK17] shows the overall design of the secure communication architecture with two applications containing the Security Requirements Declarations, the Security-Manager which controls the whole security systems, various instances of security modules that connect to modules in the communication system using the pre-defined hooks.

Details of how SeVeCom is adopting this approach will be given in the next section.

5 SeVeCom Baseline Architecture

5.1 Introduction

The rationale for SeVeCom architecture work is the following:

- Deployment of V2V and V2I in the future is based on communication standards that are not finalised yet. This means that a clear separation must be made between the architecture and technologies below.
- As the V2V and V2I infrastructure is deployed and as applications and services are deployed, security parameters and strengths might evolve.

Consequently, as outlined in Section 3.1.3 approach proposed by SeVeCom is the following:

- Define an abstract security architecture which is future proof, i.e. the abstract security architecture does not change, even though some technology standards will change, or stronger crypto systems will be included. The abstract architecture is based on a number of founding principles.
- The abstract architecture can then be mapped to concrete technology solutions.

The SeVeCom baseline architecture therefore consists of:

- The abstract architecture
- Concrete solutions that will be used in a SeVeCom proof-of-concept implementation

5.2 Architecture Principles

This section identifies the principles of the future-proof architecture for a secure vehicular communication system. In the following table, we describe each principle and explain the rationale for it.

Category	Principle	Description	Rationale
Communication management	Layered based security	Assuming that the communication system will be divided into different layers, the security system will consist of different components that connect to these layers and address the security aspects of each individual layer.	Separation of concern between application and security Technology independence
	Configurable security level	Applications can configure their security level based on individual security needs (insecure, integrity, confidentiality, integrity+confidentiality, privacy …)	Adaptation to application needs
Identity management	Resolvable Pseudonymity	Vehicle identity is replaced by a pseudonym. Higher jurisdiction allows for transcript access.	Ensures privacy as vehicle identity is not revealed, and enables liability attribution.
	Protocol identities control	Pseudonym system has full control of changing the underlying protocol stack's identities Any other identifications (e.g. application identities, upper layer protocol identities) are not transmitted in the clear	Prerequisite for effective pseudonym change and privacy support

Category	Principle	Description	Rationale
	Vehicle has a long term identity	Vehicles possess a life-long unique identity that can be used to identify the vehicle in case of legal dispute	Total anonymity is not considered a desirable property of IVC as certain situations (e.g. accidents) might require access to the actual identity of vehicle and driver
	Configure-ability of pseudonym system	Possibility to parameter the pseudonym system (e.g. in a region, during a period)	Specific policies Infrastructure scale up Individual privacy requirements
	Openness to multiple credential sources	It is possible to have independent set of pseudonyms	Takes into account existence of multiple authorities (multiple business stakeholders, multiple countries, …).
	Eviction of nodes	Management of nodes with unauthorised credentials	Takes care of broken credentials or non-vehicle attackers
Platform	Upgrade-ability of communication system	Open to new communication patterns	Development of IVC is still in the phase of significant discussions. Final communication protocols and patterns are not expected within the project's lifetime.
	Upgrade-ability of crypto system	Possibility to upgrade the crypto system to another one.	As infrastructure grows or cryptanalysis progresses, a stronger security system could be needed
	Security module	Dedicated tamper resistant trusted component	Protection of vehicle cryptographic material and safeguard data usable for liability implication[FK18]
	Hook architecture	Software interface between security part and rest of the platform software implementation	Allows flexible integration of SeVeCom security system into different communication platforms.

5.3 Abstract Architecture: Conceptual View

This section explains the SeVeCom baseline abstract architecture from a conceptual view. It first provides an overview of the conceptual view. It further describes the four modules of the conceptual view, the secure communication module, the identification and trust management module, the privacy management module and the tamper evident security module. Note that one module is only described in the overview (in-car security module). This module relates to longer term research activities of SeVeCom so it is not elaborated further. See section 4.4.1 (In-Vehicle Intrusion Detection) for an analysis report.

5.3.1 Overview

Figure 5-1 shows the security architecture in an abstract and deployment-independent view. It also shows the logic links between the individual security modules. The security modules are logical containers that group components that fall within their domain. These security components then realize specific security functions. The set of security components is not fixed but will evolve over time. There might also be multiple variants of components addressing the same issue in different ways having e.g. different security-overhead trade-offs.

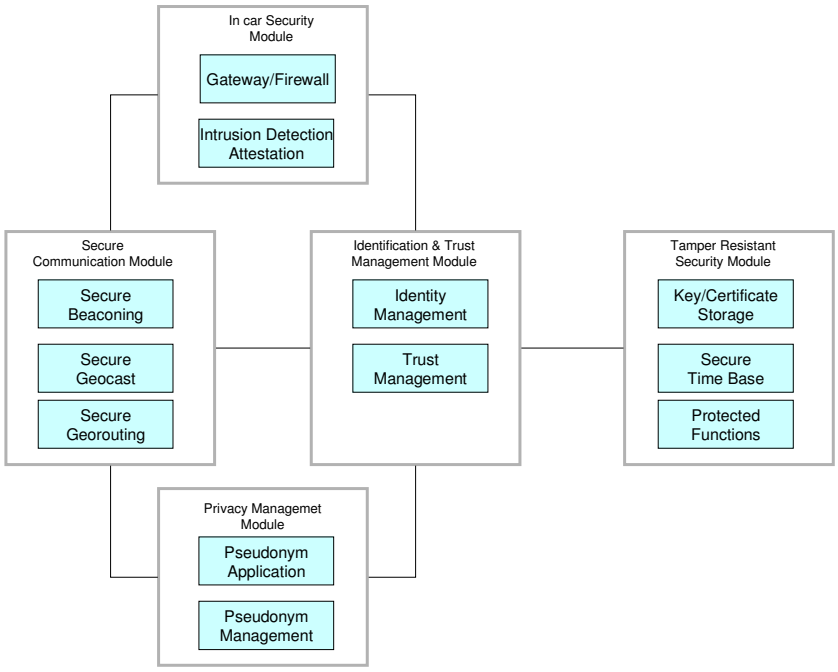

Figure 5-1: Architecture Conceptual View

The five security modules and their components are described below.

1. The **Secure Communication Module** addresses secure communications in vehicular network. There are dedicated security components for different communication patterns. Currently envisioned secure communication components include:
   - *Secure Beaconing* provides the means for the receiver to verify the authenticity and integrity of beacons.
   - *Secure Geocast* assures the reliability and security of Geocast.
   - *Secure Georouting* assures the authenticity and integrity of the route messages, and protects routing against routing attacks (e.g. rerouting, replay, dropping, and forging etc.)
2. The **Identification & Trust Management Module** provides and manages identities and certificates of all entities directly involved in vehicular communications, i.e. vehicles and road-side units.
   - *Identity Management* manages the long-term identifier (similar to an Electronic License Plate (ELP)), and certificates containing vehicular attributes.
   - *Trust Management* describes the backend infrastructure (e.g. a PKI) that provides services like public key registration, certification, and revocation services.
3. The **Privacy Management Module** leverages on pseudonyms (i.e. certified public keys) to assure a certain level of privacy to individual vehicles in vehicular networks. It can be seen as an extension to the identification & trust management module as it modifies the creation and application of identifiers. It is split into the following two components:
   - *Pseudonym Management* generates, stores, and refills the pseudonyms.
   - *Pseudonym Application* provides pseudonyms used in secure communication and decides e.g. when to change pseudonyms.
4. The **In-car Security Module** assures the security of the overall in-car system (e.g. sensors, buses, ECUs etc.) and prevents unauthorized access to critical in-vehicle systems. This module contains the following components:
   - *Gateway/Firewall* protects critical in-vehicle systems from attacks through vehicular communications. It monitors and checks the consistency of data flow between the communication system and in-car systems.
   - *Intrusion Detection/Attestation* detects tampering (intrusion) with in-car systems, and establishes trust relations between different hardware components.
5. The **Tamper Evident Security Module** provides tamper evident hardware for storage and processing of cryptographic material and safeguard data (used for liability implication). If the device's resistance breaks, physical inspection of the device provides convincing evidence of the evidence of the compromise.

- *Key/Certificate Storage* stores the private keys and the vehicle's long-term certificate.
- *Secure Time Base* protects time information source.
- *Protected Functionality* stores functions which has high secure priority, e.g. signature creation.

The logic links depicts the information flow and cooperation among the security modules. A brief description of the links is given below.

- *Identification & Trust Management – Tamper Resistant Security:* Parts of the identification credentials and necessary operations will be stored or implemented in tamper evident hardware, so there is a connection to the tamper evident security module.
- *Identification & Trust Management – In-car Security*: The in-car security module relies on the secure identification & trust management module for making access decisions based on vehicle identities and attributes
- *Secure Communication – In-car Security:* The in-car security module cooperates with the secure communication mechanisms in cases of communication of external nodes with internal systems. E.g. it assures the correctness and integrity of information (e.g. from sensors) that will be communicated into the network and it also protects in-car systems from attack through the V2V/V2I communications.
- *Secure Communication – Identification & Trust Management:* The secure communication module relies on secure identification & trust management, e.g. in order to verify the identity information received from other parties or for accessing the own key material for purpose of signature creation.
- *Secure Communication – Privacy Management:* The privacy Management Module provides pseudonyms to Secure Communication Module and also decides when to change pseudonyms.
- *Privacy Management – Identification & Trust Management:* . The privacy management can be seen as an extension to the identification & trust management module as it extends the creation and application of identifiers.

5.3.2 Secure Communication Module

This section gives some initial ideas on implementation of the individual modules. However, detailed descriptions are left to a later document.

5.3.2.1 Communication pattern notation

In order to specify messages unambiguously, it is necessary to agree to a few notations and conventions. Figure 5-2, e.g., illustrates a Beacon message. It consists of information that is authenticated, namely the message components below the horizontal bar that delimits the Authenticated data. This information can consist of zero or more components that can be either mandatory or optional, depending on the nature of the specified message.

The message contains, next to the data that is being authenticated, also a mandatory field with the integrity-protection information.

Depending on the application and their communication patterns, different message mandatory and optional fields can be included. The concatenation of these fields can be authenticated and/or encrypted. This leads to the following generic message types:

- Insecurely sent message: the payload of the message is encapsulated and broadcast in the clear, without any protection of its integrity or confidentiality.
- Authenticated message: the sender of information authenticates this information before it is broadcast. The actually sent message consists of the data and the information that protects its integrity. The latter can consist of either a digital signature or a message authentication code (MAC). Note that the integrity protection field of information that is sent in an authenticated manner can easily be stripped from that message. If this inherent property of authenticated messages is not acceptable for a particular application, one of the following message types has to be considered.
- Confidential message: if the sender and receiver share a secret key that can be used to protect the confidentiality of information, i.e., to encrypt information, then all privacy-sensitive information of a message can be encrypted under that key. There are cryptographic solutions for each possible case: group encryption if a confidential message has to be accessed by more than one recipient, encryption using a secret session key that was explicitly agreed on over a bidirectional communication channel, e.g., using a key agreement mechanism such as the Station-to-Station protocol, or encryption using a secret session key that can only be calculated by the intended recipient. The latter mechanism is particularly useful if the identity

of the intended recipient is known, but the sender and recipient cannot previously agree on a secret key, e.g., because of the availability of a unidirectional communication channel. Identity-based cryptosystems meet this requirement. Note that protecting the confidentiality of information does not guarantee that the receiver is able to determine who sent the message: anyone can forward an encrypted message without knowing its actual content.

- Securely sent message: with this message type, both the authenticity of the sender of the information and its confidentiality are protected. Each of the following possibilities has its own security properties: (i) the sender authenticates ciphertext, (ii) the sender authenticates the information first and encrypts the authenticated data, or (iii) the sender uses a scheme that provides authenticated encryption.

Note that if the sender knows the intended recipient, it is recommended to send messages of the fourth type. If this is not the case, and they share a bidirectional communications channel, it may be necessary to first establish a shared session key using a well established key agreement protocol, e.g., the Station-to-Station protocol. This protocol consists of up to three steps. The first step consists of an authenticated message which initiates the key agreement protocol. After having processed this first message, the receiver of this message shares a secret session key with the initiator of the protocol. After processing this answer, the two parties share the same key, which the initiator can use further on to send him encrypted content. This protocol also supports the anonymity of one or both parties.

Also note that replay of recorded can only be protected against using a challenge-response mechanism, i.e., using a bidirectional communications channel.

5.3.2.2 Communication Patterns

One basic requirement for all kinds of secure communications in vehicular networks is integrity protection. The basic tool for this is digital signatures which can be used for all kinds of communication patterns. Therefore, all messages will contain a signature calculated by the generating node *V* using its private key *pk(j,V)* corresponding to the *j*-th pseudonym *PK(j,V)* of *V*. $Cert_A\{PK(j,V)\}$ denotes the certificate that includes at least *PK(j,V)*, the validity period of the certificate and the signature of the pseudonym provider *A*. Optionally, the certificate may also include data fields that attest certain attributes to the owner. For simplicity, we omit notation on the pseudonym set. The messages also have a time-stamp- the sender's clock value - and a geo-stamp - the sender's coordinates, at the sending time. This mechanism can be applied to different types of communication patterns, which fall roughly in the following three main categories, as expected in VC systems:

Secure Beaconing

The format of secure beacon message contains basically all the described, generic security fields, like depicted in Figure 5-2.

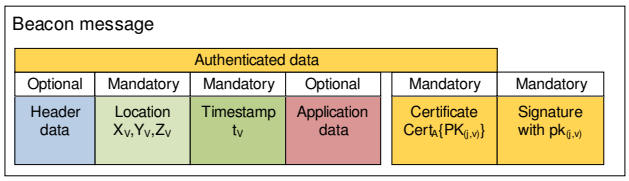

Figure 5-2: Message format of secure beacon messages

By verifying the signature using the public key in the attached certificate, all receivers can be sure that the sender actually sent the message (not another node) and that the message content has not been tampered with. Checking the optional attribute list in the certificate, vehicles can ensure that they are e.g. actually communicating with another vehicle and not an attacker's laptop. By comparing its current time with the timestamp t_V, replaying messages after a defined threshold can be detected as well. The location of the sender is subject to consistency checks, which are able to give a heuristic rating of the correctness of the position claim.

Secure Restricted Flooding/Geocast

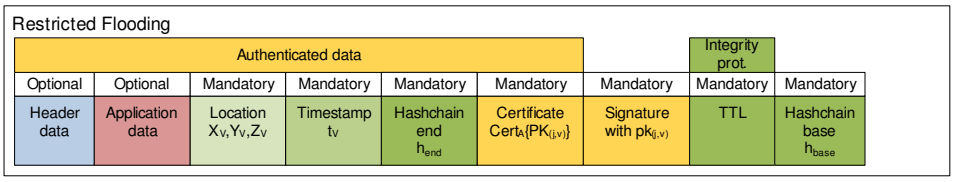

Figure 5-3: Message format for restricted flooding

In contrast to beacons, flooding & geocast messages are distributed in the network over multiple hops. In general, all mechanisms described in the introduction are applied again, which means that receivers can verify authenticity, integrity and freshness of a message. Moreover, the location of the initial sender may be checked. However, this is currently not foreseen.

In addition to the generic mechanisms, flooding requires to secure the time-to-live (TTL) value. This value is used to keep messages in certain area so that they do not spread endlessly in the network. However, because the TTL value has to be decreased at each hop, it cannot be included in the part that is authenticated by the sender's signature. Therefore, we use a hash chain mechanism. The final result of the chain h_{end} is included in the authenticated part, and every hop removes one element of the chain. Then, every receiver is able to verify the TTL by applying the hash function *TTL* times, if the result of $h^{TTL}(h_{base})$ matches the hash chain end h_{end}. By this mechanism, an attacker cannot re-increase the TTL artificially, except he has captured previous chain elements that were used locally on the way of the packet.

The packet format for TTL-restricted flooding is shown in Figure 5-3.

The case for geocast is a little different. While the distribution mechanism is basically the same, the restriction is not defined by a TTL value, but determined by a fixed geographic region. Therefore, the sender can include the restriction in the authenticated part.

Beyond cryptographic means to secure flooding and geocast, a means to thwart denial of service by massive flooding attacks is needed. For that, we intend to introduce rate control that limits the number of packets a node is allowed to send per time period.

Secure Geographic Routing

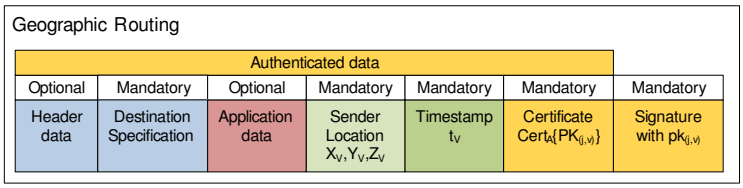

Figure 5-4: Message format for secure geographic routing

The security related format fields of secure geographic routing pretty much consist of the generically required field. The difference is mainly in the additional header fields like the destination position. This can be included in the sender-authenticated part, as it is fixed from the start.

A specific aspect which is vitally important for the routing is the plausibility of neighbour positions. As the geographic routing can rely on the beaconing mechanisms that are in place, it is also mainly the task of the beaconing to rate the plausibility of neighbour positions. The secure geographic routing has to make sure to use these ratings during the forwarding decision.

Secure Gateway Connection

In the case of bidirectional communication with a (gateway) road side unit, the requirement to keep communicated data confidential is relevant. Because encrypting larger amounts of data using the public key of the peer is computationally expensive, a key exchange protocol can be introduced to agree upon a symmetric key for encryption. For this key exchange, we use the principle of authenticated Diffie-Hellman.

In addition to the signature and the certificate as outlined before, the first packet from the originating node O to the destination D contains the required set of DH parameters, i.e. g, p and A, where $A = g^a \ mod \ p$ and a is a randomly chosen secret key of O. In the response, the peer includes $B = g^b \ mod \ p$, where b is a randomly chosen secret key of D. After this "ping-pong" message exchange, both parties can calculate the common shared secret key K: The originator O calculates K by $B^a \ mod \ p$ and at the peer D, $K = A^b \ mod \ p$.

Using the established symmetric secret key K, the peers can encrypt further communications.

[FK21]

5.3.3 Identity & Trust Management Module

The following is assumed :

- a long term identifier is associated with the vehicle
- a PKI is used to allow for the use of certificates that will be used to authenticate vehicle attributes.

The identify and trust management module is in charge of

- managing the vehicle long term identity,
- managing the credentials that will be used to authenticate vehicle attributes,
- verifying and carrying out authentication verification activities.

The Identity and Trust management module implies interactions with a back-end infrastructure and management of certificates that are revoked. We now describe these aspects.

5.3.3.1 Back-end Infrastructure

Drawing from the analogy with existing administrative processes and automotive authorities (e.g., city or state transit authorities), a large number of certification authorities (CAs) will exist. Each of them is responsible for the identity management of all vehicles registered in its region (national territory, district, county, etc.).

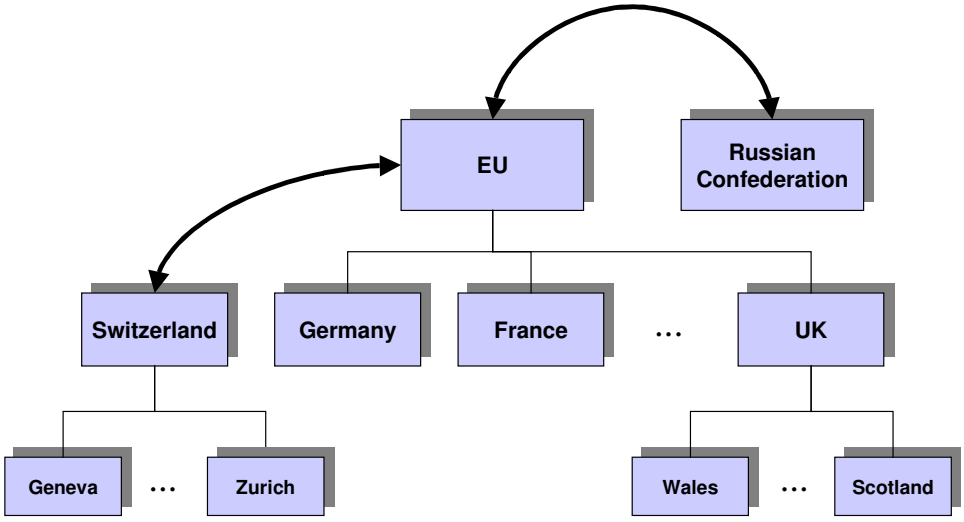

Figure 5-5: Example of Hierarchical Organization and Relations of Certification Authorities

Figure 5-5 illustrates a part of an instantiation of the CAs: an hierarchical structure within each CA and cross-certification among CAs. This way, the deployment of secure vehicular communications could still be handled locally to a great extent. At the same time, vehicles registered with different CAs can communicate securely as soon as they validate the certificate of one CA_A on the public key of CA_B. Various procedures for easily obtaining these cross-certificates can be implemented.

Nodes of the vehicular network are registered with exactly one CA. Each node, vehicle or RSU, has a unique identity *V* and a pair of *private* and *public* cryptographic keys, k_V and K_V , respectively, and is equipped with a certificate $Cert_{CA}\{V,K_V,A_V,T\}$, where A_V is a list of node attributes and *T* the certificate lifetime. The *CA* issues such certificates for all nodes upon registration, and upon expiration of a previously held certificate.

We emphasize that the *CA* manages long-term identities, credentials, and cryptographic keys for vehicles. In contrast to short-lived keys and credentials. The *CA* is also responsible for evicting nodes from the system, if necessary, either for administrative or technical reasons. The interaction of nodes with the *CA* does not need to be continuous, while the roadside infrastructure or other infrastructure-based networks (e.g. cellular) could act as a gateway to the vehicular part of the network or offer an alternative method of connectivity.

5.3.3.2 Credential Revocation

Pseudonyms are bound to the vehicles' long-term identities, with a pseudonymity resolution authority PRA being able to infer this mapping if necessary, for example, for liability attribution. Messages signed by the same vehicle using different pseudonyms can be linked by PRA. In the simplest system configuration, the CA is the pseudonym provider and the pseudonymity resolution authority. Then, it suffices for the CA to maintain a map of pseudonyms to the long-term identity of the vehicle. In general, different solutions with differing properties are possible; for example, the pseudonym to longterm identity mapping could be maintained by the pseudonym provider itself, or the pseudonym provider could maintain evidence of the mapping that only PRA can utilize to resolve the pseudonym.

Beyond pseudonym resolution, a node that is deemed illegitimate (e.g., its registration expired) or malfunctioning can be removed from the network. This is possible by revoking the pseudonyms and the long-term credentials of the node. If the long-term credentials of a node are revoked, the node is evicted but it is not automatically prevented from participating in the VC system operation. This is so because the pseudonyms that the node is equipped with, rather than the long-term credentials, are utilized for communication. However, long-term credentials are used by vehicles to obtain new sets of pseudonyms: nodes use them to establish with the pseudonym provider that they are legitimate members of the system, i.e., registered with a CA.

This implies that one option is to notify directly the pseudonym providers regarding revoked nodes. Or, in other words, place the effect of a node revocation on the pseudonym provider. This way, no communication overhead over the wireless medium is necessary. We identify a trade-off: the more frequent the pseudonym refills. are, the easier the revocation (fewer pseudonyms to revoke), at the expense of higher cost and inferior usability due to frequent executions of the refill protocol. For example, one can imagine a situation when the vehicle fails to obtain new pseudonyms, after having utilized all available valid ones, if the pseudonym provider is unreachable.

Yet, the need to revoke not-already-expired pseudonyms previously provided to a revoked node remains. If pseudonyms are not issued by the CA, coordination of the CA and the pseudonym provider is necessary. Then, revocation will have to take place via the distribution of revocation information across the network. By leveraging on the revocation effect on the side of the pseudonym provider, the size of certificate revocation lists (CRLs) could be reduced.

We provide multiple revocation options tailored to the scale of VC systems. First, a Revocation of the Trusted Component (RTC) protocol, with the CA instructing directly the TC to erase all cryptographic material and acknowledge the cease of operation, and in case RTC does not conclude successfully. Second, revocation through the distribution of compressed certificate revocation lists, namely, the RCCRL protocol, which utilizes RSUs or low-speed broadcast media to distribute the revocation information. The infrastructure acts as a gateway for dissemination of revocation information and the execution of the revocation protocols. The three methods are discussed in [41]. An alternative third approach could be to require that vehicles regularly acquire proofs that their credentials remain valid. Instead of requiring them to download revocation information, vehicles download verifiers from the CA or the pseudonym provider. These verifiers are then included when the certificate is presented to other nodes [32], [23].

5.3.4 Privacy Management Module

5.3.4.1 Pseudonym Management

As a basic guideline, processes and policies for privacy protection should be defined, with minimum private information disclosure on a need-basis, and fine-grained control mechanisms for regulating private information disclosure. Nonetheless, signed messages can be trivially linked to the certificate of the signing node; thus, the removal of all information identifying the user (e.g., driver) from node certificates does make communications anonymous.

We extend this concept first introduced by [9]: we equip each private vehicle with a set of distinct certified public keys that do not provide additional identifying information, denoted as pseudonyms. Instead of using its long-term key pair, a node utilizes the private key corresponding to a pseudonym to sign outgoing messages, and appends the pseudonym to the messages. Messages signed under the same pseudonym (i.e., using the same corresponding private key) can be trivially linked to each other. Yet, as the vehicle changes pseudonyms, linking messages signed under different pseudonyms becomes increasingly hard over time and space.

Figure 5-6 illustrates a pseudonym that has a lifetime and an identifier of the corresponding pseudonym provider A, which is in general an entity distinct from the CA. Note that there may be multiple pseudonym providers, either as independent entities specializing in this task, or as administered by different entities (e.g., various service providers, car manufacturers, highway or city transportation authorities).

PSNYM-Provider ID	PSNYM Lifetime
Public Key	
PSNYM-Provider Signature	

Figure 5-6: Basic Pseudonym Format

Figure 5-7 clues on the concept of periodic vehicle ”refills” with new pseudonyms: a node utilizing pseudonyms out of the i-th set, obtains an (i + 1)-st set of pseudonyms while it can still operate with pseudonyms in the i-th set, and switches to those in the (i + 1)-st once no pseudonym in the i-th can be used. Recall that each pseudonym is used for a period of time which can be determined by various factors. The rate of pseudonym changes determines, along with the frequency of “refills”, the size of the pseudonym set the node should obtain.

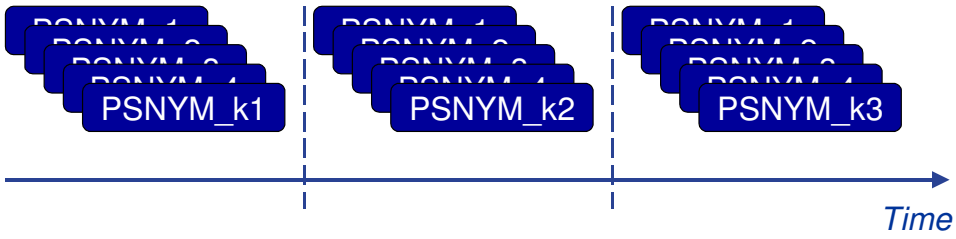

Figure 5-7: Periodic Vehicle “refill” with a New Set of Pseudonyms

5.3.4.2 Pseudonym Application

Figure 5-8 summarizes factors determining when a pseudonym change, and a choice of a pseudonym among possibly multiple available sets, S1; : : : ; Sn, of pseudonyms, should occur. The rate at which a node switches from one pseudonym to another depends on the degree of protection the vehicle seeks, local or system-wide policies, vehicle inputs (e.g., location or velocity), the veri_er of the messages issued (signed) under a specific pseudonym, and other network operation considerations (e.g., communication with an access point through the TCP/IP stack).

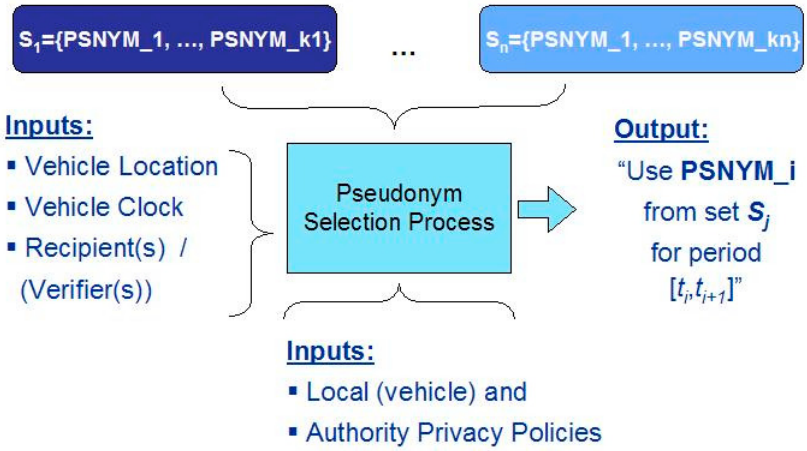

Figure 5-8: Pseudonym changing framework

The change of a pseudonym should be accompanied by a change of the node identifiers used by underlying networking protocols. In particular, this can be the Medium Access Control (MAC), and other identifiers such as IP addresses. If such identifiers do not change along with the pseudonym, messages generated by a node could be trivially linked according to the addresses used by the node's hardware and software. It is equally important to ensure that message transmissions from a node cannot be linked to each other due to the use of any alternative medium (e.g., cellular telephony) transceiver whose identifier remains fixed.

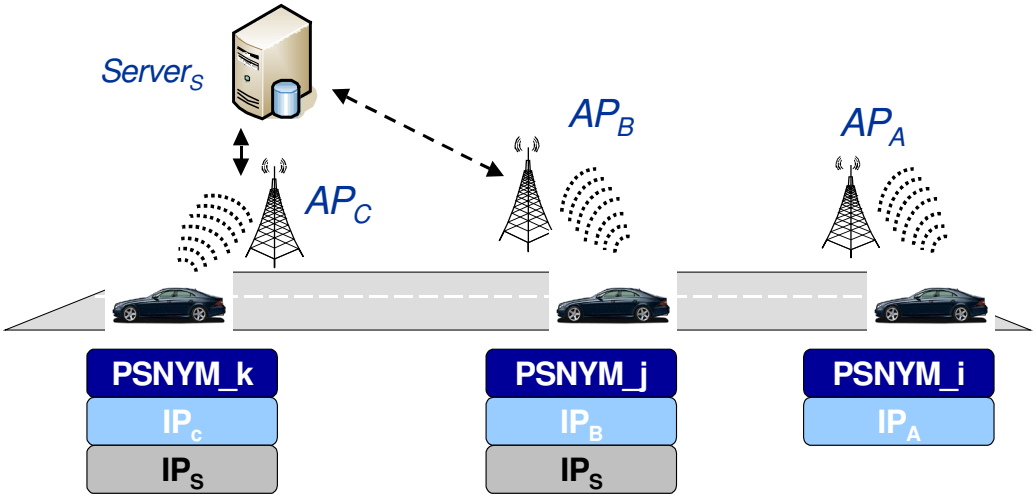

Figure 5-9: Interaction of Pseudonym Changes and Network Protocol Stack Functionality

On the other hand, the network operation may require that node identifiers remain unchanged for a specific period of time. This implies that a change of pseudonym would be ineffective and thus meaningless throughout the period a protocol identifier must remain unchanged. Two such situations are shown in Figure 5-9. First, consider a vehicle within range of an access point APA, utilizing a pseudonym PNYMi, and an IP address IPA dynamically assigned by APA; the vehicle IP address must not be changed throughout, for example, a data download session. Similarly, while in range of an APB, the vehicle utilizes PNYMj and is assigned an IPB, and establishes a session with a node S at the wire-line part of the network. If it is necessary for the vehicle to maintain the same identifier (e.g., an IP address IPS) throughout such a communication with S, it could be tracked by an eavesdropper of the wireless medium transmissions, especially if IPS is used as the vehicle reconnects to S through another APC. To remedy this, end-to-end traffic and identification (IPS) should be encrypted. Then, only the newly assigned IPC is visible over the wireless medium, as were IPA; IPB while in range of APA;APB. However, such addresses are at most locators, merely indicating that PNYMi; PNYMj and PNYMk respectively are within range of the corresponding access points.

5.3.5 Tamper Evident Security Module

In SeVeCom, we envision that the vehicles are equipped with a Tamper Evident Security Module (TESM). The purpose of the TESM is *to store sensitive information* within the vehicle, *to provide physical protection measures* to safeguard sensitive information and to provide a *secure time base*. If the module should have been tampered with, physical inspection will show evidence of the tamering, hence the term tamper evident security module. This mainly means the storage and the physical protection of sensitive cryptographic keys (e.g., private keys for signature generation). In addition, the TESM must be able *to perform cryptographic operations* (e.g., generate digital signatures) with the stored keys in order to ensure that sensitive information never needs to leave the physically secured environment provided by the TESM.

At a high level, the TESM serves as the basis of trust in the SeVeCom security architecture. In particular, without the physical protection provided by the TESM, the signature generation keys could be easily compromised, and then used to generate fake messages that appear to be authentic. Hence, in that case, the vehicles could not trust even the signed messages, and therefore, the entire security architecture would be more or less useless.

The physical protection of the TESM should ensure at least tamper evidence. However, this may not be enough, as regular inspections of the vehicles are rather infrequent (e.g., in some countries it happens in every second year), which results in a large vulnerability window. Therefore, it is desired

that the physical protection of the TESM also ensures some level of *tamper resistance*. We understand that high-end tamper resistant hardware modules are very expensive; therefore, in order for our baseline architecture to be practically feasible, we require only a minimal level of tamper resistance that can be achieved with special packaging, seals and coatings.

The main service provided by the TESM to the components that use it is the *generation of digital signatures*. In order to support this, the TESM also provides *key management* services. In particular, the TESM must be able to generate or import the private keys corresponding to the anonymous public keys of the vehicle. Furthermore, the TESM must also be able to process revocation commands originating from a trusted authority. In addition to the digital signature generation service, the HSM performs *time stamping*. This means that the TESM is equipped with a real-time clock, and upon request, it inserts the current time in a message before signing that message.[FK22]

The hardware architecture of the TESM is illustrated in Figure 5-10. The TESM has a CPU, a memory module, and some non-volatile storage. In addition, in order to ensure the freshness of the cryptographically protected messages produced by the TESM, it must also have a real-time clock, and consequently, a battery module that ensures the independent operation of that clock. Finally, the TESM also has a hardware random number generator that is used for key generation purposes.

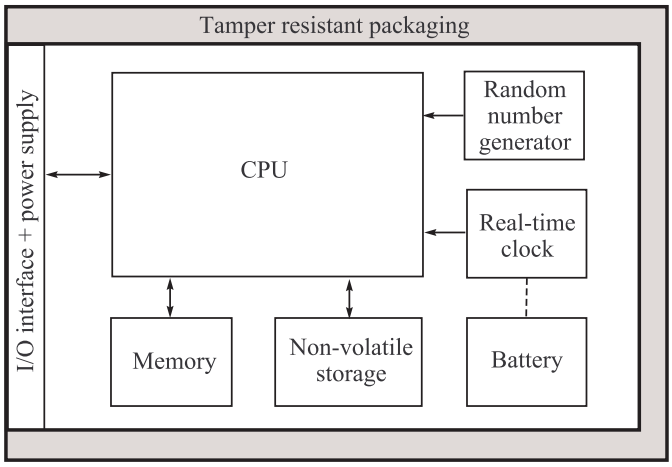

Figure 5-10: Architecture of the TESM

Note that having a trusted clock is indispensable, as otherwise the TESM could be coerced to produce cryptographically protected beacon messages *in the future* that can later be used to mislead other vehicles. Note also that one could include the GPS receiver (and some other sensors) of the vehicle within the TESM, but in our opinion, this is not indispensable as long as the TESM is equipped with its own trusted clock.[FK23] The TESM could still be fed with incorrect position information (and sensorial data), but now the attacker must do this *in real-time*, which is considerably more difficult. In particular, not having unsupervised access to the vehicle constantly, the attacker must install some rogue equipment inside the vehicle that feeds the TESM with corrupted position information. In-vehicle intrusion detection mechanisms could be used to mitigate this problem. Another reason for not including the GPS receiver in the TESM is that we would still need an external GPS antenna, which could be used to feed the GPS receiver inside the TESM with a spoofed GPS signal. Physically protecting the GPS receiver makes no sense when the GPS signal that it receives cannot be trusted anyway.

The TESM must satisfy some timing requirements that are determined by the applications that use it. The most stringent timing requirements are determined by the periodic beaconing. Periodic beaconing means that the vehicle periodically broadcasts its position, speed, and direction of movement, and in this way, it informs nearby vehicles about its presence. Many vehicle safety applications (e.g., collision avoidance, lane merge assistant, etc.) rely on this mechanism. These periodic beacon messages need to be digitally signed, which means that typically, the TESM must be able to generate a few tens of digital signatures per second.

Note that the vehicle may be required to verify an order of magnitude more digital signatures when receiving beacons from nearby vehicles. However, signature verification is a computation that uses only public information (i.e., public keys), and therefore it can be performed outside of the TESM, typically, on the OBU of the vehicle.

5.4 Abstract Architecture: Other Views

5.4.1 Deployment View

Figure 5-11 shows a deployment view of the architecture, showing how the conceptual view is mapped on a physical structure made up of three types of entities, vehicles, road side units (RSUs) and the service infrastructure:

- Vehicle entities includes all five modules of the conceptual view
- RSUs do not need two of these modules: the in-car security module (in charge to protect the vehicle against intrusion), and the privacy management module as it is assumed that RSU identities are public
- The service infrastructure includes the trust management infrastructure (PKI and access to certificate authorities) as well as secure communication capability with RSUs, and possibly directly with vehicles.

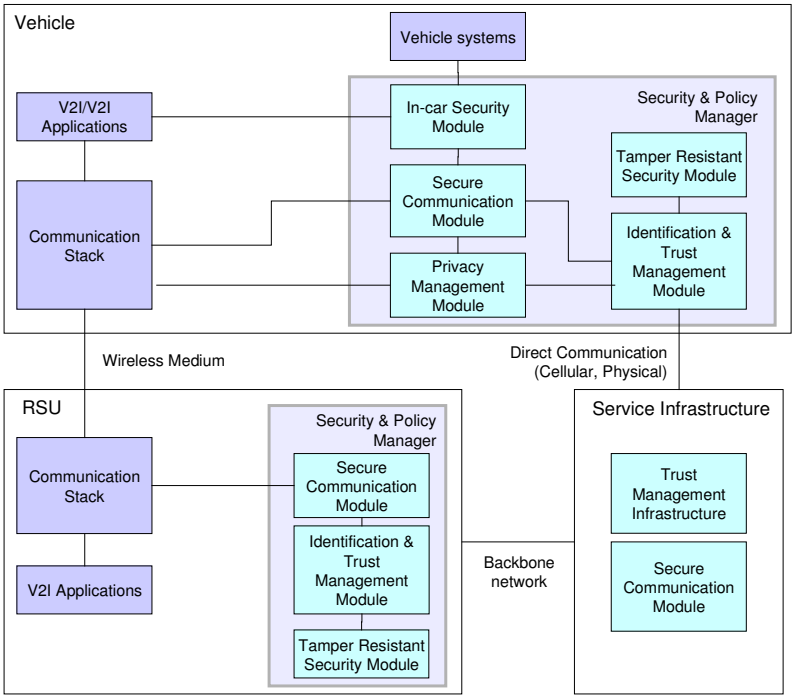

Figure 5-11: Deployment View

Vehicles, RSUs and the Service infrastructure interact as follows:

- Vehicles interact with other vehicles (not showed in figure) and with RSUs through a wireless medium
- RSUs interacts with the service infrastructure though a communication backbone (e.g. Internet based)
- Vehicles interact directly with the service infrastructure through dome direct communication capability

Figure 5-11 also shows other details of deployment:

- In vehicles and RSUs, SeVeCom conceptual modules are grouped into an overall system called the security and policy manager
- The security and policy manager internally interacts with the following entities as follows:
  - In the vehicle, the application interacts with the in-car security module, the communication stack interacts with the secure communication module and the privacy management module.
  - In the RSU, the communication stack interacts with the secure communication module.

5.4.2 Administration View

Administration capabilities are needed in order to allow the upgrade-ability of communication systems and of crypto systems. Figure 5-12 shows an administration view explaining how upgrades takes place. Vehicles and RSUs are equipped with a local administration system which interacts with a

master administration system available in the service infrastructure. Several types of updates are possible:

- Parameters (e.g. pseudonym change parameters, crypto parameters, certificates …)
- Code (e.g. a new communication pattern, a new crypto system).

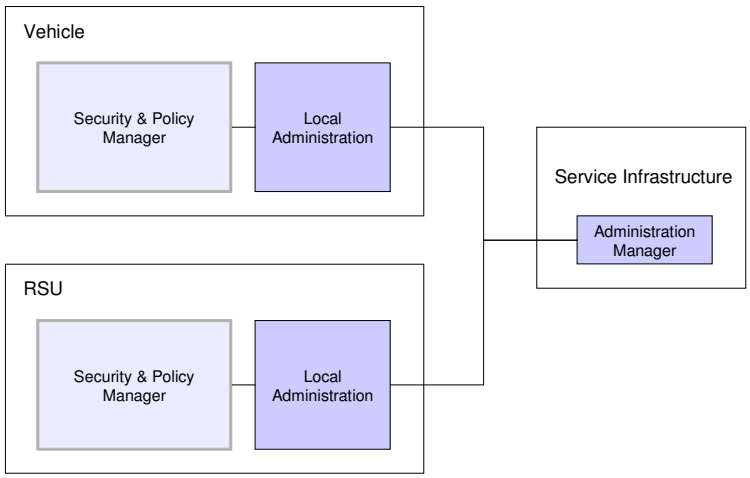

Figure 5-12: Administration View

The administration view is not elaborated further in SeVeCom.

5.4.3 Integration View

Software integration features are needed in order to allow the integration of SeVeCom security mechanisms into existing implementations. Figure 5-13 shows an integration view depicting the integration of SeVeCom security and policy manager. It consists in having a hook system which allows SeVeCom security and policy manager to be integrated with applications and/or communication stacks. The hook system is explained in detail in the next section.

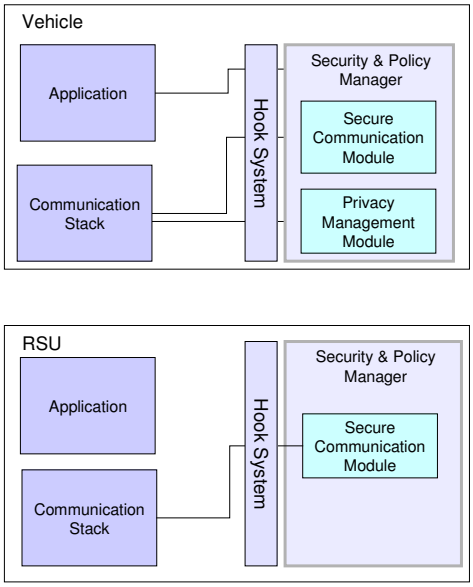

Figure 5-13: Integration View

5.5 Hook System Description

The hooking concept – which is inspired by similar architectures like Linux netfilter – is now described.

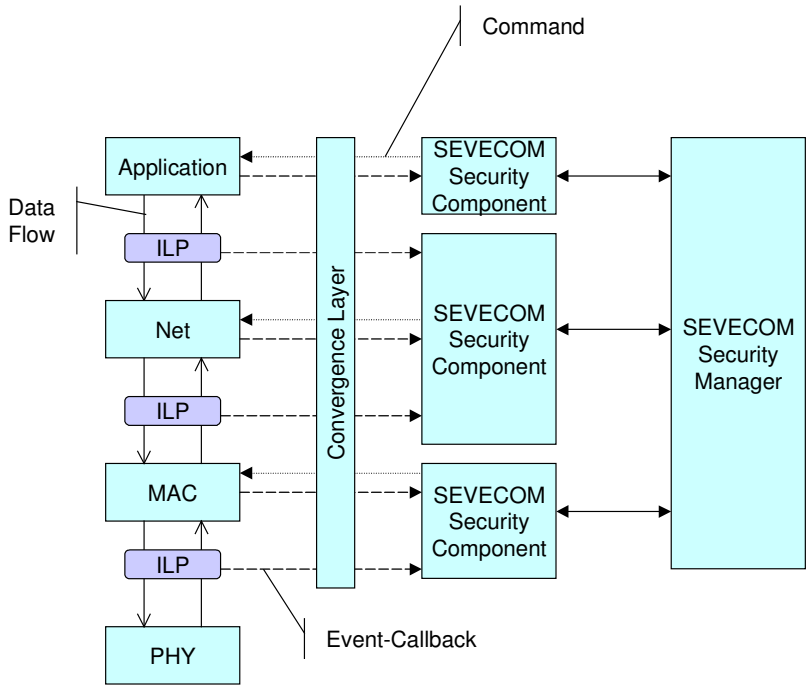

Figure 5-14: Hooking Architecture

Figure 5-14 shows the major building blocks of this hooking architecture:

- ***Functional system components*** like implementations of the individual communication layers or the application itself.
- ***SeVeCom Security Components***, which are responsible for certain tasks and interact directly with functional system components.
- The ***SeVeCom Security Manager*** that is responsible for central tasks and coordination of Security Components.
- ***Inter Layer Proxies (ILP)*** that are capable of intercepting data flows between the functional system components.
- ***Commands*** and ***Event Callbacks*** constitute the communication between security and functional system components.
- The ***Convergence Layer*** maps interactions with the security components to actual implementations of the system components

The introduced parts are described in more detail in the following sections.

SEVECOM Security Components

Every security component carries out a specific task regarding the overall security architecture. For example, the secure communication component is responsible for signing outgoing beacons and for validating the signature of incoming beacons. Moreover, it may operate many more security services to secure communication protocols or detect and react on attacks.

Interactions between functional components and security components are not one-to-one, which means that a security component may interact with multiple functional components like different communication layers. The same holds vice versa, saying that multiple security modules interact with one functional system component.

Security components may be located at various control units inside the vehicle. For example, it makes sense to implement secure communication and privacy management on the same unit that also runs the inter-vehicle communication stack. In contrast, in-vehicle security components may be located on a different control unit. The same component may therefore be instantiated in multiple control units.

All security components share a link to the general security manager, which coordinates and controls different security components. This is necessary since there are security modules that do not directly interact with functional system components, like identity & trust management module.

SEVECOM Security Manager

The security manager is responsible for instantiation, coordination and control of all security components in the system. It also provides some basic security functionalities like policy management and it is the central entity for security configuration.

Commands

Commands are blocking calls that are usually triggered by security components to functional components and should return immediately. The API of commands is individual and depends on the capabilities of the functional component. Regarding the communication stack, commands are usually not related to single packets. Instead, they may set security settings in the functional component. In some cases, functional system components may also issue commands to security components, e.g. for retrieving status information.

As an example for a command, the MAC address may have to be changed occasionally due to privacy reasons. The exact time of MAC address changes is determined by the privacy management component and may depend on a number of different influence factors outside the communication stack. The privacy management component then issues a command to the MAC component to make it change its address.

Event Callbacks

In contrast to commands, event callbacks are usually triggered by functional components and call routines of security components when specific events occur. Security components have to be registered at functional components to get called back on specific events. In case of such a callback, the functional component passes a reference to the data unit it is currently processing to the security component. The security component can then inspect the data unit and even modify it. After the callback returns, the functional system component will either continue processing the data unit or drop it, depending on the return value of the callback set by the security component.

A typical example for usage of an event callback is verifying the signature of incoming data packets. In this case, the corresponding secure communication component could register with an appropriate event that is provided by a function system component, like the network layer implementation or an Inter Layer Proxy (for details see next paragraph) between MAC and network layer. Then, packets will be passed to the secure communication component which will then verify the attached signature using the appropriate set of crypto information. If the packet passes, the callback return with a result value indicating the successful verification, otherwise it will indicate failure in which case the packet is to be dropped.

Inter Layer Proxies

In some cases it is required to intercept packets before regular processing continues at the next higher or lower layer in the communication stack. For the SEVECOM hooking concept, we therefore propose a mechanism called “Inter Layer Proxies”. Basically, these entities are inserted into the data flow between different system components and offer the same event callback functionality as system components themselves. This means, whenever a packet is received by an Inter Layer Proxy, it is passed to all registered security components in sequence. They may then modify the packet or just process it and return it back. At the end, the ILP passes the packet on to the receiving system component.

Convergence Layer and Instantiation Sketch

Though we are confident that the hooking architecture with the described mechanism to integrate security into the system is a good approach to follow, we also realize that it may be difficult to integrate it into existing communication and vehicle system solutions. Therefore, we also provide a convergence layer in the hooking architecture, which allows abstracting from specific implementations of the base system, if necessary.

As an example: Assume a communication system that completely deals with routing, medium access and the integration into the OS, and which offers a socket-based management interface that accepts incoming control packets and also constantly sends information about its internal state to a subscriber. In this case, commands and event callbacks can not be implemented directly. With the convergence layer, it is possible to translate commands into control messages and to trigger event callbacks when certain information from the management interface is received.

Introducing such a convergence layer also has the benefit that the security part does not have to be adapted for a different implementation of the base system. In this case, only the convergence layer implementation has to be modified accordingly.

5.6 Design Specification

A more detailed specification of SeVeCom baseline architecture is available as an annex.

6 Proof-of Concept Implementation

The concrete technologies to be used in SeVeCom implementations are still under discussion, as they depend on decisions made by other eSafety projects. We provide a brief status here.

Subsystem	View	Implementation objective
In car security module	Conceptual	Sevecom Separate implementation
Secure communication module	Conceptual	Sevecom reference implementation
Identification & trust management module	Conceptual	Sevecom reference implementation
Privacy management module	Conceptual	Sevecom reference implementation
Tamper evident security module	Conceptual	Sevecom software based reference implementation
Tamper evident security module	Conceptual	Sevecom software based reference implementation
Security and policy manager	Deployment	Sevecom reference implementation
Hook system	Integration	Sevecom reference implementation
Administration	Administration	Not implemented

Subsystem	View	Implementation status
Communication stack in vehicle and in RSU	Conceptual	Sevecom will use the Aktiv platform
Crypto systems	Conceptual	ECDSA is selected. Sevecom intends to use a library provided by KU Leuven

7 References

[1] R. Anderson, M. Bond, J. Clulow, and S. Skorobogatov, Cryptographic processors – a survey. Proceedings of the IEEE Publication, Feb. 2006, Vol.94, Issue 2, pp. 357- 369

[2] A. Beresford and F. Stajano. "Location privacy in pervasive computing",IEEE Pervasive Computing, 3(1):46-55, 2003.

[3] A. Beresford and F. Stajano, "Mix Zones: User privacy in location-aware services", in Proceedings of First IEEE International Workshop on Pervasive Computing and Communication Security (PerSec) 2004, a workshop in PerCom 2004.

[4] M. Bond and R. Anderson, "API level attacks on embedded systems,"IEEE Computer Magazine, Oct. 2001

[5] S. Brands and D. Chaum, "Distance-Bounding Protocols," *Theory and Application of Cryptographic Techniques*, Springer-Verlag, 1993, pp. 344–359.

[6] S.Buchegger and J.-Y. Le Boudec, Performance Analysis of the CONFIDANT Protocol, in: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc 2002), pages 226-236, Lausanne, Switzerland, 2002.

[7] J. Camenisch and E. Van Herreweghen, "Design and Implementation of the idemix Anonymous Credential System," ACM Conference on Computer and Communications Security (CCS), Washington, DC, November 2002

[8] D. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," Comm. ACM Communications of the ACM, Volume 4, February, 1981

[9] D. Chaum, "Security without identification: Transactions to make big brother obsolete," Comm. ACM 1985

[10] D. Chaum. The Dining Cryptographers Problem: Unconditional sender and recipient untraceability. Journal of Cryptology, 1(1):65--75, 1988.

[11] Cranor, Guduru, Arjula, "User Interfaces for Privacy Agents", ACM Transactions on Computer-Human Interaction, Vol. 13, No. 2, June 2006, Pages 135–178.

[12] E.Cronin, S.Jamin, T.Malkin, and P.McDaniel, "On the performance, feasibility, and use of forward-secure signatures", In *Proc. of 10th ACM Conference on Computer and Communications Security*, Oct 2003.
[13] N.Daswani, "Cryptographic Execution Time for WTLS Handshakes on Palm OSDevices", Certicom Public Key Solutions, San Jose, CA, September 2000.
[14] F. Dötzer, L. Fischer and P. Magiera, VARS: A Vehicle Ad-Hoc Network Reputation System, in: Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2005.
[15] H.Füssler, M.Mauve, H.Hartenstein, M.Käsemann, D.Vollmer, "A Comparison of Routing Strategies for Vehicular Ad Hoc Networks", in: Technical report TR-3-2002, Department of Computer Science, University of Mannheim, July 2002
[16] H. Füssler, J. Widmer, M. Käsemann, M. Mauve, H. Hartenstein, "Contention-Based Forwarding for Mobile Ad-Hoc Networks", in: Elsevier's Ad Hoc Networks, 1 (4), pages 351-369, 2003
[17] P. Golle, D. Greene and J. Staddon, Detecting and Correcting Malicious Data in VANETs, in: Proceedings of the First ACM Workshop on Vehicular Ad Hoc Networks (VANET 2004), pages 29-37, ACM Publishing, 2004.
[18] V.Gupta, D.Stebila, S.C.Shantz, "Integrating Elliptic Curve Cryptography into the Web's Security Infrastructure", SUN research paper, 2004, see http://research.sun.com/projects/crypto/p915-gupta-final.pdf
[19] C. Harsch, A. Festag, and P. Papadimitratos, "Secure Position-Based Routing for VANETs", in Proceedings of the IEEE 66th Vehicular Technology Conference VTC2007-Fall, Baltimore, Oct. 2007 (to appear).
[20] J.Hoffstein, J.Pipher, J.Silverman, "NSS: The NTRU Signature Scheme", available from the NTRU site at http://www.ntru.com/cryptolab/pdf/nss.pdf
[21] J.-P. Hubaux, S. Capkun and J. Luo, The Security and Privacy of Smart Vehicles, *IEEE Security & Privacy Magazine*, Vol. 2, No. 3, pp 49--55, May-June 2004.
[22] F. Kargl, A. Klenk, S. Schlott and M. Weber: *Advanced Detection of Selfish or Malicious Nodes in Ad hoc Networks*, Proceedings of 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004), Springer Lecture Notes in Computer Science, Heidelberg, Sept. 2004
[23] F. Kargl, S.Schlott and M. Weber, "Identification in Ad hoc Networks", Hawaiian International Conference on System Sciences (HICSS 39), January 2006
[24] B. Karp, H.T. Kung, "Greedy Perimeter Stateless Routing for Wireless Networks", in: Proceedings of the Sixth ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000), pages 243-254, August 2000
[25] J.O.Lauf, "Algorithms for Multimedia Multicast Stream Authentication", MSc Thesis, Technical University of Hamburg-Harburg, January 2004. available at http://www.sva.tu-harburg.de/lauf/diplom/index.html
[26] T. Leinmüller, E. Schoch, F. Kargl and C. Maihöfer, Influence of Falsified Position Data on Geographic Ad-Hoc Routing, in: ESAS 2005: Proceedings of the second European Workshop on Security and Privacy in Ad hoc and Sensor Networks, Visegrad, Hungary, 2005
[27] T. Leinmüller, E. Schoch, and F. Kargl, " Position Verification Approaches for Vehicular Ad Hoc Networks", IEEE Wireless Communication Magazine, Oct. 2006, Vol.13, Num.5
[28] T.Leinmüller, E.Schoch, " Greedy Routing in Highway Scenarios: The Impact of Position Faking Nodes", in: Proc. of Workshop On Intelligent Transportation (WIT), March 2006
[29] T. Leinmüller, E. Schoch, F. Kargl and C. Maihöfer, Improved Security in Geographic Ad Hoc Routing through Autonomous Position Verification, in: The Third ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2006) - in conjunction with MobiCom 2006, pages 57-66, 2006.
[30] C.Maihöfer, R.Eberhardt, E.Schoch, "CGGC: Cached Greedy Geocast", in: Proc. 2nd Intl. Conference Wired/Wireless Internet Communications (WWIC 2004), pages 13-25, February 2004
[31] S.Marti, T.J. Giuli, K.Lai and M.Baker, Mitigating routing misbehavior in mobile ad hoc networks, in: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom), pages 255-265, Boston, Massachusetts, United States, 2000
[32] S. Micali, "Efficient certificate revocation," MIT Laboratory for Computer Science, Tech. Rep. TM-542b, Mar. 1996.
[33] P.Michiardi and R.Molva, CORE: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks, in: Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security, pages 107-121, Kluwer, B.V., 2002.
[34] K. O'Neill and J. Y. Halpern, "Anonymity and information hiding in multi-agent systems," Proceedings of the 16th IEEE Computer Security Foundations Workshop, 2003
[35] P. Papadimitratos and Z.J. Haas. "Secure Link State Routing for Mobile Ad Hoc Networks." In Proceedings of the IEEE Workshop on Security and Assurance in Ad hoc Networks, in

conjunction with the 2003 International Symposium on Applications and the Internet, Orlando, Florida, January 28, 2003
[36] P. Papadimitratos, A. Kung, J.-P. Hubaux, and F. Kargl, "Privacy and Identity Management for Vehicular Communication Systems: A Position Paper." Workshop on Standards for Privacy in User-Centric Identity Management, Zurich, Switzerland, July 2006
[37] P. Papadimitratos, V. Gligor, and J.-P. Hubaux. "Securing Vehicular Communications - Assumptions, Requirements, and Principles." In proceedings of the Workshop on Embedded Security in Cars (ESCAR) 2006, Berlin, Germany, November 2006
[38] P. Papadimitratos, L. Buttyan, J-P. Hubaux, F. Kargl, A. Kung, and M. Raya, " Architecture for Secure and Private Vehicular Communications. " In Proceedings of the 7th International Conference on ITS Telecommunications, Sophia Antipolis, France, June 2007
[39] F. Picconi, N. Ravi, M. Gruteser, L. Iftode. Probabilistic Validation of Aggregated Data in Vehicular Ad-hoc Networks. In Proceedings of VANET'06, Los Angeles, California, USA, September 2006.
[40] M. Raya and J.-P. Hubaux, The Security of Vehicular Ad Hoc Networks, In SASN'05, November 2005
[41] M. Raya, P. Papadimitratos, and J-P. Hubaux, "Securing Vehicular Networks," IEEE Wireless Communications, Volume 13, Issue 5, October 2006
[42] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux, " Eviction of Misbehaving and Faulty Nodes in Vehicular Networks," IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Vehicular Networks, 4th Quarter, 2007 (to appear)
[43] E. Schoch, F. Kargl, T. Leinmuller, S. Schlott, and P. Papadimitratos. "Impact of Pseudonym Changes on Geographic Routing in VANETs." In proceedings of the European Workshop on Security and Privacy in Ad hoc and Sensor Networks (ESAS), Hamburg, Germany, October 2006
[44] P. F. Syverson and S. G. Stubblebine, "Group principals and the formalization of anonymity," Proceedings of the World Congress on Formal Methods, 1999
[45] S.Vanstone, "Deployment of Elliptic Curve Cryptography", In *Proc of the 9th Workshop on Elliptic Curve Cryptography*, September 19, 2005
[46] Article 29 Working Party, "Working document on data protection and privacy implications in eCall initiative", 26. September 2006, http://ec.europa.eu/justice_home/fsj/privacy/docs/wpdocs/2006/wp125_en.pdf.
[47] PAMPAS: Pioneering Advanced Mobile Privacy and Security, URL: http://www.pampas.eu.org/index.html
[48] MODINIS-IDM: Study on Identity Management on eGovernment, URL: https://www.cosic.esat.kuleuven.be/modinis-idm/twiki/bin/view.cgi/Main/WebHome
[49] PORTIA: Privacy, Obligations, and Rights in Technologies of Information Assessment, URL: http://crypto.stanford.edu/portia/
[50] FIDIS: Future of Identity in the Information Society, URL: http://www.fidis.net/
[51] PRIME: Privacy and Identity Management for Europe, URL: http://www.prime-project.eu.org/
[52] SecurIST: ICT Security and Dependability Taskforce, URL: http://www.ist-securist.org/
[53] eSafety, URL: http://europa.eu.int/information_society/activities/esafety/forum/index_en.htm}
[54] "DSRC: Designated Short Range Communications," URL: http://grouper.ieee.org/groups/scc32/dsrc/index.html
[55] "IEEE P1609.2/D2 - Draft Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages," November, 2005
[56] Checkpoint Intrusion Detection & Prevention http://www.checkpoint.com/products/intrusion.html
[57] IBM Internet Security Systems http://www.iss.net/products/product_sections/Intrusion_Detection_.html
[58] CISCO Self Defending Network http://www.cisco.com/en/US/products/hw/vpndevc/index.html
[59] Snort http://www.snort.org/
[60] ISO/IEC JTC 1/SC27 N3177: Final Text for ISO/IEC/ TR 15947, Information technology – Security techniques – IT intrusion detection framework
[61] MITRE: Common vulnerability enumeration http://cve.mitre.org
[62] DARPA: Common Intrusion Detection Framework (CIDF) http://gost.isi.edu/cidf/
[63] IETF Intrusion Detection Working Group http://www.ietf.org/ids.by.wg/idwg.html
[64] Vehicle Safety Communications (VSC) Project, Final Report, DOT HS 810591, Appendix H: WAVE/DSRC Security, April 2006.
[65] NHTSA Event Data Recorder Research Web site, http://www-nrd.nhtsa.dot.gov/edr-site/