

# Security Engineering for VANETs

Frank Kargl, Zhendong Ma, and Elmar Schoch  
 Ulm University, Institute of Media Informatics,  
 {frank.kargl|zhendong.ma|elmar.schoch}@uni-ulm.de

**Abstract**—The “Secure Vehicle Communication” project (SEVECOM) has the aim to develop a future proof security solution for vehicular communication (VC). As there are no clear scenarios and standardized protocols for VC yet, traditional approaches for security engineering fail, as they usually consider a specific scenario and system. At the same time, VC encompasses dozens of potential applications with very diverse properties. At this point, it is hardly possible to discuss security and privacy issues of all these scenarios in detail. On the other hand, it might be problematic to leave out an important application with distinguished properties when creating the basis of security mechanisms. Therefore, we suggest a new approach that allows to analyze a large set of applications, select typical representatives that will cover the requirements of a whole cluster of applications and develop a security solution for this subset. Additionally, we also present the results of this process that give new insight into security requirements of VANETs.

## I. INTRODUCTION

Spontaneous communication between vehicles or between vehicles and road-side infrastructure is an important research area that a significant number of projects have addressed during the recent years. Examples include Fleetnet [1], NoW [2], VSC [3], CVIS [4], and Safespot [5].

These projects suggest a long number of potential applications addressing road safety or trying to enhance driver and passenger comfort. Examples include collision warning at intersections or warning of violated traffic lights, detection and mutual warning of dangerous road conditions between cars, direct car-to-car messaging, and many more.

It is obvious that these applications may become the target of attackers that will try to interfere with the proper operation for fun, profit, or damage reasons. For instance, some pranksters might send bogus warning messages to other cars, pretending that there are dangerous road conditions ahead. This might lead to cars slowing down or breaking, resulting in traffic jams or even accidents.

The EU-funded project SEVECOM (“SEcure VEhicle COMmunication”) aims at developing a future proof security solution for vehicular communication (VC) to

prevent such attacks. Yet, the variety of envisioned applications in conjunction with almost no standardized protocols makes this a tedious task.

The first difficulty is that VC encompasses dozens or even hundreds of potential application scenarios with very diverse properties. It is not possible to discuss all these scenarios in detail. On the other hand, if one leaves out an important application with distinguished properties or a combination of properties that is not covered by others, the security analysis may suffer.

As a second difficulty, there are no precise scenarios and standardized protocols for VC yet. For this reason, traditional approaches for security requirements engineering usually fail, as they require a specification of the system.

Commonly used methods for security assessment include the Common Criteria [6] and Octave [7], but both focus on security evaluation of established systems within commercial organizations. This clearly does not fit the problems that we are facing, on which we want to assess the security problems in an application area, namely Vehicular Communication (VC).

A common way to access this is by using use-case templates. Starting a security requirements engineering process for an area, where there are no definitive scenarios or protocol specifications, just based on our intuition is very problematic. Based on a very sketchy knowledge of protocols and applications, the danger is that certain applications containing important security aspects are overseen and left out. Clearly, a deeper understanding of the applications and their security requirements is required, followed by a structured decision on what applications to consider and what to leave out.

Hence, we suggest a new approach that allows to analyze a large set of only informally specified applications, to select typical representatives that will cover the requirements of a whole cluster of applications, and develop a security solution for this subset of applications which is expected to cover the requirements of all applications considered.

We used this approach to conduct a detailed security analysis for VC applications. The results of this study will be presented at the end of this paper. Before

describing the process, we present a short overview on security in VANETs and on related approaches to security engineering.

## II. RELATED WORK

Security and privacy in VANETs are gaining increasing attention and interest from research communities. Hubaux et al. in [8] address the security and privacy challenges in VANETs, which have been previously overlooked. In [9], Hubaux and Raya provide a detailed threat analysis and propose a security architecture based on public key cryptography. Another attempt to find security requirements in VANETs is taken by the NoW project [10], in which the authors use attack trees [11] to access threats on VC system and briefly describe requirements to thwart such attacks.

In this paper, we apply cluster analysis as part of our security engineering process to find the security requirements of VANET applications. Cluster analysis is commonly used in statistical data analysis, including data mining, or image analysis. It has been used in menu interface design for in-vehicle multimedia applications, such that menu tasks can be easily accessed by the driver without roaming the interface [12]. Security engineering processes like the ones described in [13], [14], [15] all state that the first step of designing and implementing a secure system is requirements engineering. Requirements engineering is an iterative process that normally begins with threat analysis. The core activities in defining requirements include finding, modeling, and analyzing requirements, communicating requirement, agreeing on these requirements, and evolving them during the overall process [16]. In our approach, we adopt use cases as the main method in our security engineering process, where security requirements are embedded in the application use cases. In contrast, typical use cases do not include security concerns as an integral part of requirements engineering and security policies are added as an afterthought [17].

There have been studies in requirements engineering to group similar requirements. For example, automated similarity analysis among textual requirements to increase the efficiency of the requirements engineering process is described in [18], [19]. The authors of [20] propose a requirements clustering technique based on the verbs and keywords of each requirement. Nevertheless, current similarity analysis approaches in requirement engineering are all based on statistical text processing, and the motivation is to reduce redundancies and inconsistencies. To the best of our knowledge, no attempts have been made to apply cluster analysis in a security engineering process.

## III. SECURITY ENGINEERING PROCESS

Starting with a short overview, we describe the security engineering process in detail in this section. Fig. 1 shows the main steps in the process.

Because clear standards and even defined protocols are missing, which is usually the starting point for security mechanisms, we argue that it is most suitable to begin with what all activities, projects and initiatives have in common: the envisioned applications.

So the basic concept is to first collect an application list that comprises as many use cases as possible. With this list, we do a preliminary analysis of application characteristics and security requirements for all applications. After having done this, the list most probably contains all relevant constellations regarding security that may arise.

The disadvantage of this approach is the extreme complexity due to dozens of application characterizations which makes it impossible to design security mechanisms for each individually. Moreover, separate security systems for every application are neither applicable nor useful. So, the following step is to group similar applications. In many cases where applications have similar properties and security requirements, it is sufficient to consider only one representative of the group. Our approach to find these groups of applications with similar classification is to use cluster analysis.

Next, when we have gathered similar applications in their corresponding clusters, we can select a small subset of representative applications from each cluster and analyze them in more detail.

Detailed application use cases will now describe the regular operation of the applications and identify all needed components and protocols. The described systems will not have any security mechanisms in place, but only the operations as needed from an application point-of-view.

So called attack use cases then describe potential attacks against the applications and their security requirements. From that, we can derive a set of required security mechanisms that will prevent these attacks. The design of these mechanisms is the next step before the last one. As the introduction of these security mechanisms might introduce changes to the application itself or even open up new opportunities for attacks, there is a loop back to previous steps here.

Finally, the last step will analyze whether the found solutions will also apply to the other applications within each cluster.

### A. Step 1: Create Application List

The goal of this initial step is to find a detailed list of potential applications for the area to be analyzed. For

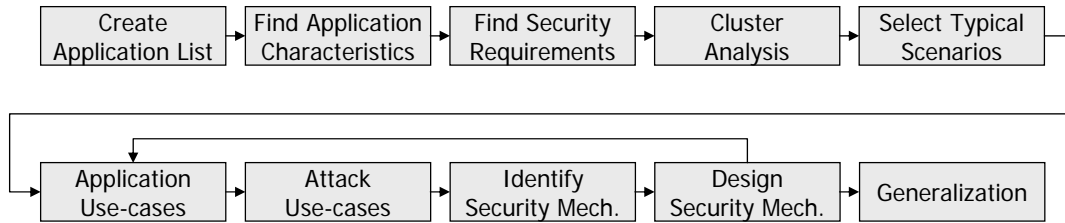


Fig. 1. Steps of Security-Requirements Engineering using Cluster Analysis (SECA) process

VC, this list will contain all applications that might be used in VC scenarios together with a short description of the application. Typically, such a list will be the result of researching the related work, intensive discussions and brain-storming sessions.

Within the project, we gathered an application list based on own discussions and material provided by VSC [21] and others [22]. One major insight from that process is the extreme variety of domains, where VC could enable applications. Beyond the typical VANET scenarios, where vehicles warn each other of hazardous road situations, communicate to avoid collisions, help the driver ramping the highway or improve navigation by sending out traffic information, there are numerous different application areas. For instance, integrating traffic infrastructure like signs and traffic lights into the VC system could improve driving and support for authorities. Commercial infrastructure nodes might lead drivers to free parking lots and let them download the latest map updates for the navigation system. Similar to that, vehicle maintenance could be improved by wireless diagnostics or just-in-time repair notification. In summary, VC application involve all situations in a vehicle's life - on the road, at home, in the garage, by warning, helping and facilitating.

### B. Step 2: Find Application Characteristics

As next step, we characterize applications to further understand details and for later classification. The first task for that is to find properties that describe characteristic aspects of the applications and can be used to distinguish different kinds of applications.

For VC, we defined properties that answer most relevant questions on the application, including general estimations on importance, technical requirements, and application situation.

After the properties are defined, each application needs to be classified in every property. Because there is no definite answer in most cases, estimates need to be given. Although it might be hard to actually answer

these questions without having a application- and protocol description available, our own work has shown that experts are usually able to come up with pretty reasonable assumptions.

While we will give meaningful classes for each property here, the final values of the properties need to be given in a numerical form, describing e.g. the importance of the property for an application, where '0' stands for irrelevant, '1' for important and '2' for very important. This is necessary so the cluster analysis algorithm can determine numerical distances between properties and applications.

1) *Influence on safety*: Among the various applications, we find different levels of influence on road safety. Many applications are **safety-critical**, like intersection collision avoidance, that is used in hazardous situations. Other applications are only intended to improve road safety to some extent which we then denominate as **safety-related**. For instance, missing a workzone warning is not likely to cause as much harm as missing a collision warning. A third category of applications is **not safety related** at all, e.g. a parking spot locator service.

Regarding security, this characteristic directly indicates how much attention an application requires.

2) *Driver involvement*: Applications feature totally different extent of driver's involvement. Whereas in some cases the driver manually triggers messages, in other cases the vehicle creates messages autonomously without even notifying the driver. Also at the receiver's side, messages may be treated by the **vehicle only**, or they may demand the driver's **awareness, attention** or even **reaction**. This is also strongly related to security requirements. If a driver is intended to react immediately e.g. on a warning message, the message content must be absolutely trustable.

3) *Interworking of communication*: When it comes to communication, several important questions need answers. The first is to clarify which parties will be involved in the communication. For traditional VANETs, communication is only **car-to-car**, which means that

cars trigger messages and deliver them to other cars. Yet in the whole scenario, also a lot of infrastructure nodes are involved whose capabilities and needs are obviously different than vehicles. For example, in an **infrastructure-to-car (I2C)** application, a traffic light might send state changes to vehicles. From the opposite perspective, vehicles might send SOS messages to infrastructure nodes with backend network to call for help (**car-to-infrastructure (C2I)**). Security is influenced, for instance, because infrastructural components of VANETs usually don't need privacy.

4) *Direction of communication:* Regarding security, it is important to distinguish between **one-way** and **two-way** communication. For example, in case of some warning applications, a vehicle might only get one packet and then has to decide whether to trust the contained information. Moreover, for typical two-way applications like electronic payment or wireless diagnostics, encryption of data is likely needed.

5) *Forwarding of messages:* While there are applications that only need **single-hop** communication, many typical ones distribute information **multi-hop**, using other nodes as forwarders. Both types of communication raise specific security questions, but secure routing is harder to obtain because routing by definition involves multiple – potentially fraudulent – nodes.

6) *Addressing:* Before messages can be sent out, one of the most important questions is who will receive them. In our application list, we have some applications that use **unicast** addressing whereas others **broadcast** information to a certain neighborhood, but also many applications apply **geocast**. Many information in VC is position dependent, also the destination of messages is often specified geographically. Therefore, securing the position data also plays a vital role.

7) *Timing constraints:* Among the applications, timing constraints vary extremely. Whereas in some cases, timely delivery has highest priority (**highly time-critical**), time is **no critical issue** in other cases. For highly time-critical applications, we assume a maximum delay of  $\sim 500ms$ , for **time-critical** applications  $\sim 1s$  and for applications for which **time is relevant**  $\sim 5s$ . Those applications with no time constraints may have delays of more than  $10s$ . Particularly, applications with highly time-critical messages are sensitive to network disturbance.

### C. Step 3: Find Security Requirements

Like in the previous step, we now have to provide a set of security requirements that will be relevant for the applications. It is important to describe only

requirements based on the application needs and not to include assumptions about potential security mechanisms here. The security requirements for VC are derived from the principle security goals such as authentication, integrity, and confidentiality etc. According to the applications and their communication characteristics, we define one or more security requirements based on the principle security goals. Since the security requirements are used as input to the cluster analysis in step 4, the values of these properties also need to be described in a numerical form, where e.g. '0' stands for irrelevant, '1' for important and '2' for very important.

1) *Authentication:* Trust is crucial in safety-related applications, in which vehicles react according to legitimate messages they received. Authentication ensures that the sender of a message is correctly identified. With **ID authentication**, the receiver is able to verify a unique ID of the sender. The ID could be the license plate or chassis number of the vehicle. Yet, in many cases, the actual identity of nodes does not play an important role – receivers are satisfied if they are able to verify that the sender has a certain property. Hence, **property authentication** is a security requirement that allows verifying properties of the sender, e.g. that the sender is a car, a traffic sign etc. For applications using location informations, **location authentication** allows to verify that the sender is actually at the claimed position, or that the message location claim is valid.

2) *Integrity:* Applications requiring **integrity** specify that the transported information must not be altered between sender and receiver.

3) *Confidentiality:* Some applications require that only the sender and the intended receiver can access the content of a message, e.g. instant messaging between vehicles. Confidentiality specifies that transported information cannot be eavesdropped on its way between sender and receiver.

4) *Privacy:* Privacy is an important factor for the public acceptance and successful deployment of VANETs. It means that the driver is able to keep and control the information related to the vehicle (e.g. identity of the driver, the driving behavior, the past and present location of the vehicle etc.) from other parties. Without privacy protection, VC provide a convenient way for an observer to track and identify the vehicle and its passengers, hence makes the Big Brother surveillance scenario more a reality than a fiction. But safety-related applications in VC also require trust between the communication partners, so total anonymous for privacy reason is not feasible. There are different security requirements for privacy, in this way the information of the vehicle and the driver can be protected as much as possible. For

example, in "vehicle-based road condition warning", a car does not need to reveal its identity, but needs to provide its location information so that other cars can estimate e.g. the relevance of received warning messages. **ID privacy** specifies how much the identity of the sender should be kept secret. Depending on the applications, **location privacy** has different levels, which range from distributing location information freely throughout the network to totally keeping it private. Although privacy requirements apply for normal communications, public authorities wishing to have access to the identity or location information of cars may have **jurisdictional access**.

5) *Availability*: Some applications, particularly safety applications, require high availability of the communication system. For example, a post-crash/breakdown warning requires that the radio channel is available such that approaching cars can receive the warning message in time. If the medium is jammed e.g. by an attacker and therefore such messages don't arrive at the receivers in a very short time, the application gets useless.

6) *Access control*: **Access control** is necessary for applications that need fine-grained definition of the rights that a user or infrastructure component has. For instance, an authorized garage may be allowed to fully access wireless diagnostics, whereas other parties may only be granted limited access. Another form of access control can be the exclusion of misbehaving nodes (e.g. by an intrusion detection system using a trust management scheme) from the VANET by certificate revocation or other means.

7) *Non-repudiation*: Certain application need to track and reconstruct what was going on in the past. In our project, the non-repudiation requirement is also called **auditability**, by which senders or receivers can prove that messages have been received or sent respectively. For some applications, messages may only be stored for a very limited time (e.g. the last 10 seconds in a ring buffer) and made permanent only in case of an incident (e.g. crash).

An example of security requirements is shown in the "Vehicle-based road condition warning" use case (see Fig. 2 in the appendix). First, the receiver should be able to authenticate the property that the sender actually is a car and that the location of the sender is correct. Otherwise, attackers may use an arbitrary wireless transmitter by the roadside or forge the location information to make upcoming cars believe the hazardous road condition ahead. Furthermore, property authentication can make sure that the sender is able to detect the road condition, e.g. a car equipped with ESP/VSC sensors in contrast to a car without appropriate sensors. The application

also requires integrity, so the message cannot be altered during transmission (e.g. a message saying wet surface ahead altered by attacker to be icy road). Regarding privacy, as the sender is a private car, the identity of it should be kept private, too. Since it is not a safety critical application, the security requirements such as jurisdictional access and availability are set to medium value. Access control and non-repudiation only play a minor role, and confidentiality is not applicable because the warning is a public information and the set of receivers is not known.

#### D. Step 4: Cluster Analysis

After describing all the different applications, their properties and security requirements, the next step is to group applications in clusters with similar properties and security requirements. It's not a trivial task to cluster these applications, because each application has an attribute vector with 23 numeric elements, which are assigned values ranging from 0 to 2 (the methodology of assigning values has been described previously in step 2 and 3).

We have used the k-means algorithm to do this clustering. K-means algorithm clusters objects based on attributes into k groups. In our case, we want to cluster the applications into k groups according to their properties and security requirements. For this iterative mechanism, first the number of clusters needs to be defined. It then attempts to identify relatively homogeneous groups of objects based on vector space of the attributes. For every cluster, property values for the cluster center and the distance of applications belonging to the cluster are available when the algorithm has finished. The k-means analysis can be done using a statistics software like SPSS and delivers results as shown in the later sections of this document. Using numerical properties and a mathematical algorithm for this step allows us to quantify the similarity between different applications, basing the grouping of different applications on an objective criterion.

#### E. Step 5: Select Typical Scenarios

The selection of cluster representatives is a manual process. There are different strategies how to do so. One strategy is to choose these applications which have the closest distance to the cluster centers, as they represent their cluster best. Another strategy would be to select the most "interesting" applications, whatever "interesting" means in the context of the research activities. Again, every cluster must be covered by at least one application.

### F. Step 6: Application Use Cases

For all selected applications, a detailed application use case is described that defines the applications in terms of involved components and operation steps. For all examples, we use an identical form so that applications and their description can be compared easily.

It is important to note that the applications are described "as is", i.e. without any security measures in place. This way, we are able to fully concentrate on the desired behavior of the application. Possible security weaknesses are to be discovered in the next step.

### G. Step 7: Attack Use Cases

In a next step, detailed descriptions of various attacks need to be found. We use a form similar to the application use case form for describing these attack use cases. They allow to find security weaknesses in the application scenarios.

### H. Step 8: Identify Security Mechanisms

Based on the attack use cases, we identify necessary security mechanisms that may be used. These security mechanisms specify only required functionalities, e.g. a protocol that can authenticate both partners of a communication or a mechanism that prevents replay attack. A detailed implementation is the design phase in step 9.

### I. Step 9: Design Security Mechanisms

This is the actual design phase of our process. Based on the identified threats and required mechanisms, we now design and propose e.g. cryptographic protocols or mechanisms which provide the necessary security functionality.

Of course there typically are many options to choose from and one will need to consider the advantages and disadvantages of these options before making a decision. Introducing security mechanisms may lead to additional attack vectors, e.g. attacks on a PKI system if one is needed to manage identities. Therefore, there is a loop in the process going back to step 7 where additional attacks targeting the security system can be described.

It may also be part of this step to analyze the effectiveness and efficiency of the proposed methods. This can e.g. be done using simulations or formal methods.

### J. Step 10: Generalization

Up to now, we have only considered the selected cluster representatives. In a final step, we have to analyze whether the security mechanisms will also work with the other applications that are to be realized.

## IV. SECURITY ENGINEERING FOR VEHICLE COMMUNICATION

Following the process described above, we have done a detailed security analysis of VANETs. Starting with a list containing 56 different VANET applications found both in external sources and by the project partners, we have first specified all of these applications in terms of the categories described in step 2. This provided a very detailed understanding of how these applications can be implemented and might operate. Of course this included decisions e.g. whether applications should be realized with infrastructure support or by direct car-to-car communication. For many of these questions, there is no consensus among researchers yet, e.g. generally infrastructure-based solutions are preferred in the US whereas in Europe direct car-to-car communication is in favor. As our project partners also include members working in this field for years and contributing to many of the earlier projects, we are confident that we have made substantial good decisions in these cases.

The next step is to determine potential security requirements as described in step 3 and determine the relevancy for each of the 56 applications. One insight gained from that is that applications have very diverse security requirements that sometimes are even contradicting. So whereas e.g. one application has strong authentication needs, another might instead have strong privacy requirements. So there will never be a "one-size fits all" security solution for VANETs, instead there should be multiple components in the security architecture that can be tailored according to the specific application needs. One of the possible solutions can be: applications first declare their security requirements, security modules on each level then configure according to the specifications, the similar requirements merge, and the contradicting requirements resolved via priorities.

Based on the information from the previous step, we started the cluster analysis (step 4). It was initially clear that not more than 10 clusters should be considered, as otherwise the resulting amount of work for specifying the application use-cases etc. would get too high.

We ran the cluster analysis with cluster sizes from 5 to 10. According to the average and maximum distances from applications to the respective cluster centers, and the distribution of applications per cluster, we decided that 8 clusters is a reasonable number.

The next step was to select one or two representative applications per clusters. Instead of selecting the applications closest to the cluster centers, we preferred the ones that are also considered interesting e.g. by other projects like CVIS or Safespot and by consortia like the C2C-

CC, as they will probably also be the first applications to be specified and implemented. Our list of reference applications consists of:

- 1) SOS services
- 2) Stolen vehicles tracking
- 3) Map download/update
- 4) Intersection collision warning
- 5) Vehicle-based road condition warning
- 6) Electronic license plate
- 7) Road surface conditions to TOC
- 8) Software update/flashing
- 9) Emergency vehicle signal preemption
- 10) Work zone warning

Step 6 requires to describe each of these applications in a detailed use case form. Fig. 2 shows an example of a use case description for "Vehicle-based road condition warning". These application use cases are detailed enough to provide a basis for both protocol design and implementation and for the identification of security vulnerabilities.

Step 7 consists of designing attacks against these applications. Fig. 3 shows an example of an attack use case describing an attack against the vehicle-based road condition warning. By forging messages, attackers may influence driving behavior and in the worst case provoke accidents.

We have currently reached this state where attack use cases have been created and discussed. The following steps are yet to be done and are only outlined here.

Based on the attack shown in Fig. 3, one can e.g. identify in step 8 that a mechanism for prevention of message forgery is needed. Only messages that transport "real" data, should be permitted.

In step 9, message forgery might e.g. be prevented by a system with unique IDs per vehicle where every message must be signed by the originator of the warning. A cooperative intrusion detection system might then detect cars emitting a lot of bogus messages and nodes may decide to ignore such messages further on.

## V. RESULTS AND OUTLOOK

Even until now, our work provided us with a couple of interesting insights and lessons learned. We have compiled one of the most complete and precise categorization of VC applications and their requirements. As the requirements are very diverse and esp. in security terms sometimes contradicting, any security solution will need to be both very flexible and dynamically configurable to adapt to the applications needs. If multiple applications will run in parallel within a node – which will probably be the case – this leads to additional problems, as

e.g. authentication and privacy requirements need to be prioritized.

In general we found that it is very hard to analyze security requirements and engineer security solutions for an application domain that is not fully understood and specified. We have therefore developed a process that helps in analyzing properties and requirements in an open and new application domain like Vehicle Communication and nevertheless allows a qualified identification of security requirements and needed security mechanisms. Yet, the approach can not be seen as a way to reveal definitely all necessary security requirements, especially because the application characteristics as the input to the clustering are not stable.

Follow the model, our current work consists of identification of mechanisms that will prevent attacks and that are needed to ensure the security of VC applications. This includes mechanisms like property- and geo-authentication, privacy protection mechanisms that also allow governmental access to identities, and secure storage of key material in cars. Once these mechanisms are all identified, we will continue to work on their detailed design and implementation. At the end, we will be able to provide a security framework to VC designers that will be mandatory to deploy these systems in the real world.

## ACKNOWLEDGEMENTS

Parts of this work have been carried out in contribution to the SEVECOM project [23] that is supported by the European Commission e-Safety initiative under contract no. IST-027795. We also would like to thank all project partners for their valuable contributions and comments.

## REFERENCES

- [1] W. Franz, C. Wagner, C. Maihöfer, and H. Hartenstein, "Fleetnet: Platform for inter-vehicle communications," in *Proc. 1st Intl. Workshop on Intelligent Transportation*, Hamburg, Germany, Mar. 2004.
- [2] "NoW - Network on Wheels Project," <http://www.network-on-wheels.de><http://www.network-on-wheels.de>, 2005. [Online]. Available: <http://www.network-on-wheels.de>
- [3] "Us vehicle safety communication consortium," <http://www-nrd.nhtsa.dot.gov/pdf/nrd-12/CAMP3/pages/VSCC.htm>. [Online]. Available: <http://www-nrd.nhtsa.dot.gov/pdf/nrd-12/CAMP3/pages/VSCC.htm>
- [4] "CVIS Project," [http://www.ertico.com/en/activities/efficiency\\_environment/cvis.htm](http://www.ertico.com/en/activities/efficiency_environment/cvis.htm).
- [5] "SafeSpot Project," <http://www.safespot-eu.org/>.
- [6] "Common Criteria Portal," <http://www.commoncriteriaportal.org/>.
- [7] "OCTAVE Information Security Risk Evaluation," <http://www.cert.org/octave/>.

- [8] J.-P. Hubaux, S. Čapkun, and J. Luo, “The security and privacy of smart vehicles,” *IEEE Security and Privacy*, vol. 4, no. 3, pp. 49–55, 2004. [Online]. Available: <http://icawww.epfl.ch/Publications/luo/HubauxCL04.pdf>
- [9] M. Raya and J.-P. Hubaux, “The security of vehicular ad hoc networks,” in *Proc. of Third ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2005)*, Alexandria, USA, Nov. 2005. [Online]. Available: <http://icawww.epfl.ch/Publications/raya/RayaH05C.pdf>
- [10] A. Aijaz, B. Bochow, F. Dötzer, A. Festag, M. Gerlach, R. Kroh, and T. Leinmüller, “Attacks on inter vehicle communication systems - an analysis,” in *Int'l Workshop on Intelligent Transportation (WIT)*, Mar. 2006.
- [11] B. Schneier, “Attack Trees,” *Dr. Dobbs's Journal*, vol. 24, pp. 21–29, Dec. 1999. [Online]. Available: <http://www.ddj.com/documents/s=896/ddj9912a/9912a.htm>
- [12] M. Toms, M. Cummings-Hill, D. Curry, and S. Cone, “Using cluster analysis for deriving menu structures for automotive mobile multimedia applications,” in *SAE 2001*, 2001.
- [13] M. Bishop, *Computer Security: art and science*. Pearson Education, Inc., 2003.
- [14] R. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons Inc, 2001.
- [15] C. Eckert, *IT-Sicherheit: Konzepte, Verfahren, Protokolle*. R. Oldenbourg Verlag, 2006.
- [16] B. Nuseibeh and S. Easterbrook, “Requirements engineering: a roadmap,” in *ICSE - Future of SE Track*, 2000, pp. 35–46. [Online]. Available: [citeseer.ist.psu.edu/nuseibeh00requirements.html](http://citeseer.ist.psu.edu/nuseibeh00requirements.html)
- [17] P. T. Devanbu and S. G. Stubblebine, “Software engineering for security: a roadmap,” in *ICSE - Future of SE Track*, 2000, pp. 227–239. [Online]. Available: [citeseer.ist.psu.edu/devanbu00software.html](http://citeseer.ist.psu.edu/devanbu00software.html)
- [18] J. N. och Dagl et al., “Evaluating automated support for requirements similarity analysis in market-driven development,” in *REFSQ 01*, 2001. [Online]. Available: [citeseer.ist.psu.edu/nattochdag01evaluating.html](http://citeseer.ist.psu.edu/nattochdag01evaluating.html)
- [19] S. Park, H. Kim, Y. Ko, and J. Seo, “Implementation of an efficient requirements-analysis supporting system using similarity measure techniques,” *Information and Software Technology*, vol. 42, no. 6, pp. 429–438, 2000. [Online]. Available: [citeseer.ist.psu.edu/park00implementation.html](http://citeseer.ist.psu.edu/park00implementation.html)
- [20] J. Palmer and Y. Liang, “Indexing and clustering of software requirements specifications,” *Information and Decision Technologies*, vol. 18, no. 4, pp. 283–299, 1992.
- [21] V. S. C. P. (VSC), “Task 3 final report: Identify intelligent vehicle safety applications,” U.S. Department of Transportation, Tech. Rep., Mar. 2005. [Online]. Available: <http://www-nrd.nhtsa.dot.gov/pdf/nrd-12/1665CAMP3web/images/CAMP3scr.pdf>
- [22] F. Doetzer, T. Kosch, and M. Strassberger, “Classification for traffic related inter-vehicle messaging,” in *Proceedings of the 5th IEEE International Conference on ITS Telecommunications*, Brest, France, June 2005. [Online]. Available: <http://www13.in.tum.de/personen/doetzer/publications/Doetzer-05-ClassificationTrafficRelatedMessaging.pdf>
- [23] “SEVECOM - Secure Vehicle Communications Project,” <http://www.sevecom.org/>.

## APPENDIX

## Application use case

<b>Use Case</b>	Vehicle-based road condition warning
<b>Creator</b>	Frank Kargl, UULM
<b>Goal in Context</b>	Vehicles that detect hazardous road conditions send warnings to other approaching vehicles, so that their drivers can adapt their behaviour accordingly.
<b>Scope &amp; Level</b>	Application use case
<b>Preconditions</b>	None
<b>Success End Condition</b>	Drivers receive warnings before reaching hazardous road segments
<b>Failed End Condition</b>	System fails to warn drivers
<b>Involved components</b> (Any logical components, both hardware and software that are involved in application implementation)	Sensors for detection of hazardous road conditions, e.g. - ABS, ASR, or ESP/VSC sensors can detect slippery or icy roads - rain sensors that are used for starting the wipers can detect wet roads On-board processing and wireless communication units
<b>Trigger</b>	Sensors detecting potential hazardous road conditions
<b>Operation description</b> (Complete textual description of application operation)	Sensors constantly monitor road conditions and create a risk-estimation for multiple classes of hazards (e.g. slippery road, wet road, strong wind, ...). When at least one of these parameters exceeds a given threshold, the car starts emitting geocast messages that are sent to all nearby road segments which lead to this position. The messages contain the risk-estimations for all hazard-classes. Vehicles receiving such a message will forward the message according to the general geocast-/relevancy-based-forwarding strategy. Vehicles receiving such a message will additionally issue a optical/acoustical warning to the driver. Options: - The warning might be modulated according to the estimated strength of the hazard contained in the message. - Vehicles may apply consistency checks with own sensors or messages received from other cards to detect false-alarms.

<b>Characteristics</b>										
<b>Safety relation</b>	No relation			Safety relevant	X		Safety critical			
<b>In-car system</b>										
<b>Driver involvement</b>										
<b>Communication</b>	C2C		X	C2I			I2C			
	One-way	X	Two-way		Single-Hop		Multi-Hop	X		
	Unicast		Broadcast		Geocast	X	Relevancy	X		
<b>Timing</b>	Timing constraints		5s	Periodic messages			X			
<b>Security requirements</b>										
<b>ID Authentication</b>	0									
<b>Property auth.</b>	2									
<b>Location auth.</b>	2									
<b>Integrity</b>	2									
<b>Confidentiality</b>	0									
<b>ID privacy</b>	2									
<b>Location privacy</b>	0									
<b>Jurisdiction. Access</b>	1									
<b>Availability</b>	1									
<b>Access control</b>	0									
<b>Auditability</b>	0									

Fig. 2. Example Application Use Case

## Attack use case

<b>Use Case</b>	Forging of Warning Messages						
<b>Related appl. use case</b>	Vehicle-based road condition warning						
<b>Creator</b>	Frank Kargl, UULM						
<b>Primary Attack Goal</b>	DoS	X	Inform. Theft		Intrusion		Tampering
<b>Used Techniques</b>	Masquer.		Eavesdrop.		Auth. Violation		Loss/Modific.
	Repudiat.		Forgery	X	Sabotage		
<b>Goal in Context</b> (Textual description of attackers goal/motivation)	Issue false warnings so that drivers get irritated and may go slower than necessary. Due to hard braking, rear-end collisions may occur.						
<b>Attacked components</b> (Any logical components, either hardware, software, or user, that are targeted by this attack)	Wireless communication						
<b>Pre-requirements for attack</b>	Wireless communication equipment, capable of creating and sending forged messages						
<b>Attack description</b> (Complete textual description of attack operation)	<p>Attacker places itself near the target area and emits forged messages warning e.g. because of slippery or icy road conditions. The destination area for the geocast may be selected based on topographic features or simply set to a maximum area so that as many cars as possible will be affected.</p> <p>Messages will be automatically distributed in the destination region and drivers will receive warning messages, to whom they are supposed to react accordingly.</p>						
<b>Attack success factors</b> (Reasons why attack may succeed)	Drivers will recognize the warning and slow down.						
<b>Attack failure factors</b> (Reasons why attack may fail)	<p>If there are no cars in the one-hop neighbourhood to distribute the messages, the attack fails.</p> <p>Drivers might simply ignore the warnings.</p>						
<b>Effects of attack</b> (regarding driver and road traffic)	The attack will cause the drivers to slow down, causing traffic jams or in worst case rear-end collisions.						
<b>Severity</b>	low	X	medium		high		fatal

Fig. 3. Example Attack Use Case