A Security Credential Management System for V2X Communications

Benedikt Brecht, Dean Therriault, André Weimerskirch, William Whyte, Virendra Kumar, Thorsten Hehn, and Roy Goudy

***Abstract*— The U.S. Department of Transportation (USDOT) issued a proposed rule on January 12, 2017 to mandate vehicle-to-vehicle safety communications in light vehicles in the U.S. Cybersecurity and privacy are major challenges for such a deployment. We present a Security Credential Management System (SCMS) for vehicle-to-everything (V2X) communications in this paper, which has been developed by the Crash Avoidance Metrics Partners LLC under a cooperative agreement with the USDOT. This system design is currently transitioning from research to proof-of-concept and is a leading candidate to support the establishment of a nationwide Public Key Infrastructure for V2X security. It issues digital certificates to participating vehicles and infrastructure devices for trustworthy communications among them, which is necessary for safety and mobility applications that are based on V2X communications. The SCMS supports four main use cases, namely, bootstrapping, certificate provisioning, misbehavior reporting, and revocation. The main design goal is to provide both security and privacy to the largest extent reasonable and possible. To achieve a reasonable level of privacy in this context, vehicles are issued pseudonym certificates, and the generation and provisioning of those certificates are divided among multiple organizations. Given the large number of pseudonym certificates per vehicle, one of the main challenges is to facilitate efficient revocation of misbehaving or malfunctioning vehicles, while preserving privacy against attacks from insiders. The proposed SCMS supports all identified V2X use-cases and certificate types necessary for V2X communication security. This paper is based upon work supported by the USDOT. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors ("we") and do not necessarily reflect the view of the USDOT.**

***Index Terms*— Device-to-device communication, vehicle safety, cryptography, intelligent transportation systems.**

Manuscript received September 17, 2017; accepted December 26, 2017. This work was supported by cooperative agreements USDOT and CAMP VSC3 under Grant DTNH22-14-H-00449/0003 and USDOT and CAMP VSC5 under Grant DTNH2214H00449/0005L. The Associate Editor for this paper was P. Kachroo. *(Corresponding author: Benedikt Brecht.)*

B. Brecht is with Volkswagen Group of America Inc., Auburn Hills, MI 48326 USA (e-mail: benedikt.brecht@vw.com).

D. Therriault is with General Motors Corporation, Detroit, MI 48265 USA (e-mail: dean.therriault@gm.com).

A. Weimerskirch is with Lear Corporation, Southfield, MI 48033 USA (e-mail: aweimerskirch@lear.com).

W. Whyte and V. Kumar are with OnBoard Security, Wilmington, MA 01887 USA (e-mail: wwhyte@onboardsecurity.com; vkumar@onboardsecurity.com).

T. Hehn is a Private Person (e-mail: thehn@gmx.de).

R. Goudy is with Nissan Group of North America, Farmington Hills, MI 48331 USA (e-mail: goudyr1@nrd.nissan-usa.com).

Crash Avoidance Metrics Partners LLC.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2018.2797529

I. Introduction

VEHICLE-to-Vehicle (V2V) communications between nearby vehicles in the form of continuous broadcast of Basic Safety Messages (BSMs) has the potential to reduce unimpaired vehicle crashes by 80% through active safety applications [2]. Following a series of field operational tests, the US Department of Transportation (USDOT) issued a proposed rule on January 12th, 2017 to mandate the inclusion of V2V technology in light vehicles in the US [3]. Vehicles will broadcast BSMs up to ten times per second to support V2V safety applications. BSMs include the senders' time, position, speed, path history and other relevant information, and are digitally signed. The receiver evaluates each message, verifies the signature, and then decides whether a warning needs to be displayed to the driver. The correctness and reliability of BSMs are of prime importance as they directly affect the effectiveness of safety applications based on them. To prevent an attacker from inserting false messages, the sending vehicles digitally sign each BSM, and the receiving vehicles verify the signature before acting on it. This approach has been recommended by many different studies of the system in both Europe and America [4]–[10].

A Public-Key Infrastructure (PKI) that facilitates and manages digital certificates is necessary for building trust among participants and for proper functioning of the system. The Security Credential Management System (SCMS) proposed in this paper implements a PKI with unique features. This SCMS design is currently a leading candidate for the V2X security backend in the US. It is distinguished from a traditional PKI in several aspects. The two most important aspects being its size (i.e., the number of devices that it supports) and the balance among security, privacy, and efficiency. At its full capacity, it will issue approximately 300 billion certificates per year[1] for 300 million vehicles. To the best of our knowledge, the PKI whose root is run by the Europay-Mastercard-Visa Consortium (EMVCo) is the largest existent PKI in the world and issues in the single-digit billions of certificates per year [11], while the largest current government-run PKI, deployed by the Defense Information Systems Agency for the Common Access Cards program [12], is several orders of magnitude smaller and issues under 10 million certificates per year. At the core of its design, the proposed SCMS has several novel cryptographic constructs to provide a high

[1]This number may be even greater if pedestrian and cyclist-borne units become part of the system.

level of security and privacy to the users while keeping the system very efficient. As a result, the presented SCMS design is significantly different from any previously implemented PKI. However, it is somewhat similar to the design of the European C2X PKI [4]. The SCMS was designed with privacy (both against SCMS insiders and outsiders) being the highest priority. The SCMS design also provides efficient methods for requesting certificates and handling revocation. An early version of the proposed SCMS has already been implemented, operated and tested in the Safety Pilot Model Deployment [13], and the design presented here has been specified and implemented in the SCMS Proof-of-Concept (PoC) Project and is going to be deployed for the USDOT's Connected Vehicle Pilot Program (CV Pilot Program) [3]. Besides V2V safety applications, there will also be vehicle-to-infrastructure and infrastructure-to-vehicle (V2I / I2V) applications[2] to support safety, mobility and environmental applications. While V2V safety applications are the current focus of research and deployment in the US, it is essential for the SCMS to also be able to cover V2I applications. A wide variety of V2I applications were analyzed and categorized, and the SCMS design was updated to support all identified V2I application categories. This includes infrastructure-originating broadcast messages (e.g., traffic light announcements) as well as service announcement and provisioning (e.g., Internet access). The first application category requires that road-side equipment (RSE) authenticates broadcast messages, whereas the second application category requires that on-board equipment (OBE) in the vehicle establishes a communication channel to the RSE.

The interface specification of the current version of the SCMS is available from [14]. This paper along with [15] introduces several novel concepts including (1) distributed provisioning of certificates for privacy protection against insider attacks, (2) butterfly keys for communication-efficient request of an arbitrarily large number of certificates by a device, (3) linkage values for efficient revocation of seemingly unrelated pseudonym certificates of a device, and (4) elector ballots for management of root certificate authority and electors. All of these concepts were developed across several projects consisting of a diverse group of participants at CAMP, and the authors of this paper don't claim sole authorship for any of them. The goal of this paper is to provide an overview of the features of the SCMS along with rigorous arguments for the appropriateness of the design decisions made. There have been many alternate designs proposed in research and in industry; this paper does not aim to provide a comprehensive comparison of alternate designs, but such a comparison can be found in [16]. We provide a list of acronyms used throughout this paper as an online resource given by Crash Avoidance Metrics Partners LLC.

II. SCMS Design Overview

In this section, we present the design of the SCMS by briefly explaining its components and then discussing the rationale behind them. We say that an SCMS component is *intrinsically-central*, if it can have exactly one distinct instance for proper functioning. A component is *central*, if it is chosen to have exactly one distinct instance in the considered instantiation of the system. Distinct instances of a component have different identifiers and do not share cryptographic materials. While there is only one SCMS, components that are not central can have multiple instances. It is assumed that all components support load balancing if needed. Figure 1 gives an overview of the overall system architecture. The lines connecting different SCMS components in Figure 1 are relationship lines, meaning that in at least one of the use cases, one component sends information or certificates to the other. The SCMS was originally designed for V2V use cases [15], but later extended to support V2I as well. We will later present the different types of certificates required to support both V2V and V2I applications. In the SCMS, there are components that are included merely for V2V services (e.g., Linkage Authority), for generic V2X services (e.g., Intermediate CA), and for combined V2V and V2I services with separate features for V2V and V2I, respectively (e.g., Pseudonym CA). Additionally, there is a dotted line connecting a device to the SCMS to indicate out-of-band secure communication. The SCMS can be simplified at the expense of losing some flexibility, e.g., by making all the components *central*.

A. Threat Models and Application Concepts

An (unpublished) risk assessment was performed for V2V safety applications, which at the moment are being considered for driver notification only (i.e., not for control, and not for traffic management or other mobility applications). For instance, a V2V safety application might be designed to provide a warning to the driver in case of an imminent forward collision but such an application will not be able to control the car based on the wireless Dedicated Short Range Communication (DSRC) message input alone. It is assumed that the cryptographic private keys and other security- and privacy-sensitive materials are stored and used securely in secure hardware. The risk assessment concluded the major risk to be that of users deactivating the DSRC system in their cars in case they receive too many false warnings due to bogus messages broadcast by malicious devices. Since V2V is a collaborative technology that requires participation from a majority of vehicles, the system can be considered a failure if too many nodes were to deactivate for any reason.

The risk assessment also concluded that there is no safety-of-life risk from security hacking attacks on the SCMS, since the result of hacking the SCMS is that an attacker can either send false messages or create a denial of service attack on the system (e.g., by widespread illegitimate revocation of devices). Neither of these hacks are a direct risk to safety-of-life in the sense of making collisions more likely than they would be in the absence of the system given the above assumption about driver notification applications. However, these hacks would clearly reduce the safety-of-life benefits of the system, and as such, the security of the SCMS components is very important to the system's success.

Cybersecurity aspects of the vehicle on-board DSRC computing platform and the SCMS components are excluded in this paper. However a secure design and separation of safety-critical systems from the rest of the vehicle systems are

[2]For the sake of a compact representation, we will denote V2I / I2V applications as V2I applications

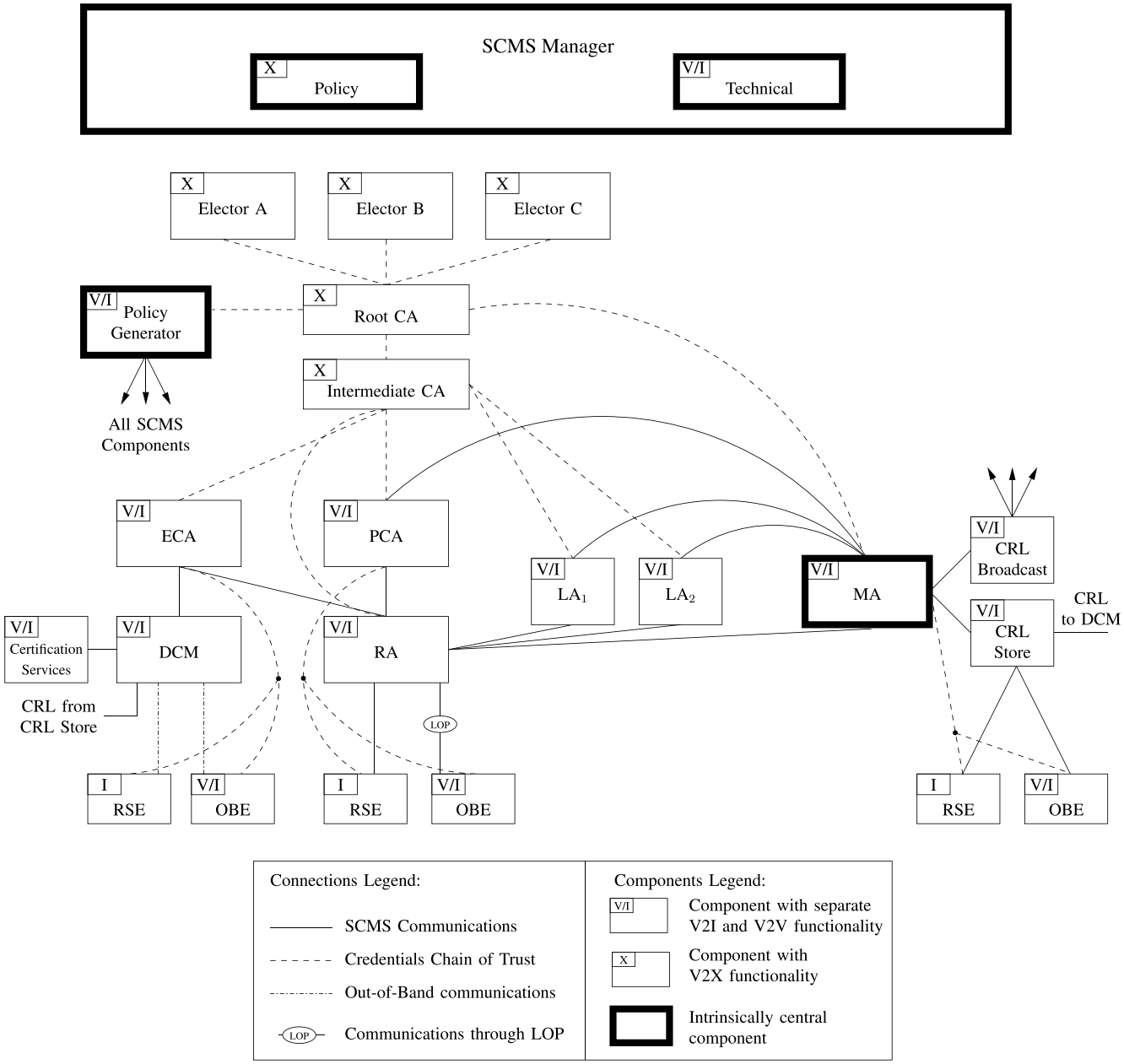

Fig. 1. SCMS architecture overview.

needed. We believe that the cybersecurity concerns of DSRC units are similar to other wireless interfaces such as cellular and Bluetooth.

Furthermore, the risk assessment concluded that there are risks to privacy from security attacks by SCMS outsiders as well as insiders. It was thus concluded that the SCMS must counter / mitigate the following types of attacks:

- Attacks on end-users' privacy from SCMS insiders
- Attacks on end-users' privacy from outside the SCMS
- Authenticated bogus messages leading to false warnings

We address the first two items listed above by what we call "Privacy by Design." The third item is addressed by misbehavior detection (to identify misbehaving and malicious devices) and efficient revocation (to reject messages from revoked devices) to avoid / minimize the harm to the system's trustworthiness. While the future risk assessments are expected to provide a better understanding of the properties and parameters of the system, the types of certificates to be issued by the SCMS are considered to be stable at the moment. For the design of the SCMS, we assume that today's cryptographic mechanisms are acceptably secure. This assumption becomes incorrect if quantum computers of sufficient scale are developed as they could break the Elliptic Curve Digital Signature Algorithm (ECDSA) for all practical curve sizes. The design is modular and flexible, which allows an upgrade to post-quantum cryptographic algorithms once such algorithms are widely accepted and standardized. Note that certain solutions need to be redesigned for post-quantum algorithms, such as the Butterfly key expansion (see Section IV) since it assumes

a cryptographic algorithm based on the discrete logarithm problem.

1) Privacy by Design: A key goal of the security system is to protect the privacy of end users, in particular people driving vehicles for private use. Since most privately owned vehicles are associated with a single registered owner, the ability to track the vehicle may be used to associate vehicle operation with the registered owner, and so the system is designed to make it hard to track that vehicle based on its data transmissions. This is achieved in two ways:

1) For future applications that involve sending unicast or multicast (as opposed to broadcast) messages, the application design should use encryption and other mechanisms to prevent the identity of parties to the communication being leaked. Additionally, those applications should be opt-in so that participants have the opportunity to assess the trade-off between privacy and the value of the service offered and only use the application if it offers a good trade-off.[3] These privacy-preserving technologies can be implemented without having significant design implications for the SCMS.
2) For applications that involve sending broadcast messages from end-user vehicles, the privacy-preserving aspects of the SCMS defined below are used to make it difficult for eavesdroppers in two physically distant locations to tell whether BSMs transmitted at the two locations originated from the same vehicle.

We identify two types of attackers, inside attackers and outside attackers. An outside attacker has access to BSMs but not to any other information such as certificates that have not been broadcast yet. An inside attacker has access to BSMs and to other information, such as information generated during the certificate issuance process. To maintain privacy against outside attackers, we propose that end-entity devices are issued with a large number of certificates (we quantify this in section II-C) and that they make frequent changes in the certificates accompanying BSMs (e.g., every 5 minutes, or as specified in [17] – note that the SCMS supports a range of strategies and that the change strategy is ongoing research from a privacy perspective [16], [18]). To provide defense against inside attackers, the SCMS operations are divided among its components, and those components are required to have organizational separation between them (i.e., each component is run by a separate organization such that information sharing between organizations can be controlled). The SCMS is designed such that at least two of its components need to collude to gain meaningful information for tracking a device.

We define "unlinkability" informally (cf., [19, Sec. 4] for a more formal definition) as the concept that the greater the distance in time and space between two transmissions from the same device, the harder it is to determine that those two transmissions did in fact come from the same device.

Unlinkability is not a binary property of the system. For example, an eavesdropper who is able to record all messages sent by a vehicle will be able to track that vehicle by constructing the path indicated by that vehicle's BSMs. However, it is a design goal that the V2V communications system does not increase the risk that an individual may be tracked.

For purposes of the SCMS described in this paper, the requirement is that if a vehicle's broadcast messages (a) contain data that is unique to the vehicle and (b) can be linked to a location, the data should change frequently so that it is extremely difficult for an eavesdropper to track that vehicle.

Note also that any mobile device may potentially be tracked in other ways that are not addressed by the SCMS (or by any application or data-level mechanism) and are out of scope of this research, e.g., by RF fingerprinting [20] or by deploying a large-scale sniffer network. Additional technical mechanisms and possibly legal regulations are required to counter such attacks.

2) Misbehavior Detection & Revocation: The SCMS enforces revocation by periodically distributing a Certificate Revocation List (CRL) with an updated list of entries. Devices use the CRL to identify and reject messages from revoked devices. In addition, the SCMS maintains *internal* blacklists of revoked devices to deny future certificate requests by them. For V2V safety applications, each device receives a multitude of certificates so that traditional CRLs would not work in our context since they would grow too large. We make the revocation of devices efficient by using a novel concept of *linkage values* (cf., Section V-B), which is an extension of similar ideas presented in [21]. For other applications, traditional CRLs are supported as well. It is foreseen that CRL entries will contain some indicator of priority or severity so that recipients who are limited in storage capacity can store only the k entries of greatest interest to them, with the minimum value of k system-wide set to 10,000. Section VI-F discusses the CRL size and presents ideas for its dissemination.

Misbehavior detection is the name we give the process of identifying devices that are either misbehaving or malfunctioning. It requires *local* misbehavior detection in vehicles to recognize anomalies, misbehavior reporting by devices to the SCMS, and *global* misbehavior detection by the SCMS to analyze misbehavior reports and to decide which devices to revoke. The local and global misbehavior detection algorithms are the focus of on-going work and are not further discussed in this paper.

B. SCMS Structure

In this section we briefly describe the different SCMS components. Figure 1 shows components within the system by their logical roles. An implementation of the system may combine multiple logical roles within a single organization with proper separation of the logical roles. Components marked V/I provide separate V2V and V2I functionality while components marked 'X' provide general functionality for the whole V2X system. Figure 1 shows three pairs of RSEs and OBEs. These are of the same type and used to illustrate different use cases of the SCMS. The leftmost pair is used to demonstrate the connections required for bootstrapping, the pair in the middle shows the connections required for certificate provisioning and misbehavior reporting, and the rightmost pair shows the

[3] In practice, it is not clear that end-users are skilled in evaluating these tradeoffs, but this approach is widely accepted in principle.

connections required for retrieval of the CRL via the CRL Store. The content sent over the connections originating from and ending at the rightmost pair has a special property: Even if a device is involved only in V2V activities, it may report misbehavior and receive revocation information on devices involved in V2I activities (such as an infrastructure component) and vice versa. There are four types of connections in the SCMS:

- Solid lines Solid lines represent regular, secure communications, including certificate bundles
- Dashed lines represent the credentials chain of trust. This line shows the chain of trust for signature verification. Enrollment certificates are verified against the ECA certificate, pseudonym, application, and identification certificates are verified against the PCA certificate, and certificate revocation lists are verified against the CRL Generator (part of the MA) certificate. This line does not imply data transfer between the components connected by this line.
- Dash-Dotted lines represent Out-of-Band communications, e.g., the line between the RSE and the DCM. This will be explained in more detail in Section V-A.
- Connections marked with LOP go through the Location Obscurer Proxy (LOP). The Location Obscurer Proxy is an anonymizer proxy stripping all location-related information from requests.

All online components communicate with each other using a protected and reliable communication channel, utilizing protocols such as those from the Transport Layer Security (TLS) suite [22]. There is an air-gap between some components and the rest of the system (e.g., Root CA, Electors). Data is encrypted and authenticated at the application layer if it is forwarded via an SCMS component that is not intended to read that data (e.g., data generated by the linkage authority that is addressing the Pseudonym CA but routed via Registration Authority). The following components are part of the SCMS design:

- SCMS Manager: Ensures efficient and fair operation of the SCMS, defines organizational and technical policies, and sets guidelines for reviewing misbehavior and revocation requests to ensure that they are correct and fair according to procedures.
- Certification Services: Specifies the certification process and provides information on which types of devices are certified to receive digital certificates.
- CRL Store: A simple pass-through component that stores and distributes CRLs.
- CRL Broadcast: A simple pass-through component that broadcasts the current CRL through, e.g., RSEs or satellite radio system.
- Device: An end-entity (EE) device that sends or receives messages, e.g., an OBE, an after-market safety device (ASD), an RSE, or a Traffic Management Center (TMC) backend.
- Device Configuration Manager (DCM): Attests to the ECA that a device is eligible to receive enrollment certificates, and provides all relevant configuration settings and certificates during bootstrapping.
- Electors: Electors represent the center of trust of the SCMS. Electors sign ballots that either endorse or revoke an RCA or another elector. The SCMS Manager distributes those ballots to all SCMS components, including devices, to establish trust relationships in RCAs and electors. An elector has a self-signed certificate, and all entities of the system will implicitly trust the initial set of electors. Therefore, all entities have to protect electors against unauthorized alteration, once they installed the initial set.
- Enrollment CA (ECA): Issues enrollment certificates, which act as a passport for a device to authenticate against the RA, e.g., when requesting certificates. Different ECAs may issue enrollment certificates for different geographic regions, manufacturers, or device types.
- Intermediate CA (ICA): This component serves as a secondary Certificate Authority to shield the root CA from traffic and attacks. The Root CA issues the Intermediate CA certificate.
- Linkage Authority (LA): Generates pre-linkage values, which are used to form linkage values that go in the certificates and support efficient revocation. There are two LAs in the SCMS, referred to as LA_1 and LA_2. The splitting prevents the operator of an LA from linking certificates belonging to a particular device.
- Location Obscurer Proxy (LOP): Hides the location of the requesting device by changing source addresses, and thus, prevents linking of network addresses to locations.
- Misbehavior Authority (MA): Processes misbehavior reports to identify potential misbehavior or malfunctioning by devices, and revokes and adds them to the CRL, if necessary. It also initiates the process of linking a certificate identifier to the corresponding enrollment certificates and adding them to the RA's internal blacklist. The MA contains two subcomponents: Global Misbehavior Detection, which determines which devices are misbehaving; and CRL Generator (CRLG), which generates, digitally signs and releases the CRL to the public.
- Policy Generator (PG): Maintains and signs updates of the Global Policy File (GPF), which contains global configuration information, and the Global Certificate Chain File (GCCF), which contains all trust chains of the SCMS.
- Pseudonym CA (PCA): Issues short-term pseudonym, identification, and application certificates to devices. Individual PCAs may be, e.g., limited to a particular geographic region, a particular manufacturer, or a type of device.
- Registration Authority (RA): Validates and processes requests from devices. From those, it creates individual requests for certificates to the PCA. The RA implements mechanisms to ensure that revoked devices are not issued new certificates, and that devices are not issued more than one set of certificates for a given time period. In addition, the RA provides authenticated information about SCMS configuration changes to devices, which may include a component changing its network address or certificate, or relaying policy decisions issued by the SCMS Manager. Additionally, when sending pseudonym

certificate signing requests to the PCA or forwarding information to the MA, the RA shuffles the requests/reports to prevent the PCA from taking the sequence of requests as an indication for which certificates may belong to the same batch and the MA from determining the reporters' routes.

- Root Certificate Authority (RCA): An RCA is the root at the top of a certificate chain in the SCMS and hence a trust anchor in a traditional PKI sense. It issues certificates for ICAs as well as SCMS components like PG and MA. An RCA has a self-signed certificate, and a ballot with a quorum vote of the electors establishes trust in an RCA. RCA certificates must be stored in secure storage that is usually referred to as a Trust Store. An entity verifies any certificate by verifying all certificates along the chain from the certificate at hand to the trusted RCA. This concept is called chain-validation of certificates and is the fundamental concept of any PKI. If the RCA and its private key are not secure, then the system is potentially compromised. Due to its importance, an RCA is typically off-line when not in active use.

Note that the MA, PG, and the SCMS Manager are the only intrinsically-central components of the SCMS.

C. Certificate Provisioning Model

The focus in this section is on the provisioning of pseudonym certificates to OBEs. The provisioning of other certificate types are either straight-forward, or represent subsets of the pseudonym certificate provisioning process. We have developed a pseudonym certificate provisioning model that balances several conflicting requirements:

- **Privacy vs. Size vs. Connectivity**: Certificates should be used only for short periods of time for privacy reasons. The devices cannot store a very large number of certificates due to limited memory storage and its cost in a vehicle environment. On the other hand, most vehicles cannot establish frequent connectivity to the SCMS to download new certificates on demand.
- **CRL Size and Retrospective Unlinkability**: The SCMS should be able to revoke misbehaving or malfunctioning devices,[4] but putting all valid certificates of a device on the CRL would make it very large. We have designed a mechanism to revoke a large number of certificates efficiently without revealing certificates that were used by the device before it started misbehaving.
- **Certificate Waste vs. Sybil Attack**: Certificates must be changed periodically for privacy reasons. One option is to have a large number of certificates, each valid one after the other for a short period of time. This would result in a large number of unused certificates.[5] Another option is to have multiple certificates valid simultaneously for longer time periods. This would enable masquerading as multiple devices by compromising a single device (the so-called *Sybil attack* [23]) [16].

In the SCMS model used for Safety Pilot Model Deployment [13], a certificate was valid for a specific 5-minute period, which would correspond to $105,120$ certificates per year. Given the connectivity constraints, in full deployment a device may need up to 3 years' worth of certificates, which would amount to more than $300,000$ certificates. This approach is prohibitively expensive in terms of automotive-grade storage requirements on the device. We studied different variants of this model, but none of them offer a good balance among all the properties listed above. Instead, we recommend to adopt a model originating with the CAR 2 CAR Communication Consortium (C2C-CC) [4], with certain modifications to suit our requirements.

In the C2C-CC model, multiple certificates are valid in a given time period, the certificate validity period is days rather than minutes, and the certificate usage pattern can vary from device to device, e.g., a device could use a certificate for 5 minutes after start-up, then switch to another certificate, and use that for a longer time-period before changing certificates again, or even until the end of the journey. This model offers enough flexibility to find a good balance among our different requirements except "CRL size and retrospective unlinkability", which we address by our construct of linkage values (cf. Section V-B). In particular, our proposal is to use a model originating with the C2C-CC with the following parameter values:

- Certificate validity time period: 1 week
- Number of certificates valid simultaneously (batch size): minimum 20
- Overall covered time-span: $1-3$ years

These parameters can be further refined by the SCMS Manager so that devices are compatible with each other. Two points are worth noting:

1) This model provides a reasonable level of privacy against tracking while keeping the storage requirements low due to a significantly higher utilization of certificates (i.e., far fewer certificates are wasted compared to the Safety Pilot model). Certificates of a device that does not use all its certificates in a week (e.g., say a device only uses 13 of the provided 20 certificates in a given week) cannot be linked. Moreover, if a device re-uses certificates, it is linkable only within a week.
2) The model allows for an easy topping-off mechanism of pseudonym certificates. We have designed a mechanism such that devices do not need to explicitly request new certificates but the SCMS will automatically issue new certificates throughout the life-time of the device, until the device stops picking-up certificates for an extended period of time. The RA will provide the certificates in batches worth one week (e.g., as a zip file), and devices will download the batches using TCP/IP connection. The batches will be organized in files, named using device information and time period. The RA will provide information when to expect new certificate batches

[4]CRLs can be avoided if devices were required to request new certificates frequently such that misbehaving devices could be refused new certificates. This would require frequent connectivity to the SCMS, which may not be a reasonable assumption at least under current circumstances.

[5]Based on the US Census Bureau's annual American Community Survey for 2011, the average daily commute time is less than 1 hour, so more than 95% of certificates would be wasted, if each certificate had a unique validity period.

TABLE I
CERTIFICATE TYPES

			Shuffle	On broadcast CRL if revoked	Simultaneous Validity for given PSID	Linkage Values	Continuous Generation	Issuing certificates for multiple time periods	Pseudonymity	Misbehavior Reporting	Non-Traceability	Encryption
1	OBE Enrollment Certificate								X			
2	OBE Pseudonym Certificate		X	X	X	X	X	X	X	X	X	
3	OBE Identification Certificate						X	X		X		
4	RSE Enrollment Certificate											
5a	RSE Enc. & App Certificate	Online RSE		X						X		X
5b		Offline RSE		X						X		

(e.g., once per month), and the device will have full control over what and how much they download, when they download, and they can also resume interrupted downloads.

III. CERTIFICATE TYPES

Within the overall Connected Vehicle system there are different application types which may have different requirements for certificate management. A complete specification of the SCMS therefore should include determining how many different certificate management process flows need to be supported. We define certificates to be of the same *certificate type* if they are issued following identical process flows. Based on an analysis of 119 DSRC-based applications, it was determined that five end-entity certificate types satisfied all use cases:

- OBE enrollment certificates: enrollment certificates are provided during bootstrap and they are used later to request message signing and/or encryption certificates.
- RSE enrollment certificates: RSE enrollment certificates are the equivalent to OBE enrollment certificates for RSEs.
- OBE pseudonym certificates: pseudonym certificates provide pseudonymity, unlinkability, and efficient revocation of a multitude of certificates. This is accomplished by using security features such as shuffling, linkage values, butterfly key expansion, and encryption of certificates by PCA to the OBE in order to provide protection against an insider attack at the RA. This type of certificate is used to sign basic safety message (BSM) broadcasts by OBEs. Further, this type of certificate is used for authorization purposes when unlinkability is required. In the case of BSM broadcast, pseudonym certificates might be attached to each signed BSM to increase the ability of a receiver to verify each received BSM, or they might be attached to a few signed BSMs only [24].
- OBE identification certificates: identification certificates are used when a device needs to identify itself. This type of certificate does not provide pseudonymity nor unlinkability. During creation of OBE pseudonym certificates, the SCMS applies privacy preserving mechanisms such as shuffling and encryption by the PCA to the OBE. These mechanisms are not necessarily used when creating OBE identification certificates. However, butterfly key expansion is used to allow for continuous generation of certificates. Identification certificates are used for authorization purposes, such as signed authorization messages of an OBE to an RSE to gain access to a barred road.
- RSE application certificates: RSEs use these certificates to sign broadcast messages, to sign service announcement messages, and optionally to provide an encryption key that an OBE can use to send encrypted data. Note that this is the only EE certificate type which includes an encryption key.

The types of certificates provided by the SCMS are listed in Table I, along with a list of features. The table shows required features of each certificate type, where an 'X' means that a certificate type needs to implement that feature. The features are as follows:

- Shuffle: Shuffling is performed in the RA and is useful only in combination with Butterfly keys that are described below. Shuffling makes sure that PCA cannot learn which certificates are assigned to which devices. Shuffling is required to provide privacy against SCMS insiders.
- On broadcast CRL if revoked: certificate information of this certificate type can be added to a CRL that allows easy recognition of revoked certificates. Enrollment certificates and OBE identification certificates are never distributed via broadcast CRL but they are blacklisted at the RA so that they cannot get new certificates.
- Simultaneous validity for given PSID: PSIDs [10] are embedded in certificates and the message payload. They describe application types and enable the receiver to parse a received message. For OBE pseudonym certificates, we allow that more than one certificate is valid per time period, as described in Section II-C (e.g., 20 certificates are valid per week).
- Linkage Values: Linkage values are embedded in certificates and they allow for efficient revocation, cf. Section V-C.

- Continuous Generation: The SCMS will continuously generate certificates once an initial request has been made. This feature allows for devices to quickly obtain certificates once a connection to the SCMS is available and facilitates top-up strategies. This feature is particularly useful if a multitude of certificates is required.
- Issuing certificates for multiple time periods: The SCMS will in one session issue certificates that are valid for more than one time period. For instance, the SCMS might issue certificates each valid for one week, but altogether the certificates cover a period of several years.
- Pseudonymity: A pseudonym certificate does not contain any real-world identifier.
- Misbehavior reporting: Devices provide misbehavior reports to the SCMS in order to enable the SCMS to detect misbehavior and revoke misbehaving devices. All certificate types that are broadcast over-the-air can be included in a misbehavior report.
- Non-traceability: A device receives multiple certificates such that it can use different identities at different times. This raises complexity to trace a device, i.e., to determine whether two broadcast messages originate from the same device based on the embedded cryptographic material.
- Encryption key: Besides the public key for signing, the certificate holds a second public key to encrypt messages to the certificate owner.

Table I shows all types of certificates along with the features they provide. Note the following remarks.

- Due to the large amount of OBE pseudonym certificates per device, linkage values are mandatory for this type. Linkage values are not used in OBE identification certificates since those certificates are only used for V2I applications and since RSE can store large CRLs.
- We suggest the non-traceability requirement for OBE pseudonym certificates only.

IV. Butterfly Key Expansion

The typical process for a device to request certificates from a PKI would be to generate a private/public key pair. The device creates a certificate signing request (CSR) including the public key and provide the CSR to the PKI over a secure channel. The PKI's CA will then sign the certificate and provide it to the requester. For OBE pseudonym certificates, such an approach has disadvantages as thousands of public keys would need to be generated in the device and then sent to the SCMS. Butterfly keys are a novel cryptographic construction to overcome this disadvantage by allowing an OBE to request an arbitrary number of certificates; each certificate with a different signing key and each encrypted with a different encryption key. This is done using a single request that contains only one signing public key seed, one encryption public key seed, and two expansion functions. Certificates are encrypted by the PCA to the OBE to avoid the RA being able to relate certificate content with a certain OBE. Without butterfly keys, the OBE would have to send a unique signing key and a unique encryption key for each certificate. Butterfly keys reduce upload size, allowing requests to be made when there is only suboptimal connectivity, and also reduce the work to be done by the requester to calculate the keys. Further, butterfly keys significantly simplify the topping-off mechanism described in Section II-C. Butterfly expansion for signing key is described below for elliptic curve cryptography, but it could easily be adapted to any discrete-logarithm-type hardness assumption. Butterfly expansion for encryption key is identical to that of the signing key, except for a minor difference in the way the inputs to AES are derived, see below for more details. In the following, we denote integers by lower-case characters and curve points by upper-case characters. The elliptic curve discrete logarithm problem is basically the statement: Given P and $A = aP$, but not a, it is hard to compute the value of a [25]. Butterfly keys make use of this as follows. There is an agreed base point, called G, of some order l. The *caterpillar keypair* is an integer, a, and a point $A = aG$. The certificate requester provides to RA the value A and an expansion function, $\mathrm{f}_k(\iota)$, which is a pseudo-random permutation in the integers mod l. Note that ι is a simple counter iterated by the RA.

In the current design the expansion function for signing keys, $\mathrm{f}_k(\iota)$, which is used to generate points on the NIST curve NISTp256 [26], is defined as

$$\mathrm{f}_k(\iota) = \mathrm{f}_k^{int}(\iota) \bmod l, \text{ where} \tag{1}$$

1) $\mathrm{f}_k^{int}(\iota)$ is the big-endian integer representation of

$$\mathrm{DM}_k(x+1) \parallel \mathrm{DM}_k(x+2) \parallel \mathrm{DM}_k(x+3), \tag{2}$$

2) $\mathrm{DM}_k(m)$ is the AES encryption of m using key k in the Davies-Meyer mode, where the output of the function is XORed with the input to generate the final output, i.e. $\mathrm{DM}_k(m) = \mathrm{AES}_k(m) \oplus m$,
3) $x+1$, $x+2$, and $x+3$ are obtained by simply incrementing x by 1 each time, e.g., if $x = 01\ldots00$, then $x+1 = 01\ldots01$, $x+2 = 01\ldots10$, $x+3 = 01\ldots11$,
4) 128-bit input x for AES is derived from time period $\iota = (i, j)$ as: $\left(0^{32} \| i \| j \| 0^{32}\right)$.

The expansion function for encryption keys is also defined as above except x is derived as: $\left(1^{32} \| i \| j \| 0^{32}\right)$. i is a global value (e.g. representing a week) and j is a counter within i and corresponds to the number of certificates per i (e.g. 20 certificates per week). Both values are set by the SCMS Manager.

Note that in the above definition, AES is used in the Davies-Meyer mode, as f_k does not need to be invertible. Also, AES is applied 3 times to ensure that the outputs of f_k are uniformly distributed with negligible biases, if any.

Now RA can generate up to 2^{128} *cocoon public keys* as $B_\iota = A + \mathrm{f}_k(\iota) * G$, where the corresponding private keys will be $b_\iota = a + \mathrm{f}_k(\iota)$, so the public keys are known to the RA but the private keys are known only to the OBE. The RA includes the cocoon public keys in the certificate requests sent to the PCA.

If these expanded public keys were used unaltered by the PCA, the RA, which knows which public keys come from a single request, could recognize those public keys in the certificates and track the OBE. To avoid this, for each cocoon public key B_ι, the PCA generates a random c_ι and obtains

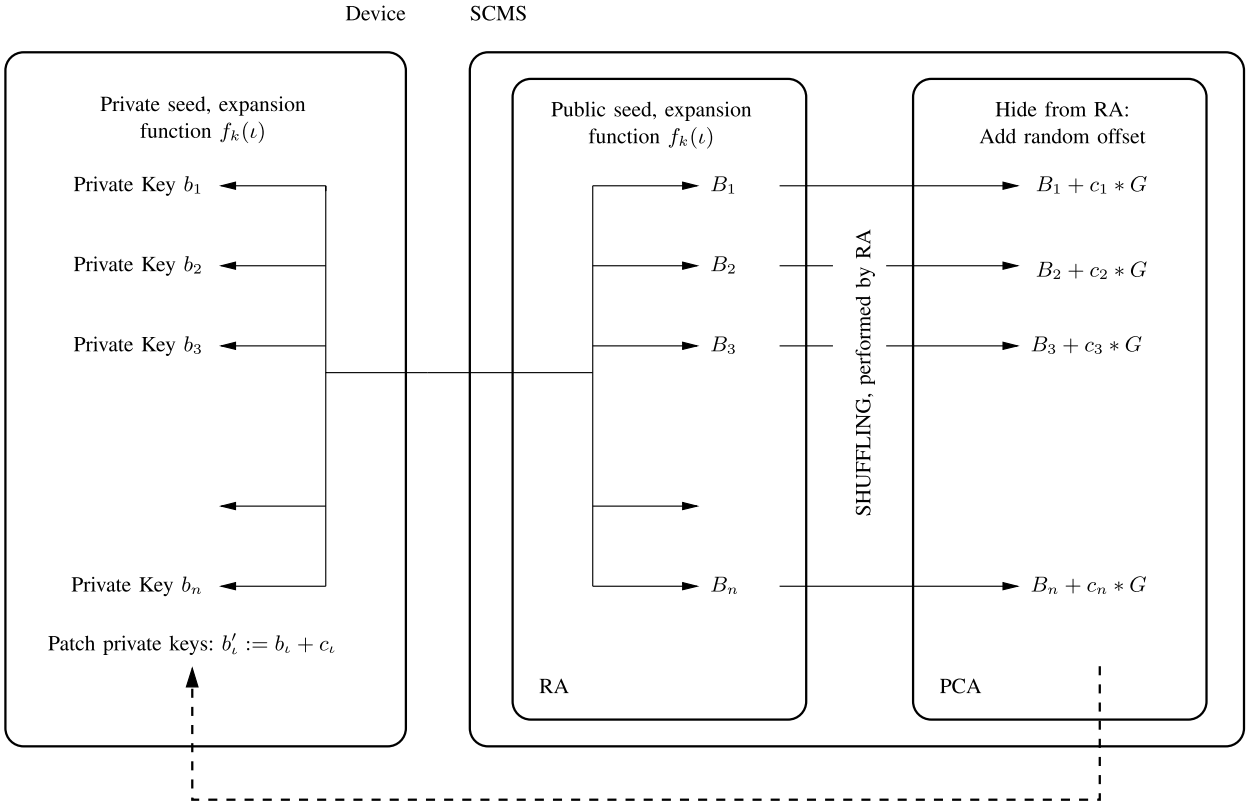

Fig. 2. Butterfly key expansion concept.

$C_\iota = c_\iota G$. The *butterfly public key* which is included in the certificate is $B_\iota + C_\iota$. The PCA returns both the certificate and the private key reconstruction value c to the RA to be returned to the OBE.[6] To prevent the RA from working out which certificate corresponds to a given public key in a request, the certificate and the reconstruction value c_ι are encrypted to the OBE. The OBE will update its private keys resulting in b'_ι with $b'_\iota = b_\iota + c_\iota$. To prevent the PCA from knowing which certificates go to which OBE, each certificate must be encrypted with a different key. The encryption keys are also generated with the butterfly key approach: the OBE provides a caterpillar encryption public key $H = hG$, the RA expands it to cocoon public encryption keys $J_\iota = H + f_e(\iota)G$, and the PCA uses these keys to encrypt the response. Figure 2 provides an overview of the butterfly key expansion concept for signing keys.

A. Security of Butterfly Keys

In this section, we briefly discuss the security of butterfly keys. We say that the above construction of butterfly keys is secure, if any efficient (i.e., polynomial-time) adversary has only a negligible (i.e., smaller than any polynomial) chance of winning in the following game: adversary receives a polynomial number of butterfly public keys (i.e., $(A + \mathrm{f}_k(1) * G + c_1 * G), (A + \mathrm{f}_k(2) * G + c_2 * G), \ldots, (A + \mathrm{f}_k(q) * G + c_q * G)$), and it needs to figure out at least one of the butterfly private keys (i.e., $a + \mathrm{f}_k(x) + c_x$ for any $1 \leq x \leq q$). As f_k is assumed to be a pseudo-random permutation, it is hard for any polynomial-time adversary to distinguish f_k's outputs from truly random values. Hence, we can simplify (without any loss of generality) the above security definition as follows.

Security Definition (Informal): The above construction of butterfly keys is said to be secure, if any efficient adversary that is given a polynomial number of butterfly public keys (i.e., $(A + b_1 * G), (A + b_2 * G), \ldots, (A + b_q * G)$, where $b_1, b_2, \ldots, b_q$ are randomly chosen) has only a negligible probability of correctly guessing one of the butterfly private keys (i.e., $a + b_x$ for any $1 \leq x \leq q$).

Theorem (Informal): The construction of butterfly keys is secure assuming that the elliptic curve discrete logarithm problem is hard.

Proof: We prove the above theorem by contradiction. We first assume that the construction of butterfly keys is *not* secure (i.e., there exists a polynomial-time adversary, henceforth butterfly-keys-adversary, with a non-negligible probability of winning the security game), then using this adversary we construct another polynomial-time adversary, henceforth discrete-log-adversary, that solves the elliptic curve discrete

[6]This description covers explicit certificates, i.e., certificates that include the public key explicitly. In fact, the SCMS as currently implemented issues *implicit certificates* [27] for devices. The implicit certificate generation process inherently changes the public key in a way that leaves input and output uncorrelatable by the RA and is consistent with the motivation for butterfly keys.

logarithm problem with a non-negligible probability, although with a polynomial loss in the winning probability. However, as the theorem says the latter can not be true, the construction of butterfly keys must be secure.

The discrete-log-adversary is given a pair of curve points (P, A), and it needs to output a, s.t. $A = aP$. It randomly selects q integers $b_1, b_2, \ldots, b_q$, and using them generates q butterfly public keys $(A + b_1 * P), (A + b_2 * P), \ldots, (A + b_q * P)$, and gives them to the butterfly-keys-adversary. When the butterfly-keys-adversary returns its response c, the discrete-log-adversary uses c to compute its own response as follows: pick a random number from 1 through q, say y, now the response is $c - b_y$.

It is clear that the discrete-log-adversary runs in polynomial-time if the butterfly-keys-adversary runs in polynomial-time. It remains to be shown that the winning probability of discrete-log-adversary is non-negligible. To this end, we note that if the butterfly-keys-adversary wins, then $c = a + b_x$ for some $1 \leq x \leq q$. Since, y is picked at random from 1 through q after the butterfly-keys-adversary has responded, $y = x$ with probability $1/q$, and hence $c - b_y = a$ with probability $1/q$ times the winning probability of butterfly-keys-adversary. Therefore, if the winning probability of butterfly-keys-adversary is non-negligible, so is that of discrete-log-adversary. ■

V. Device Bootstrapping and Certificate Provisioning

The SCMS supports four primary use cases: device bootstrapping, certificate provisioning, misbehavior reporting, and global misbehavior detection & revocation. Bootstrapping and certificate provisioning will be presented in this section, and misbehavior reporting, detection and revocation will be described in the next section.

A. Device Bootstrapping

The life cycle of a device in the SCMS starts with Bootstrapping. It equips the device with all the information required to communicate with the SCMS and with other devices. It is required that correct information is provided to the device during bootstrapping and that the CAs issue certificates only to certified devices. Any bootstrapping process is acceptable that results in this information being established securely.

The bootstrapping process includes a device, the DCM, the ECA and the certification services component. We assume that the DCM has established communication channels with other SCMS components, such as the ECA or the policy generator, and that it will communicate with the device to be bootstrapped using an out-of-band channel in a secure environment. Bootstrapping consists of two operations: initialization and enrollment. Initialization is the process by which the device obtains certificates it needs to be able to trust received messages. Enrollment is the process by which the device obtains an enrollment certificate that it will need to sign messages to the SCMS.

Information received in the initialization process includes (1) the certificates of all electors, all root CAs, and possibly of intermediate CAs as well as PCAs, and (2) the certificates of the misbehavior authority, policy generator, and the CRL generator to send encrypted misbehavior reports and verify received policy files and CRLs.

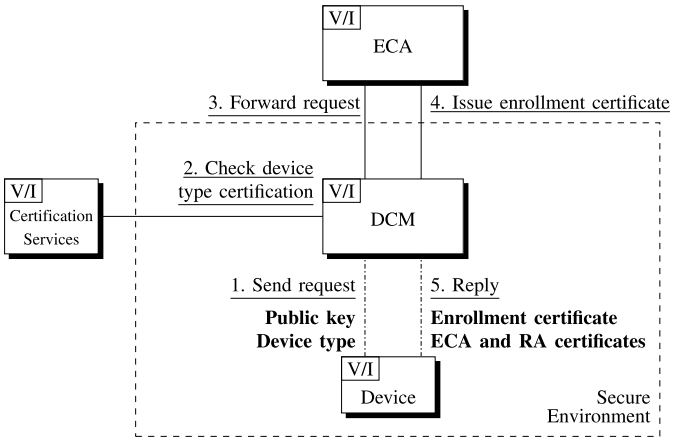

Fig. 3. Exemplary enrollment process.

In the enrollment process, the device receives information required to interact with the SCMS and participate in the V2X communications system. This includes (1) the enrollment certificate, (2) the certificate of the ECA, and (3) the certificate of the RA and other information necessary to connect to the RA. During the enrollment process, the certification services provide the DCM with information about device models which are eligible for enrollment. The DCM must receive trustworthy information about the type of device to be enrolled to ensure only eligible devices are enrolled. Figure 3 shows an exemplary enrollment process.

B. Overview of Certificate Provisioning

The certificate provisioning process for OBE pseudonym certificates is the most complex process in the SCMS because it has to protect end-user privacy and minimize the required computational effort for resource-constrained device. In the following, we focus on the pseudonym certificate provisioning since the certificate provisioning of other certificate types is a straight-forward subset in terms of functionality. Figure 5 illustrates this process, which is designed to protect privacy against inside and outside attackers. The SCMS design ensures that no individual component knows or creates a complete set of data that would enable tracking of a vehicle. The RA knows the enrollment certificate of a device that requests pseudonym certificates, but even though the RA delivers pseudonym certificates to the device, it is not able to read the content of those certificates as the PCA encrypts them to the device. The PCA creates each pseudonym certificate individually, but it does not know the recipient of those certificates, nor does it know which certificates the RA delivers to the same device. The LAs generate masked hash-chain values and the PCA embeds them in each certificate as so called linkage values. The MA unmasks them by publishing a secret linkage seed pair on the CRL, which efficiently links and revokes all future pseudonym certificates of a device. However, a single LA is not able to track devices by linking certificates or to revoke a device, but both LAs, the PCA, and the RA need to collaborate

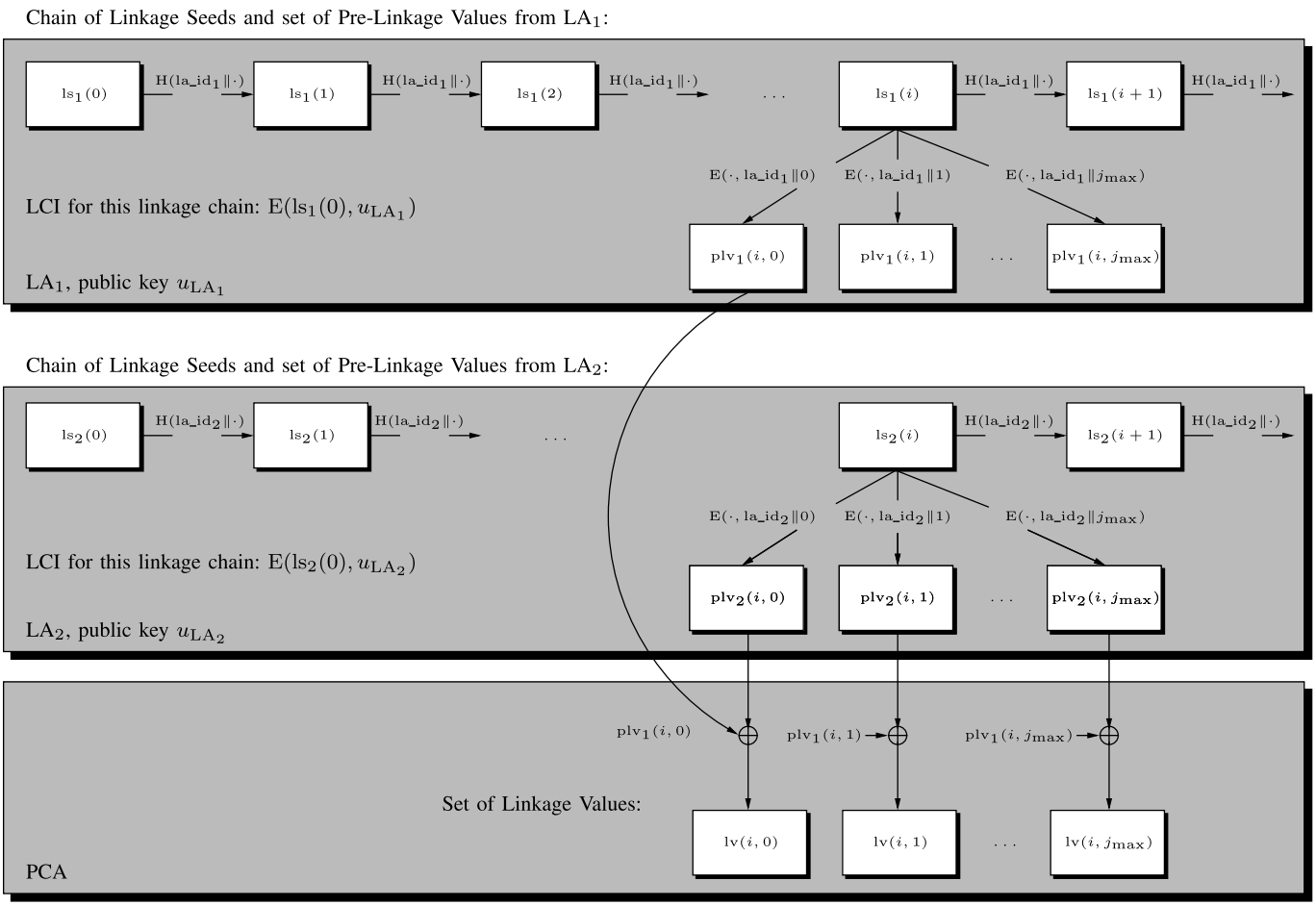

Fig. 4. Creation of linkage values.

for the revocation process. Privacy mechanisms in the SCMS include:

- **Obscuring Physical Location.** The LOP obscures the physical location of an end-entity device to hide it from the RA and the MA.
- **Hiding Certificates from RA.** The *butterfly key expansion* process ensures that no one can correlate the public key seeds in requests with the resulting certificates. Details are given in Section IV. Encrypting the certificates to the device prevents the RA from relating certificates with a device.
- **Hiding Receiver and Certificate Linkage from PCA.** The RA expands incoming requests using Butterfly keys and then splits these requests into requests for individual certificates. It then shuffles requests of all devices before sending them to the PCA. This prevents the PCA from learning whether any two certificate requests belong to the same device, which would violate our privacy goal by enabling the PCA to link certificates. The RA should have configuration parameters for shuffling, e.g., the POC shuffles either 10,000 requests or a day's worth of requests, whatever is reached first.

C. Linkage Values

For any set of pseudonym certificates provided to a device, the SCMS inserts *linkage values* in certificates that can be used to revoke all of the certificates with validity equal to or later than some time i, e.g. the current week. The PCA calculates these linkage values by XORing the pre-linkage values generated by the Linkage Authorities LA_1 and LA_2. The LAs can generate the pre-linkage values in advance. Figure 4 provides an overview of the linkage value generation.

Let $\mathrm{la\_id}_1, \mathrm{la\_id}_2$ be 32-bit identity strings associated with LA_1, LA_2, respectively. For a set of certificates, first the LA_1 (resp., the LA_2) picks a random 128-bit string called the initial linkage seed $\mathrm{ls}_1(0)$ (resp., $\mathrm{ls}_2(0)$), then for each time period (e.g., a week) $i > 0$ calculates the linkage seed $\mathrm{ls}_1(i) \leftarrow \mathrm{H}_u(\mathrm{la\_id}_1 \,\|\, \mathrm{ls}_1(i-1))$ (resp., $\mathrm{ls}_2(i) \leftarrow \mathrm{H}_u(\mathrm{la\_id}_2 \,\|\, \mathrm{ls}_2(i-1))$). In this coherence, $\mathrm{H}_u(m)$ denotes the u most significant bytes of the SHA-256 hash output on m, and $a\|b$ denotes concatenation of bit-strings a and b. We suggest to use $u = 16$. Note that the linkage seeds (i.e., hash chains) created by the LAs have the property that it is easy to calculate forward (i.e., $\mathrm{ls}(i)$ from $\mathrm{ls}(i-1)$) but it is computationally infeasible to calculate backward (i.e., $\mathrm{ls}(i-1)$ from $\mathrm{ls}(i)$). Now pre-linkage values are calculated by means of a pseudorandom function. We choose to implement this by an encryption function, such as AES, in the Davies-Meyer mode. Each LA encrypts the linkage seeds as $\mathrm{plv}_x(i, j) \leftarrow [\mathrm{E}(\mathrm{ls}_x(i), (\mathrm{la\_id}_x \| j)) \oplus (\mathrm{la\_id}_x \| j)]_v$, $x \in \{1, 2\}$, where $\mathrm{E}(k, m)$ is the AES encryption of m with key k, $a \oplus b$ is the exclusive-OR of bit-strings a, b, and $[a]_v$

denotes the υ significant bytes of bit-string a. We suggest flexible use of υ to account for the number of deployed devices and potential weaknesses of the underlying cryptographic primitives in terms of collision resistance. Currently, $\upsilon = 9$ appears to suffice. The value i denotes a time period (e.g., a week) and j denotes certificates within a time period (e.g., 20 certificates per week). Each LA calculates pre-linkage values in the same manner, but each with randomly selected initial seed. We denote the resulting values as plv_1 and plv_2. In order to select a specific linkage chain from an LA, we use Linkage Chain Identifiers (LCI)s. An LCI is the initial linkage seed $\text{ls}_1(0)$ or $\text{ls}_2(0)$ that LA_1 or LA_2, respectively, encrypts to itself, e.g., $\text{E}(pk_1, \text{ls}_1(0))$, where pk_1 is the public key of LA_1.

The LAs encrypt pre-linkage values individually for the PCA but send them to the RA for association with a certificate request. The PCA XORs the pre-linkage values to obtain the linkage value $\text{lv} = \text{plv}_1 \oplus \text{plv}_2$. Similar processing is required when a device processes the CRL. We present the details of this process and the information that the CRLG needs to publish in Section VII.

1) Hiding Linkage Information: The PCA computes the linkage value to be included in a certificate by XORing the two pre-linkage values from the LAs, which the two LAs generate independently and encrypt for the PCA to prevent the RA from colluding with one of the LAs and mapping pre-linkage values to linkage values. Therefore, no single component is able to link pseudonym certificates of a single device.

2) Response Encryption: The PCA creates individual certificates, which the RA collects and provides for download to the device. To prevent the RA from knowing which certificates belong to the same device, the PCA encrypts each individual certificate to the device. The PCA and the device use the butterfly key expansion process to encrypt each certificate with a different key.

3) Linkage Value Length: The linkage values and pre-linkage values are chosen to be 9 bytes in length. This length is appropriate because the unique identifier is not just linkage value but (i, linkage value); this length ensures negligible probability that two devices use the same linkage value in the same time period. Consider that there are $2.5 * 10^8$ cars and each has on average 40 certificates per linkage period (one week), then in any linkage period identified by a given i, there are $10^{10} \approx 2^{33}$ linkage values in use. The relevant quantity when calculating collision probabilities is the number of pairs of values, which in this case is about 2^{66}. With 72-bit linkage values, this means that the chance of a collision between two linkage values is about 2^{-6}. In other words, there will be about one collision every 64 linkage periods, which with linkage period length of a week means about one linkage value collision in the system of 250 million vehicles every year. This is small and in addition a linkage value collision matters only in case of a revocation, i.e., if a linkage value was assigned to a revoked device. Assuming revocation rates below 1%, this implies that a linkage value collision that actually makes a difference will happen less than once every 100 years. This could be reduced further by increasing the size of the linkage value. However, that would increase the size of all certificates, which would increase channel congestion.

D. Misbinding Attacks

In this type of attack, an attacker Mallory misuses the public-key of her target Alice. Mallory reads Alice's public key and requests a certificate over that public key from the SCMS. While Mallory does not know the corresponding private key, she can still mount attacks in which she gets a message signed by Alice's private key and then attaches her certificate rather than Alice's. In the context of the BSM this is not a particularly significant attack, but SCMS-issued certificates could be used for a wide range of applications and in some of those applications the attack could have a greater impact. It is therefore useful to consider countermeasures.

The countermeasure chosen is the one specified in [10]: when a message is signed, the hash that is signed is calculated over both the message itself and the certificate. This binds the hash to the certificate and provides assurance that the certificate provided with the message is in fact the certificate that the sender intended to be used. Since this certificate misbinding attack is only of any use if the false certificate is different from the true one, this approach of including the hash completely eliminates this attack.

Other mechanisms were considered. For example, this attack is possible only if the certificate request messages do not provide proof of possession; and this is the case in the SCMS, since operational certificate requests are not signed with the key for the certificate to be issued but with the key for the enrollment certificate. Signing certificate requests with the enrollment certificate is a key part of the trust architecture of this system and so cannot be changed, but we considered signing the certificate request with both the enrollment certificate and the private key for the certificate to be issued. However, this has shortcomings compared to hashing the certificate into the message. First, if a requester were to legitimately request two certificates for the same key pair, misbinding would still be possible. Second, hashing the certificate into the message allows the receiver to determine for sure that the certificate is the one intended, rather than relying on the CA enforcing a particular mechanism for certificate request. For these reasons we determined that proof-of-possession, although it is widely used in internet protocols, does not add significant value in our system and that including the certificate in the message hash is the only countermeasure necessary.

E. Detailed Description of Pseudonym Certificate Provisioning Process

In the following, we present a detailed step-by-step description of the pseudonym certificate provisioning process, illustrated in Figure 5.

- **Step 1.** The device creates a request by generating butterfly key seeds, signing the request with its enrollment certificate, attaching its enrollment certificate and encrypting the request to the RA. The device then sends the request to the RA via LOP. The LOP functions as a pass-through device for requests. It obscures the device's

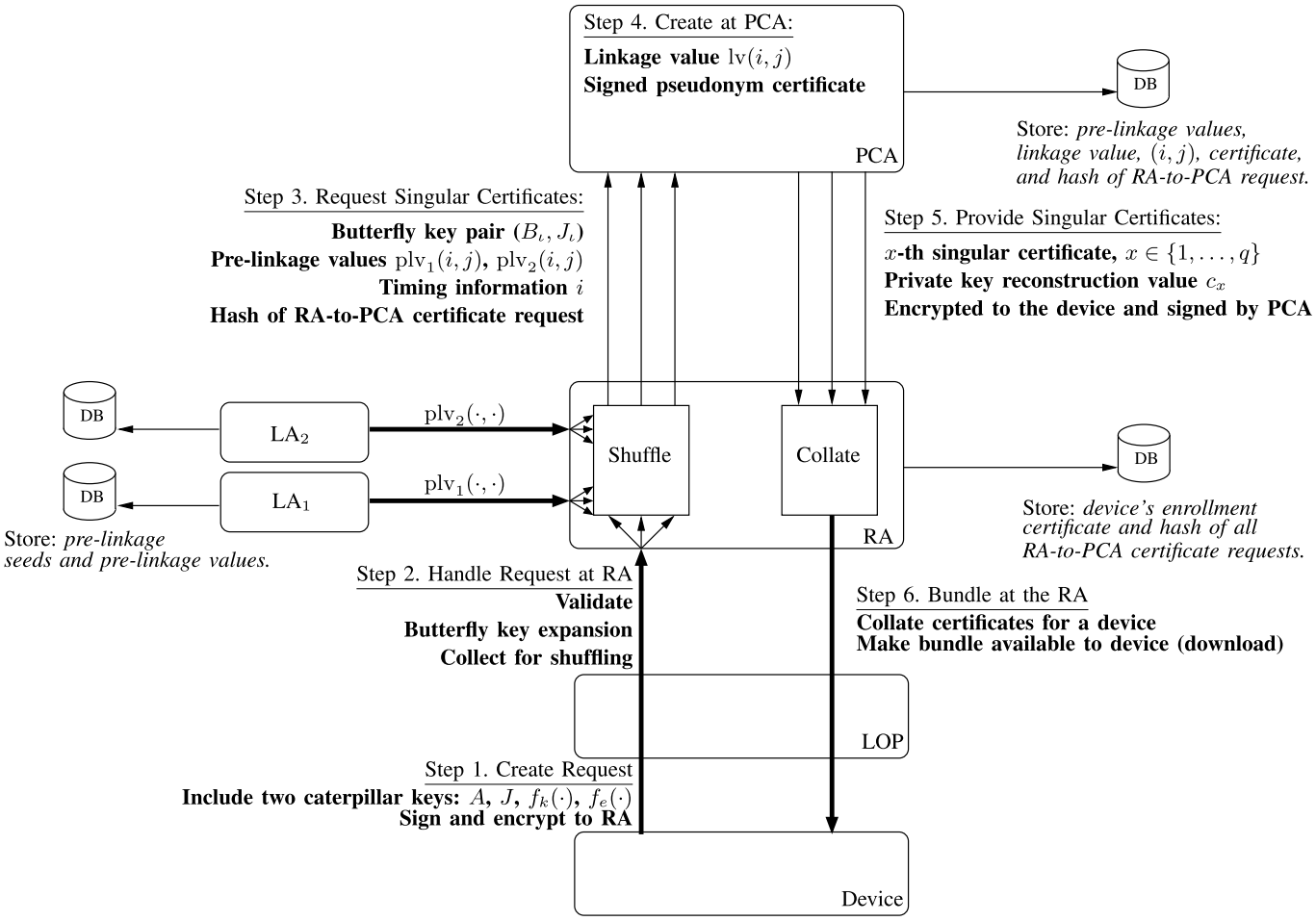

Fig. 5. Certificate provisioning.

identifiers (e.g., IP address) by replacing these identifiers with its own, such that the request appears to the RA as originating from the LOP. The functionality of the LOP is very similar to the masquerading feature implemented in many Internet routers.

- **Step 2.** The RA decrypts the request, validates the device's enrollment certificate to authenticate the devices and verifies that the device is not revoked. Further, it checks if this is the only request by the device. If all checks succeed, the RA sends an acknowledgment to the device and performs the butterfly key expansion as explained in Section IV. Otherwise, the RA rejects the request. The RA collects several such requests from different devices along with the sets of pre-linkage values received from the LAs. Once enough such requests are available, the RA shuffles the individual expanded requests.
  Note that during pre-generation of additional pseudonym certificates, the RA requests pre-linkage values from each of the LAs for a particular initial linkage seed that is associated with that device using the LCI to identify the corresponding linkage chain.
- **Step 3.** The RA sends requests for individual pseudonym certificates to the PCA, one certificate per request, where each request consists of a to-be-signed certificate, a response encryption public key, one encrypted pre-linkage value from each of the LAs $(\mathrm{plv}_1(i,j), \mathrm{plv}_2(i,j))$, and the hash of the RA-to-PCA pseudonym certificate request.
- **Step 4.** The PCA decrypts the pre-linkage values and computes the linkage value $\mathrm{lv}(i,j) = \mathrm{plv}_1(i,j) \oplus \mathrm{plv}_2(i,j)$. It then adds the linkage value to the to-be-signed certificate, and implicitly signs it to create a pseudonym certificate. It then creates a private key reconstruction value. Subsequently, it encrypts both the pseudonym certificate and the private key reconstruction value, using the response encryption public key.
- **Step 5.** The PCA signs the encrypted packet generated in step 4, and sends it to the RA. Signing the encrypted packet provides a guarantee to the device that the PCA encrypted the packet for the device. This prevents a man-in-the-middle attack where an insider at the RA substitutes the valid response encryption key with another key for which the RA knows the private key, and thus the RA may be able to see the contents of the pseudonym certificate including the linkage value.
- **Step 6.** The RA collects the encrypted packages for one week and bundles them for a given device - so called

TABLE II
INFORMATION STORED BY SCMS COMPONENTS.

Component	Stored Information
LA	Initial linkage seed, pre-linkage value
PCA	Encrypted pre-linkage values from both LAs and their corresponding (i, j) values, linkage value, certificate, and hash of RA-to-PCA pseudonym certificate request
RA	Enrollment certificate and its validity period, hash value of RA-to-PCA pseudonym certificate request

batches. The RA provides the batches to the device for download.

For revocation purposes, the SCMS components involved in the pseudonym certificate provisioning store different information corresponding to a given pseudonym certificate as listed in Table II.

VI. REMOVING MISBEHAVING DEVICES

The removal of misbehaving devices in an efficient manner is an essential design objective. We separate the removal of misbehaving devices into (1) reporting misbehavior, (2) globally detecting misbehavior, (3) investigate misbehavior, and (4) revoking a misbehaving device.

A. Misbehavior Reporting

V2V messages from misbehaving or defective devices can contain false or misleading information. We distinguish between intentional and unintentional misbehavior, where the latter includes all faults and error cases of devices. In both cases, it is crucial that benign participants neglect messages from misbehaving devices. One approach to accomplish this is to run misbehavior detection algorithms on the device (local misbehavior detection) to identify misbehaving nodes. Another approach is to report potentially misbehaving devices to the SCMS. The SCMS will run misbehavior detection algorithms and then inform all participants about certificates, which are no longer trustworthy. In the misbehavior reporting process, devices will send misbehavior reports to the MA via the RA. The RA will combine and shuffle the reports from multiple reporters to prevent the MA from tracking the reporter's path based on the reports. The format of a misbehavior report is not fully defined yet, but a report will include suspicious and alert-related BSMs, associated pseudonym certificates, a misbehavior type, as well as the reporter's pseudonym certificate and corresponding signature from the time the report was created. The reporter will encrypt the report to the MA. In the following, we will focus on the process of Global Misbehavior Detection and Revocation for the case of OBE pseudonym certificates. Subsequently, we describe the processes required for other types of certificates.

B. Global Misbehavior Detection

Global misbehavior detection (GMBD) is the process to identify potential misbehavior in the system, investigate suspicious activity, and if confirmed, to revoke certificates of misbehaving devices. The MA owns and executes the GMBD process. A CAMP research project has developed some GMBD algorithms. CAMP will integrate those into the current SCMS implementation. However, as the V2X landscape continues to evolve and new threats and forms of misbehavior are discovered, we expect that development of additional algorithms will continue over time. We understand the development of Misbehavior Detection methods and algorithm as an iterative tasks that will continue throughout the lifetime of the SCMS. One example of misbehavior, however primitive, would be a malicious actor who intentionally projects the position of the sending vehicle 3 meters to the left (or right for right-hand drive countries). These messages would cause alerts to the oncoming traffic, which oncoming vehicles would detect as potential misbehavior. A receiving vehicle would store these messages (assuming multiple) and put them into a misbehavior report, along with all defined data and details. It would encrypt the report to the MA and send it to RA for submission to the MA. As other vehicles also detect this misbehaving vehicle, they would also send misbehavior reports to the MA. As the number of reports grows, it would trigger the misbehavior detection algorithms and initiate the misbehavior investigation process potentially leading to the revocation of the malicious devices certificates.

It is worth noting that the sending vehicle could handle this type of misbehavior locally. We expect OEMs and device developers to tackle misbehavior at the device level from many angles to prevent malicious messages from being sent or used within safety applications. Misbehavior Detection requires that the MA can learn whether multiple misbehavior reports point to the same device. It also requires the MA to collect information that it publishes in a CRL to revoke a device's certificates. Additionally, the MA needs to provide the RA with the information required to perform blacklisting which blocks the misbehaving device from getting new certificates. The SCMS design requires the following components to collaborate to support misbehavior detection which introduces a form of checks and balances::

1) The MA, PCA and one of the LAs have to collaborate to reconstruct linkage information.
2) The MA, PCA, RA, and both the LAs have to collaborate to produce revocation information for the CRL.
3) The MA, PCA and the RA have to collaborate to determine the enrollment certificate of the misbehaving device, which the RA will add to its blacklist.

The MA executes step 1 as part of the Misbehavior Investigation to determine whether a device or a set of devices did indeed misbehave. After MA marked a device as misbehaving, MA executes steps 2 and 3 as part of Revocation to determine revocation information for the CRL and the enrollment certificate that RA adds to its internal blacklist.

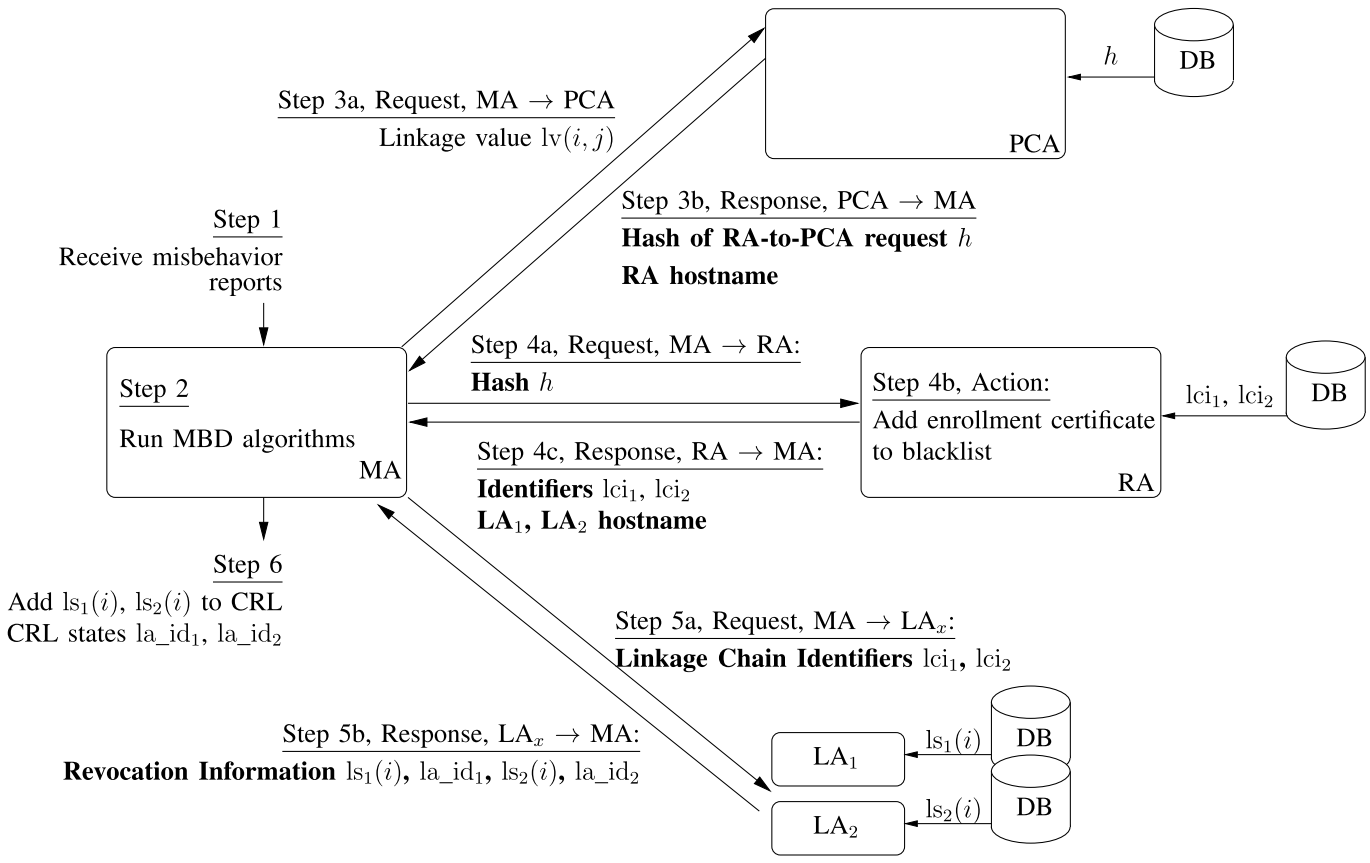

Fig. 6. Pseudonym certificate revocation.

C. Misbehavior Investigation

Misbehavior Investigation is the process to determine whether suspicious activities are indeed due to misbehavior, and to identify misbehaving devices. The Misbehavior Detection algorithm running in the MA initiates a process that depends on inputs from PCA and one LA. This separation introduces checks and balances into the system. We recommend a mechanism that limits the number of requests PCA and LA accept as well as the amount of information returned to MA to protect privacy to the highest level possible. Finally, we recommend that PCA and LA keep records of every request and that the SCMS Manager audits these log files regularly.

In the following, we present a detailed step-by-step description of this process. Note that Steps 1 and 2 are included for completeness and cover Misbehavior Reporting and Global Misbehavior Detection, respectively.

- **Step 1.** The MA receives misbehavior reports, including a reported pseudonym certificate with linkage value $\text{lv} = \text{plv}_1 \oplus \text{plv}_2$.
- **Step 2.** The MA runs global misbehavior detection algorithms to determine which reported pseudonym certificates might be of interest, i.e., for which pseudonym certificates it needs to retrieve linkage information.
- **Step 3.** The MA requests the PCA to map the linkage values lv of the identified pseudonym certificates to the corresponding encrypted pre-linkage values $(\text{plv}_1, \text{plv}_2)$ from the PCA's database. The PCA returns the encrypted pre-linkage values to MA.
- **Step 4.** The MA requests either LA_1 or LA_2 to find out whether a set of encrypted plv_1 (or, resp., plv_2) point to the same device. The LA will only respond if the number of encrypted plv pointing to the same device is above a defined threshold (e.g., five). There may be additional protective measures to reduce the amount of information returned to the MA.

D. Revocation and Blacklisting

The MA revokes and blacklists a device if it determines during Misbehavior Investigation that the device was indeed misbehaving. Next we present a detailed description of the Revocation and Blacklisting process, which identify the linkage seeds and the enrollment certificate corresponding to a pseudonym certificate. Figure 6 illustrates this process, with the first two steps summarizing the Misbehavior Investigation.

- **Step 3.** The MA requests the PCA to map the linkage value lv of the identified pseudonym certificate to the corresponding hash value of the RA-to-PCA pseudonym certificate request. The PCA returns this value and the hostname of the corresponding RA to the MA.
- **Step 4.** The MA sends the hash value of the RA-to-PCA pseudonym certificate request to the RA. The RA can map the hash value to the corresponding enrollment certificate and add it to its blacklist. The RA does not reveal the

enrollment certificate to the MA. The MA receives the following information from RA, which it then uses to gather information necessary for revocation:

  - The hostnames of the LAs involved in creating the pseudonym certificates linkage value,
  - An array of LCIs for each LA. The LA can use an LCI to look up the linkage chain and the underlying linkage seed. The RA returns multiple linkage chain identifiers only if a device owns certificates from multiple independent linkage chains, which we consider an exception.

- **Step 5.** The MA requests the LA_1 (resp., the LA_2) to map lci_1 (resp., lci_2) to the linkage seed $\text{ls}_1(i)$ (resp., $\text{ls}_2(i)$), where i is the currently valid time period. Both the LAs return their linkage seed to the MA. Further, each LA provides to MA its linkage authority ID (la_id_i). Note that given a linkage seed $\text{ls}_1(i)$ and the corresponding la_id_i, only the *forward* linkage seeds (i.e., $\text{ls}_1(j)$ for $j \geq i$) can be calculated, and thus *backward* privacy of the revoked device is maintained.
- **Step 6.** MA adds the linkage seeds $\text{ls}_1(i)$ and $\text{ls}_2(i)$, and the corresponding pair of LA IDs la_id_1, la_id_2 to the CRL. The CRL globally states the current time period i. For efficiency reasons, the CRL may group entries with the same LA ID pair together to save over-the-air bytes. Then the MA's CRLG signs the CRL and publishes it.

E. Global Misbehavior Detection & Revocation for Non-Pseudonym Certificates

The misbehavior detection and revocation process shown above is tailored to the case of pseudonym certificates. This type of certificate is the only type using linkage values which allows for efficient revocation of a multitude of certificates. All other types of certificates allow for a simpler process for misbehavior detection and revocation. Without linkage values, all revoked certificates need to be identified separately on the CRL in order to revoke them. The process reads as follows:

- **Step 1.** The MA receives misbehavior reports, including a reported certificate. This certificate does not contain linkage values.
- **Step 2.** The MA presents the certificate to the PCA and requests the hash of the individual certificate request from the PCA. The PCA delivers the hash of the individual certificate request, along with appropriate host information.
- **Step 3.** Using the hash of the individual certificate request, the MA instructs the RA to add the enrollment certificate to its blacklist. Furthermore, the MA asks whether there are any non-expired certificates for this device.
- **Step 4.** The RA replies with a list of hashes of individual certificate requests of non-expired certificates for the device to be revoked.
- **Step 5.** Using the additional hashes of the individual certificate requests, the MA can retrieve the other non-expired certificates from the PCA.
- **Step 6.** The MA adds CertIDs (a truncated hash of the certificate of, say, 8 bytes) of all non-expired certificates to the CRL.

Note that there are two possible optimizations of the process described above. First, the PCA could return only the CertIDs of the predecessor certificates and successor certificates in Step 5. Second, Step 5 can be avoided if the RA chooses to store the CertID of certificates not using linkage values (if certificates are not encrypted by PCA to the device and RA is able to read certificates provided to devices).

F. CRL Size

The size of the CRL grows linearly with the number of revoked entities. The assumption is that all OEMs will provide at least enough storage for 10,000 entries, which translates to a file size of approximately 400 KB. Therefore, a good CRL design will tag entries with information that allows devices to identify the 10,000 entries that are of highest priority to them: e.g., entries could be tagged with a location, or with the severity of misbehavior associated with that device, or with an indicator that the private keys have been made public. The final CRL design is still under development; a preliminary design is provided in IEEE Std 1609.2-2016, but it does not provide clear mechanisms for prioritization.

Note that currently there is no way to undo a revocation, and a revoked device can be reinstated only by repeating the process of bootstrapping, cf. Section V-A.

1) Revocation and Tracking: The value of having multiple linkage authorities is that it prevents an insider from gaining information that would enable him to track a vehicle, while at the same time allowing identification of a specific device under controlled circumstances. There are two circumstances where this is useful:

- Revocation: as described above, the LAs enable efficient revocation via CRLs.
- Misbehavior detection: If part of the misbehavior detection process is to check whether two messages, signed with different certificates, origin from the same device, then the LAs can be used to support this internal investigation in a privacy-preserving manner. It is still a subject of research to determine what information LAs should be allowed to provide to MA, and under which circumstances.

G. CRL Series: Identifying the Authorized Revoker for a Certificate

Within the system there may be multiple *CRL Sequences*, where a sequence is a temporally ordered set of CRLs addressing the same group of devices. The design "pins" each entity in the system to a specific CRL Sequence and allows different CRL sequences to be generated by different SCMS components.

There are a number of advantages to this:

- In the real world, jurisdiction over different vehicle or component types may rest in different authorities; it is therefore sensible to architect the system in such

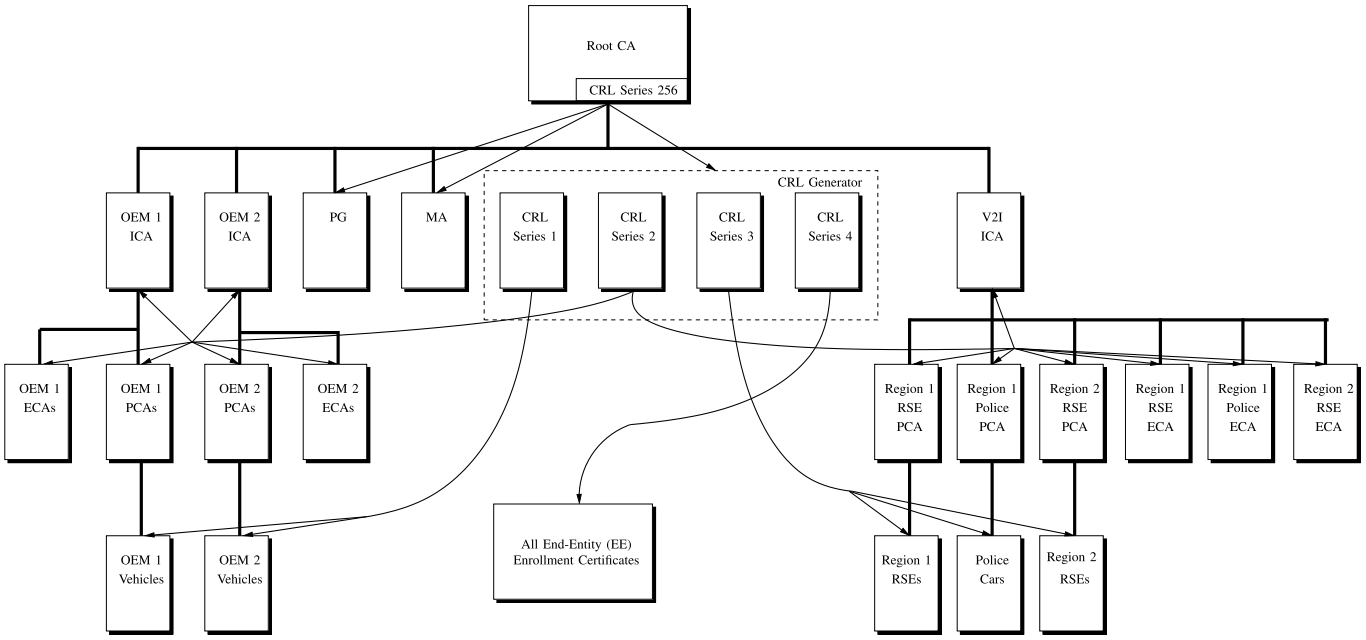

Fig. 7. CRL series.

a way that devices do not have to be managed by a single central authority. This allows the appropriate authority to take responsibility for managing particular applications or fleets, avoiding a situation where a central CRL issuer might be reluctant to take responsibility for issuing CRLs for particular devices due to, for example, concerns about liability.

- The natural revocation cycle length for different types of devices may be different. It may be appropriate to update the end-entity CRL daily, but the SCMS component CRL once a month. Alternatively, it might be desirable to publish CRLs continually as new revocation information becomes available – this may be the case for CRLs for BSM senders as there may be a significant volume of revoked devices. Finally, some fleets, devices, or applications might hardly ever experience revocation. In this case they might not have regularly issued CRLs, but may instead issue a new CRL only when necessary.
- Given that it makes sense to have different CRL Sequences, it is sensible to identify which CRL sequence is relevant to an entity:
  - It improves efficiency of a revocation check when a message is received from that entity, as only the most recent CRL in only the relevant sequence need be checked;
  - It removes the risk that an identifier collision would result in an entity being accidentally revoked.
  - It ensures that if one CRL generator is compromised, it can only falsely revoke devices that it was supposed to have jurisdiction over, and cannot falsely revoke other devices.

Note that although the ability to have different CRL signers is useful, if there are multiple CRL signers this potentially increases the communications burden on the vehicles as they now have to obtain multiple CRLs. This is addressed by having a central CRL store where all current CRLs are included in a single file which can be downloaded by vehicles as they obtain connectivity.

This is done in the IEEE 1609.2 certificate and CRL structure by use of the *Certificate Revocation Authority CA* (CRACA) ID and the CRL Series field. The CRACA is a CA in the chain of the potentially-revoked entity that either issues the CRL itself, or directly issues the CRL Generator certificate. The CRL Series appears in both the entity's certificate and the CRL and allows to distinguish between different CRL Sequences from the same CRL Generator. This provides unambiguous identification of the relevant CRL Sequence, similarly to (but more compactly than) the functionality provided by the CRLDistributionPoints in X.509 certificates. Figure 7 shows the CRL series structure for the SCMS Proof of Concept. There is one main CRL generator. It manages four CRL Series, covering vehicle pseudonym certificates (series 1); all SCMS components except those identified below (series 2); vehicle identification and RSE application (series 3); and enrollment certificates (series 4). Additionally, the Root CA manages a CRL series, series 256, which can be used to revoke the Policy Generator, CRL Generator, and MA certificates. Note that for all of these CRL sequences, the Root CA acts as the CRACA, as it is the CA on the chain of the to-be-revoked devices; the CRL generator acts simply as an agent of the root CA and is kept separate so that the root CA can be kept offline to the greatest extent possible.

VII. CRL Processing and Distribution

To revoke all pseudonym certificates of a given device from a time period i forward, both seeds $\mathrm{ls}_1(i)$ and $\mathrm{ls}_2(i)$, the linkage IDs $\mathrm{la\_id}_1$, $\mathrm{la\_id}_2$, and timing information i and $j_{\max}$ are published on the CRL and distributed. A device

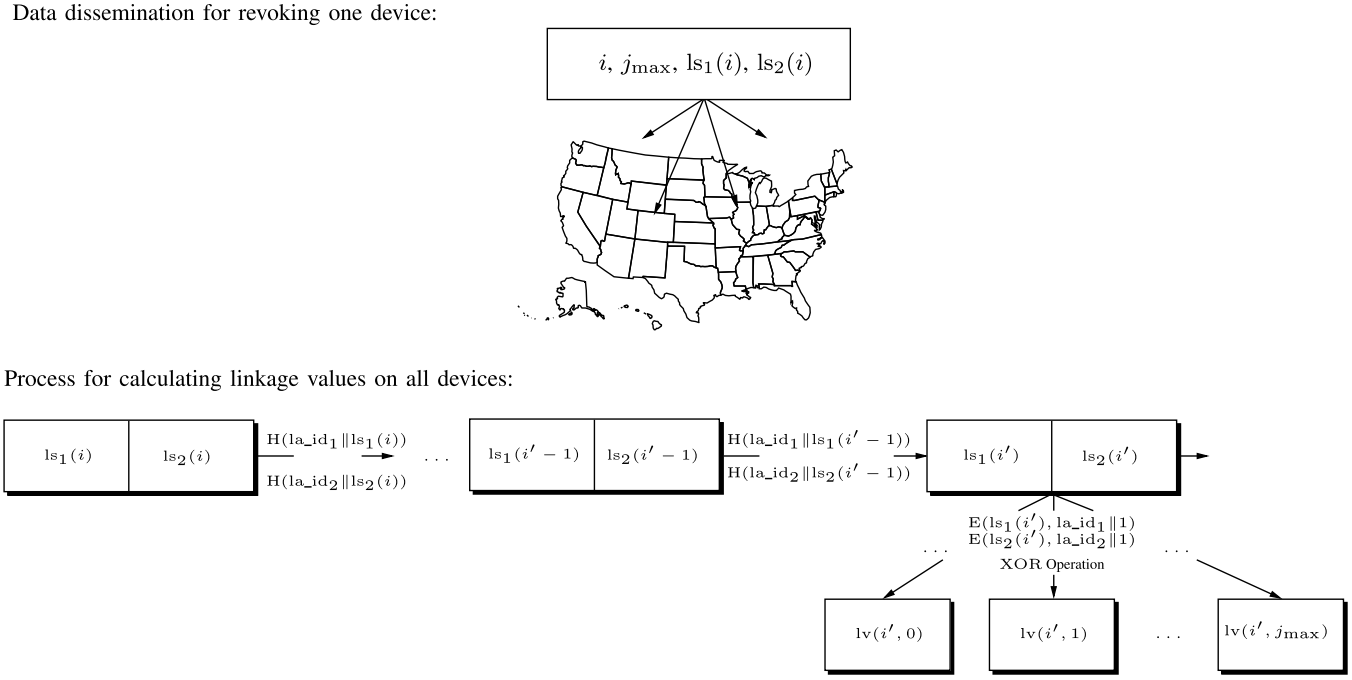

Fig. 8. Revocation of linkage values.

that received the CRL forward hashes both seed values individually, calculates the pre-linkage values of the current time-period and then XORs the pre-linkage values to obtain the current linkage value. Figure 8 shows the information which needs to be published for the revocation of a given device at time instant i, as well as the process for finding all possible linkage values for this device and $i' > i$. Note that this mechanism protects backward-privacy since certificates of revoked devices cannot be identified for time-periods before the linkage seeds were published.

The current baseline approach for CRL distribution would be distribution to each device via multiple channels, e.g. RSEs, cellular, satellite communications, or customer WiFi. One possible innovative method would be the use of a collaborative distribution model as defined in a preliminary way by [21]. In collaborative distribution, some devices are initially seeded with CRLs (by RSEs, by cellular data, or by some other means) and then distribute the CRLs to their peer devices as they drive past them in the normal course of events. Devices that have received the CRL become distributors in their turn, allowing efficient coverage of the entire system with a small number of initial seeders. Simulation results are promising, e.g., [21] shows that the area of Zurich can be provided with a CRL using a single RSE within a few hours. However, interesting open research questions include how best to use relatively limited channel capacity in the 5.9 GHz band and whether there are any security challenges/concerns that require mitigation.

VIII. Re-Enrollment

Re-enrollment of a device might be necessary due to several reasons. We define re-enrollment as either of the following:

- **Reinstatement:** A device is reinstated if the original enrollment certificate is reinstated by removing it from the RA's blacklist.
- **Re-bootstrapping:** A device is re-bootstrapped if the device is wiped and then bootstrap is executed to issue a new enrollment certificate. This is similar to a factory reset and requires a secure environment.
- **Re-issuance:** A device is reissued if the public key of the enrollment certificate is reused to issue a new enrollment certificate. The device keeps all pseudonym certificates and uses the same butterfly key parameters.
- **Re-establishment:** A device is re-established if the device's integrity can be verified remotely and the device then requests a new enrollment certificate using the old enrollment certificate to authenticate the request.

Note that we strongly suggest to only use re-bootstrapping and re-establishment but no reinstating or re-issuing.

Device re-enrollment is useful in the following scenarios:

- **Change of cryptography:** Advances in cryptanalysis might make it necessary to replace the underlying cryptographic algorithms. In the next decades, this will likely be the case to introduce post-quantum cryptography algorithms. In this case, devices need to receive updated firmware, ideally over-the-air, and then request new enrollment certificates that use the updated cryptographic scheme.
- **Device revocation via CRL:** Re-bootstrapping is the only option if the MA revoked a device and listed it on the CRL.
- **Enrollment certificate roll-over:** It is good practice and a security requirement in the SCMS to limit the life-span of enrollment certificates, which motivates the need for a rollover over-the-air to a new enrollment

certificate, which is equivalent to re-establishing a device. A device can request a new enrollment certificate if the MA has not revoked the current enrollment certificate. The device creates a new private/public key pair and includes that public key in its certificate rollover request to RA. The device digitally signs the rollover request with its current enrollment certificate. The RA verifies the request, forwards it to ECA, and the ECA, in turn, signs the requested enrollment certificate containing the new public key.

- **Device revocation due to a revoked ECA:** If an ECA has been revoked, such that a device now holds an invalid enrollment certificate, re-enrollment is necessary as well. As a standard approach, the device should be re-bootstrapped. A re-establishment of devices that hold an enrollment certificate from a revoked ECA creates the risk to issue a new enrollment certificate to a malicious device. Hence we strongly recommend re-bootstrapping in a secure environment of all affected devices.
- **Root CA and ICA revocation:** If a Root CA certificate is revoked, it is assumed that a new Root CA certificate is established by means of electors (see Section IX) and all relevant components have been equipped with a new certificate under the new Root CA certificate. ECAs need to be re-certified, and the SCMS Manager has to give permission to re-establish devices that hold an enrollment certificate issued by a re-certified ECA if there is evidence that there was no ECA compromise. Otherwise devices need to be re-bootstrapped.

  In case of either a Root CA certificate or ICA certificate revocation the revocation information will be distributed via an updated CRL. Devices will pick up the updated CRL, and check, if the certificate is in the trust chain to their enrollment, pseudonym, application, or identification certificate. In case of the enrollment certificate a device will send a re-enrollment request to the RA, using their current enrollment certificate. RAs receive information from the SCMS Manager, which ECAs are re-certified and will allow devices with an enrollment certificate signed by re-certified ECAs to request a new enrollment certificate. The device needs to request new pseudonym, application, or identification certificates with the new enrollment certificate afterwards.

  In case only the device's pseudonym, application, or identification are affected by the revoked Root CA certificate, the device needs to send new provisioning requests with the current enrollment certificate to the RA. RA will stop all pending pre-generation jobs with the previous PCA certificate and delete all pre-generated certificates as soon as it receives the updated CRL.

  The case for a revoked ICA is equivalent, except that it's not necessary to create a new Root CA certificate and introduce it to the system via Electors.

IX. Elector-Based Root Management

Given that devices can re-enroll as described in subsection VIII after a root CA certificate's validity period ends or a revocation was necessary and a new root CA certificate has been established for replacement, how can a device start trusting this new root CA certificate? The trust in an initial root CA certificate is implicit, as it is installed in a secure environment with out-of-band communication during bootstrapping of the device. One option would be to get the device back to that secure environment and use out-of-band communication to install the new root CA certificate. However, this is suboptimal due to the required effort and will render the overall V2X system partly out-of-order until all devices have installed the new certificate.

To manage the root CA certificate over time and gain resilience against compromises on any level, the SCMS needs the ability to heal itself, which means to bring itself into a state where it can endure another singleton compromise or end of the validity period of a Root CA. This recovery should occur while keeping the devices operational whenever possible, that is, capable of sending, receiving and validating BSMs, and be able to restore the system hierarchy without requiring physical access to devices. Elector-based Root Management is the solution that provides those means by installing a distributed management schema on top of the SCMS Root CAs.

A. Distributed Management & Electors

A distributed management scheme, like a democracy, contains within itself the power to replace an established hierarchy and does not succumb to a single failure. The concept of *Electors*, which together have the power to change and manage the trust relationships of the system, adds such a scheme to the SCMS design. Within a system like the SCMS, the number of electors should be $2n + 1$, where n is the number of simultaneous elector expiration/compromises that the SCMS can tolerate.

Like in a democracy the Elector-based Root Management introduces a *Ballot* with *Endorsements*. The electors cast *Votes* by signing an endorsement of a given root CA or elector certificate. A ballot aggregates all these endorsements. When a quorum of valid elector endorsements is on the ballot, any component in the system can trust the ballot.

The electors are not part of the PKI hierarchy, and therefore they can use a different crypto-system than the SCMS PKI. In fact, each of them can use a different one. This raises the probability that in case of a root CA or elector certificate compromise due to a broken cryptography, the system is still able to heal itself.

The resulting system may have multiple, self-signed root CA certificates, each of which operates at the top of their trust chain. Each root CA's certificate is endorsed by a ballot with at least a quorum of votes from non-revoked electors. Devices need to verify the trust chain up to a root CA certificate, at which point they must verify that a quorum of non-revoked electors has endorsed that root CA certificate.

B. Ballots & Endorsements

Electors operate by signing endorsements. A ballot can include the following basic types of endorsements:

- Add root CA certificate
- Add elector certificate

TABLE III
ROOT MANAGEMENT MESSAGE IMPACT ON DEVICES.

Action	Impact on device operations
Revocation of an Elector	As long as there are at least three electors with a quorum of two, then one elector may be removed without impacting operation: The remaining electors are still a quorum and their endorsements of the root CA certificate would still be valid. A single revoked elector would not stop operations of any device. A replacement elector may then be added back to the system to return to a state with three valid electors. A larger number of electors may be used to improve the system's resilience to compromise or failure of these top-level trust anchors.
Revocation of a Root CA	Revoking a root CA certificate would stop operations of devices that possess certificates chaining up to the revoked root CA certificate. Those devices would need to re-enroll and be re-provisioned with a different root CA before they could be trusted by other devices.
Addition of an Elector	A new self-signed elector certificate that is endorsed by a quorum of valid electors can be trusted by devices and other SCMS components without the need of returning them to a secure environment. In addition, this new elector can endorse existing root CA certificates without the need for any updates of the existing valid certificates, including the device's pseudonym certificates.
Addition of a Root CA	A new, self-signed root CA certificate that is endorsed by a quorum of valid electors can be trusted by devices and other SCMS components without the need of returning them to a secure environment. Devices can immediately begin to trust messages that chain up to the new root CA.

- Revoke root CA certificate
- Revoke elector certificate

Each ballot contains only one type of endorsement. SCMS components, including devices, receive ballots adding a certificate via a certificate chain file distributed by the PG. They receive ballots removing a certificate via the CRL distributed by the CRL store. All components know the quorum and the certificates of the initial set of electors and therefore can validate the endorsements contained in the ballot. Once the ballot is validated, the component can follow the endorsed action to add or remove the ballots certificate from its trust store.

The SCMS Manager will coordinate the production of the ballot messages.

C. Structure of Ballots

The ballot which aggregates all independent elector endorsements is an ASN.1 structure. This structure contains the following elements:

1) The certificate of the root CA or elector to be endorsed
2) A sequence of endorsements, each containing:
   a) The type of endorsement
   b) The hash id of the certificate to be endorsed
   c) The generation time of the endorsement
   d) A signature of the elector.

Note that the validity period of a ballot is implicitly given by the validity period of the endorsed certificate.

D. Revocation/Endorsement Impact on devices

A key consideration in the design of the root management system is to maintain secure operation of devices without requiring recall or manual re-enrollment of individual devices (as described in section VIII). Table III outlines the status of devices through the addition or revocation of electors and root CAs.

X. ORGANIZATIONAL SEPARATION

Within the SCMS design, different SCMS components represent different logical functions. Dedicated organizations have to provide some of these logical functions to provide an acceptable level of privacy for V2V safety communication applications using BSMs and to prevent a single organization from being able to determine which pseudonym certificates belong to a device. This capability would allow an attacker to track a vehicle by combining this information with captured BSMs.

This section identifies which SCMS components must be organizationally separate. The general rule is that two components cannot be run by the same organization if the combined information held by the components would allow

an insider to determine which pseudonym certificates belong to a device. This leads to the following specific requirements for organizational separation:

- **PCA and RA:** If one organization would run these two components, the organization would know which pseudonym certificates had been issued to which device. This is because the RA knows the requests to which certificates correspond, and the PCA knows the corresponding pseudonym certificates.
- **PCA and one of the LAs:** If one organization would run the PCA and either (or, both) of the LAs, it could link all pseudonym certificates (from any batch) issued to any device since LA knows a set of pre-linkage values that go into the certificate set, and PCA sees these pre-linkage values at certificate generation time.
- **LA_1 and LA_2:** If one organization would run both the LAs, it would know all the pre-linkage values and XOR them opportunistically to obtain the linkage values, which appear in plaintext in pseudonym certificates. This would allow identification of which pseudonym certificates belong to the same device.
- **LOP and (RA or MA):** The LOP hides the device's location from the RA and the MA, respectively, and no single organization should jointly run these components.
- **MA and (RA, LA, or PCA):** No single organization should run a combination of the MA and any of the RA, the LA or the PCA. If combined, the MA could circumvent restrictions during misbehavior investigation and learn more information than necessary for misbehavior investigation and revocation purposes.

When other certificate types then pseudonym certificates are generated, no specific organizational separation is required.

XI. Conclusions and Outlook

We introduced a Security Credential Management System (SCMS) for V2X communications, with a special emphasis on V2V safety application communication. This SCMS design is a leading candidate for the V2X security backend design in the US. One of the remaining challenges is to define policies that balance among security, privacy, and efficiency that will support the establishment of a nationwide system. The proposed solution uses five certificate types to cover all identified V2X application categories. This article focused on V2V safety communication security and the life-cycle of OBE pseudonym certificates since this certificate type requires most consideration in terms of privacy protection and complexity. The SCMS is designed to scale with the number of devices and to protect privacy of end-users against inside and outside attackers by separation of duties.

Next steps towards an SCMS deployment are as follows:

- Define an effective CRL dissemination based on the concept of Collaborative Distribution [21].
- Large scale test deployments of a proof-of-concept SCMS.
- Development of misbehavior detection algorithms, implementation and test of misbehavior reporting, investigation and detection in a proof-of-concept implementation.

Acknowledgments

The presented results are a culmination of efforts by many parties and people. This includes members of the US Department of Transportation (USDOT), the Crash Avoidance Metrics Partners Vehicle Safety Consortium (CAMP VSC3, CAMP VSC5), and the Vehicle Infrastructure Integration Consortium (VIIC). The OEMs Daimler, Ford, GM, Honda, Hyundai-Kia, Nissan, Toyota, and VW-Audi participate in CAMP VSC3 and the VIIC. BMW and Chrysler participate in the VIIC. The authors are deeply grateful to Bill Anderson, Stephen Farrell, the late Scott Vanstone, and members of the USDOT and CAMP VSC3 for reviewing earlier versions of the manuscript and providing important suggestions for improvements.

References

[1] Crash Avoidance Metrics Partners LLC. *Glossary*. Accessed: Feb. 24, 2018. [Online]. Available: https://wiki.campllc.org/display/SCP/Glossary

[2] U.S. Department of Transportation. *How Connected Vehicles Work*. Accessed: Feb. 24, 2018. [Online]. Available: http://www.its.dot.gov/factsheets/pdf/JPO_HowCVWork_v3.pdf

[3] U.S. Department of Transportation-National Highway Traffic Safety Administration. *U.S. DOT Federal Motor Vehicle Safety Standards; V2V Communications; NPRM*. Accessed: Feb. 24, 2018. [Online]. Available: https://www.federalregister.gov/documents/2017/01/12/2016-31059/federal-motor-vehicle-safety-standards-v2v-communications

[4] N. Bißmeyer, H. Stübing, E. Schoch, S. Götz, J. P. Stolz, and B. Lonc, "A generic public key infrastructure for securing Car-to-X communication," in *Proc. 18th World Congr. Intell. Transp. Syst.*, 2011, p. 12.

[5] *Intelligent Transport Systems (ITS); Security; Threat, Vulnerability and Risk Analysis (TVRA)*, document ETSI TR 102 893V1.1.1 (2010-03), 2010.

[6] *Intelligent Transport Systems (ITS); Security; Security Services and Architecture*, document ETSI TS 102 731V1.1.1 (2010-09), 2010.

[7] *Intelligent Transportation Systems (ITS); Security; Stage 3 Mapping for IEEE 1609.2*, document ETSI TS 102 867 v1.1.1, Jun. 2012.

[8] Secure Vehicle Communication. *Security Architecture and Mechanisms for V2V/V2I*. Accessed: Feb. 24, 2018. [Online]. Available: https://sevecom.eu/Deliverables/Sevecom_Deliverable_D2.1_v3.0.pdf

[9] F. Ahmed-Zaid *et al.*, "Vehicle safety communications consortium—Applications (VSC-A)," Crash Avoidance Metrics Partners Veh. Safety Commun. Consortium, Tech. Rep. DOT HS 811 492A, 2006. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/811492a.pdf

[10] *Annex E.4.1: Why Sign Data Instead of Using a Message Authentication Code?* IEEE Standard 1609.2, 2013.

[11] EMVCo, LLC. *Worldwide EMV Deployment Statistics*. Accessed: Feb. 24, 2018. [Online]. Available: https://www.emvco.com/about_emvco.aspx?id=202

[12] Wikipedia. (2016). *Public Key Infrastructure—Wikipedia, the Free Encyclopedia*. Accessed: Feb. 28, 2017. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Public_key_infrastructure&oldid=755924137

[13] U.S. Department of Transportation. *Safety Pilot Model Deployment*. Accessed: Feb. 24, 2018. [Online]. Available: http://safetypilot.umtri.umich.edu

[14] Crash Avoidance Metrics Partners LLC. *SCMS Interface Specification*. Accessed: Feb. 24, 2018. [Online]. Available: https://stash.campllc.org/projects/SCMS/repos/scms-asn/browse

[15] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn, "A security credential management system for V2V communications," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Dec. 2013, pp. 1–8.

[16] J. Petit, F. Schaub, M. Feiri, and F. Kargl, "Pseudonym schemes in vehicular networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 228–255, 1st Quart., 2015.

[17] *On-Board System Requirements for V2V Safety Communications*, document J2945/1_201603, SAE, 2016.

[18] *Intelligent Transport Systems (ITS); Security; Pre-Standardisation Study on Pseudonym Change Management*, document ETSI ETSI TR 103 415, Dec. 2018.

[19] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymity—A proposal for terminology," in *Designing Privacy Enhancing Technologies* (Lecture Notes in Computer Science), vol. 2009, H. Federrath, Ed. Berlin, Germany: Springer, 2001, pp. 1–9.
[20] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proc. MobiCom*, 2008, pp. 116–127.
[21] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, "Efficient certificate revocation list organization and distribution," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 3, pp. 595–604, Mar. 2011.
[22] T. Dierks and E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.2*, document RFC 5246, 2008.
[23] J. R. Douceur, "The Sybil attack," in *Proc. IPTPS*, 2002, pp. 251–260.
[24] H. Krishnan and A. Weimerskirch, "'Verify-on-demand'—A practical and scalable approach for broadcast authentication in vehicle-to-vehicle communication," in *Proc. SAE World Congr.*, 2011, pp. 536–546. [Online]. Available: https://www.sae.org/publications/technical-papers/content/2011-01-0584/
[25] *Standard Specifications for Public-Key Cryptography*, IEEE Standard 1363-2000, 2000.
[26] *Recommended Elliptic Curves for Federal Government Use*, Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Jul. 1999.
[27] *Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV)*, SEC Standard 4, 2011.

Benedikt Brecht started his career in the automotive industry at Volkswagen in 2010. He was involved in projects of Volkswagen AG Group Research before moving to Product Development. Since 2013, he has been with Group IT as a Program Manager with the Volkswagens Connected Car Program. He was assigned to work in the U.S. as a Senior IT Connected Vehicle Engineer with the Department of Safety Affairs and Advanced Research on Vehicle-to-X (V2X) topics in 2015. He is currently a Principal Investigator of the U.S. Department of Transportation-funded Crash Avoidance Metrics Partners Program involved in security topics for V2X communication. Before that, he was a Research Associate with the University of Potsdam and the Freie Universitaet Berlin.

He received the bachelor's and Diploma degrees in IT systems engineering and computer science from the Hasso Plattner Institute and the University of Potsdam, Germany.

Dean Therriault is an IT Architect with General Motors (GM) with a focus on networks, infrastructure, and security. Before joining GM, he was with a number of automotive OEMs in the Detroit area designing enterprise backoffice directory and PKI solutions. After four years with IBM, he joined GM as an IT architect and a Systems Engineer and began working on an enterprise PKI strategy. Shortly thereafter, he transferred to the Global Vehicle Safety Team and got involved in the Connected Vehicle Program. Currently, he is assigned to the Crash Avoidance Metrics Partners Program as a Principal Investigator involved in a number of Security Credential Management System-related projects.

André Weimerskirch established the Transportation Cyber Security Group at the University of Michigan Transportation Research Institute and co-founded the embedded systems security company ESCRYPT which was sold to Bosch in 2012. He is a VP Cyber Security at Lear Corporation. He is active in all areas of automotive and transportation cyber security and privacy. He is a Main Designer of the vehicle-to-vehicle security system, which will likely be the largest security system ever deployed. He has published numerous articles in automotive and embedded cyber security. He is the Co-Founder of the American workshop on embedded security in cars (escar USA). He is the Vice Chair of the SAE Vehicle Electrical System Security Committee, co-chairs the Mcity Cybersecurity Working Group, University of Michigan, co-organizes the SAE ComVEC Heavy Vehicle Cybersecurity Session, and is a member of the joint SAE/ISO Cybersecurity Working Group.

William Whyte received the B.A. degree from the Trinity College and the Ph.D. degree in statistical mechanics of neural networks from Oxford University. He is responsible for the strategy and research behind the OnBoard Security's activities in vehicular communications, security, and cryptographic research. Before joining OnBoard Security, he was the Chief Technology Officer of NTRU Cryptosystems. He previously served as a Senior Cryptographer with Baltimore Technologies, Dublin, Ireland. He is the Chair of the IEEE 1363 Working Group for new standards in public key cryptography and has served as a technical editor of two published IEEE standards, the IEEE Std 1363.1-2008 and the IEEE Std 1609.2-2006, as well as the ASC X9 standard X9.98.

Virendra Kumar received the B.Tech. degree in electrical engineering from the IIT Varanasi (BHU), Varanasi, India, and the Ph.D. degree in computer science from the Georgia Institute of Technology, Atlanta, GA, USA. During his B.Tech. and Ph.D. studies, he had the opportunity to visit and work at several leading research labs, including Microsoft Research, Alcatel-Lucent Bell Labs, Symantec Research Labs, and Gemplus (now, Gemalto) Research and Development. He is a Principal Scientist at OnBoard Security Inc., where his research and consulting services are focused on security and privacy in connected vehicles. Before joining OnBoard Security, he was a Cryptographer at Security Innovation Inc., and before that a Security Consultant at ESCRYPT Inc. He has published research papers at top tier conferences and journals, and he is an inventor of multiple patents issued by the U.S. Patent and Trademark Office. His primary research interests include the design and analysis of cryptographic protocols with an emphasis on efficiency and practicality.

Thorsten Hehn received the Ph.D. (Dr.-Ing.) degree in digital communications from the University of Erlangen–Nuremberg, Germany, in 2009. He joined Volkswagen AG in 2009 and was with the Volkswagen Group of America, USA, from 2012 to 2014. His work within Volkswagen was always focused on either security or communications technologies. Most recently, his main attention is toward 3GPP-based 5G topics and sidelink communications. He has published numerous journal papers, and he is an inventor of multiple patents in the field of connected car.

Roy Goudy received the bachelor's degree in metallurgical engineering from the University of Washington, the master's degree in physics from the University of Utah, and the master's degrees in automotive systems engineering and mechanical engineering from Lawrence Technological University. He is a Senior Principal Engineer with the Nissan Technical Center North America, Farmington Hills, MI, USA, where he is responsible for the research and development of cooperative safety and mobility applications.