CARCLOUD: A Secure Architecture for Vehicular Cloud Computing

Marvy B. Mansour
British University in Egypt,
Cairo, Egypt
marvy.badr@bue.edu.eg

Cherif Salama[1], *Hoda K. Mohamed*[2]
Computer and Systems Engineering Department,
Ain shams University,
Cairo, Egypt
cherif.salama[1], hoda.korashy[2]@eng.asu.edu.eg

Sherif A. Hammad
Avelabs,
Cairo, Egypt
sherif.hammad@avelabs.com

***Abstract*—Vehicular Cloud Computing (VCC) is an emerging technology that vehicle drivers use for different applications. VCC applications include Location-Based Service (LBS) applications, which require from the vehicles to send frequent location updates to LBS Providers for real-time services. Unfortunately, this may lead to tracking and identification of drivers, and so breaching their privacy. In this paper, we propose a secure architecture for Vehicular Cloud Computing, called CARCLOUD, in order to solve the previous problems. Our proposed system consists of four main phases: Vehicle Bootstrapping Phase, Vehicle and LBS Provider Certificate Provisioning Phase, Vehicle and LBS Provider Certificate Revocation Phase, and finally the LBS Request in Vehicular Cloud Computing Phase. In our system, we use a novel idea that allows a Road-Side Unit (RSU) to form cluster containing all vehicles within its coverage range, and to act as the Cluster Head of cluster formed. Also, we introduce to use RSU Clouds for LBS applications to guarantee service delivery for vehicles. In addition, our system includes a novel Reward System to reward or penalize vehicles while using LBS applications. Finally, we present an analysis of security and privacy of proposed system, and show that our system provides protection against both internal and external system attacks.**

Keywords—Certificate Provisioning in VANET, Certificate Revocation in VANET, LBS Request in Vehicular Cloud Computing, Location Privacy, Security and Privacy Techniques in Vehicular Cloud Computing, Vehicle Tracking.

I. INTRODUCTION

In Vehicular Ad-hoc Networks (VANETs), vehicles communicate with each other through Vehicle-to-Vehicle (V2V) communications. Also, in VANETs, vehicles exchange data with Road-Side Unit (RSU) / Infrastructure through Vehicle-to-Infrastructure (V2I) communications. V2V and V2I communications use Dedicated Short Range Communications (DSRC). However, due to emerging needs of different types of Internet applications, vehicles need to access Cloud to use these applications. So, Vehicular Cloud Computing (VCC) became an important interesting topic for researchers from different industries. In VCC, RSU acts as a Gateway for the VANET to access Public Cloud. In our system, we allow RSU to communicate with the Zone Controller that acts as a Gateway for Cloud access. We assume that each Zone Controller has RSU (1-10) in its domain, where each RSU has vehicles (1-b) in its coverage range. Moreover, a Public Cloud is split into distinct regions (1-p), e.g. city, town, district, etc. Also, Private Clouds, operated by private organizations, can exchange information with Public Cloud through firewalls. We assume that there are Private $Cloud_{1\text{-}q}$, where $q > p$. Also, each Public Cloud can communicate with other Public Clouds. However, Cloud-to-Cloud communication is not our scope.

For vehicles in VANET, each have their own unique Public / Private (Pu / Pr) key pairs, also use Public Key Infrastructure (PKI). All vehicles have their digital certificates, which include their public keys. PKI manages and facilitates digital certificates, which are used to sign messages to build trust among participants. To provide sender authentication and thwart an attacker from inserting false messages, the sender vehicle digitally signs its messages (which includes sender position, speed, etc.). While the receiver vehicle verifies signature before acting on received message. Key Hashed Message Authentication Code (HMAC) is used to verify the authentication and data integrity of message. Moreover, in VANET, unlinkability of sender vehicle to message it sends, known as sender anonymity, is a main demand. Also, another vital need is to achieve unlinkability of consecutive messages sent from the same vehicle, known as tracking. It is worth to note that our system entities exhibit multiple trust levels, such that a Location-Based Service (LBS) Provider is a semi-trusted entity, whereas a vehicle is an untrusted entity. While other entities, such as RSUs, are trusted, but might be overtaken by an adversary and turn to be untrusted entity(s).

Contributions of this paper are as follows. Our proposed system is divided into four main phases: First, Phase I of our system is Vehicle Bootstrapping Phase, which includes two main processes. The Initialization Process shows the certificates needed by a vehicle to trust messages received. While, the Enrollment Process shows the certificates needed by a vehicle to send messages. Second, Phase II of our system is Vehicle and LBS Provider Certificate Provisioning Phase, which is divided into two main processes. Part A is Vehicle Certificate Provisioning Process, which shows the steps of a vehicle requesting and then receiving a pseudonym certificate. Then, Part B is LBS Provider Certificate Provisioning Process, which shows the steps of an LBS Provider requesting and then receiving a pseudonym certificate. Third, Phase III of our system is Vehicle and LBS Provider Certificate Revocation Phase, which is divided into two main processes. Part A is Vehicle Certificate Revocation Process, which shows the steps of revoking the certificate of a malicious vehicle. Then, Part B is LBS Provider Certificate Revocation Process, which shows the steps of revoking the certificate of a malicious LBS Provider. Finally, Phase IV of our system is LBS Request in Vehicular Cloud Computing Phase, which shows both the steps of a vehicle sending an LBS Request to an LBS Provider, and the steps of an LBS Provider replying to request sent by offering the needed service. In this phase, we introduce to use RSU Clouds for LBS applications to guarantee service delivery for vehicles. Also, in this phase, we show how our system uses a novel Reward System to reward or penalize vehicles while

using LBS applications. Moreover, in our system, we use a novel idea that allows a RSU to form cluster containing all vehicles within its coverage range, and to act as Cluster Head (CH) of cluster formed. Our system uses Index values for secure efficient Certificate Provisioning and Revocation, and LBS Request.

The rest of this paper is organized as follows. In Section II, we discuss the previous security and privacy techniques in VANET and VCC. Then, in Section III, we describe the roles of our system entities. While in Section IV, we demonstrate the four main phases of our proposed system. Whereas in Section V, we analyze the security and privacy of proposed system. Finally, we conclude the paper in Section VI.

II. Background and Related Work

VANET security has attracted many works as [1-4]. While other works have worked in in-vehicle security as [5]. Also, some others have focused in security and privacy of VCC [6].

For example, in [7], the Intrusion Detection System (IDS) proposed for VCC is used to detect only malicious vehicles and didn't take into account a malicious Service Provider (SP). Also, in [8], the proposed framework protects Vehicular Cloud against malicious vehicles and clients, but not against malicious SPs. While in [9], a secure incentive-based architecture for VANET was presented. However, this architecture didn't consider an untrusted SP case, and also didn't penalize malicious vehicles. Whereas in [10], they proposed a service-oriented security framework for VCC, which didn't include any certificate revocation method in order to revoke certificates of malicious vehicles or SPs. Moreover, in [11], a secure protocol for privacy preserving in Cloud-based vehicular Delay Tolerant Networks (DTNs) was presented. Despite that, the protocol didn't protect vehicles from untrusted SP. In addition, in [12], the authors proposed VANET-based Clouds framework that includes a secure and privacy-aware service. This framework included a certificate revocation method for vehicles only and not for an SP, so it cannot maintain privacy of a vehicle against an untrusted SP. Furthermore, in [13], they designed and implemented an IDS framework for VANET to protect against attacks from malicious vehicles, but it didn't protect from malicious SPs.

Moreover, according to our knowledge, none of the previous works as [3, 14] have proposed an integrated architecture for VCC that includes Certificate Provisioning and Revocation for both a Vehicle and an LBS Provider, as well as LBS Request in VCC. Thus, previous works didn't protect against untrusted SPs, and so they are not suitable to provide a secure architecture for VCC. Our system provides a solution to the previous problems.

RSU Cluster Formation

In VANET, vehicles detect their neighbors by periodically broadcasting a "hello" message. Vehicles are grouped into clusters, known as Vehicular Clouds (VCs). Previous works in [3, 7, 11, 13, 15] have used dynamic clustering techniques, where a vehicle acts as the CH. The main drawback of the previous dynamic clustering techniques is that structure of cluster continuously changes (unstable) for the following reasons: 1) high mobility of vehicles, 2) CH moves outside the cluster, and 3) when two CHs co-inside (that is when their coverage range overlap).

Thus, in our system, in order to avoid the previous problems of forming dynamically changing cluster and CH: We allow a RSU to form a cluster that contains all vehicles within its coverage range, and to be the CH of cluster formed. The public key of RSU, the CH, is broadcasted to all nodes and is used to authenticate messages from RSU. Messages sent between RSU and other nodes in its cluster, are encrypted and authenticated. When a vehicle enters range of a new RSU, RSU will send its public key to vehicle for V2I communication. Also, in turn the vehicle will send its public key to RSU. If any vehicle leaves or joins cluster without notifying RSU, the CH, it will be considered as malicious vehicle. In this case, RSU will inform Misbehavior Authority (MA), mentioned in Section III, to revoke certificate of this vehicle, as explained in Section IV.

III. Description of Roles of Entities in Proposed System

In this section, a detailed description of the roles of each entity in our proposed system is given, where there is an organizational separation between the entities.

1- Cloud Manager: Issues and relays policy decisions to Cloud entities and vehicles. Sends and receives reports from Cloud entities to ensure their fair operation. Provides authenticated information about trust updates or configuration changes of a Cloud entity to other entities, e.g. an entity that changes its certificate or address. Sets measures for evaluating misbehavior reports based on procedures set.

2- Vehicle$_{1\text{-}b}$: When a certain group of vehicles (1-b), which lies within the same proximity from each other, communicate with each other via V2V communication, they create a Vehicular Cloud (VC). Vehicles in a VC send Basic Safety Messages (BSMs) to each other needed for safety applications through their On-Board Units (OBUs).

3- Roadside Unit + Address Mask1 (RSU + AM1), and AM2: RSU includes AM1 unit. *For RSU:* Acts as a gateway for Cloud access, and a CH for a VC. For more details, refer to Sections I and II. *For AM1:* Hides the locations of sender by changing its source addresses. This is similar to the function of Location Server (LS) in the Geographic Location/Privacy (Geopriv) standard of IETF Geopriv Working Group [16], where the LS acts as a proxy for users (location generators). LS reduces the resolution of location information received, and changes it to another format based on privacy policies specified in location information sent by user. Also, AM1 shuffles the Misbehavior Reports. *For AM2:* Entity present in Cloud that has similar function as AM1, but it is used for LBS Providers.

4- Zone Controller (ZC): For details, refer to Section I.

5- Registration Authority + Index Generation (RA + IG): RA includes IG unit. *For RA:* Registers a vehicle or an LBS Provider by storing their data, sent to it from them and other entities. Validates and processes requests sent to it from a vehicle or an LBS Provider, by checking that an entity is not included in its Internal Blacklist. Also, RA checks that an entity requests one set of certificates during a specific time period. A vehicle could only request certificate from its regional RA, which is the RA of zone where the vehicle exists, e.g. if vehicle exists in Zone$_1$, then RA$_1$ is its regional RA. Individual RAs may be limited to a specific geographic region.

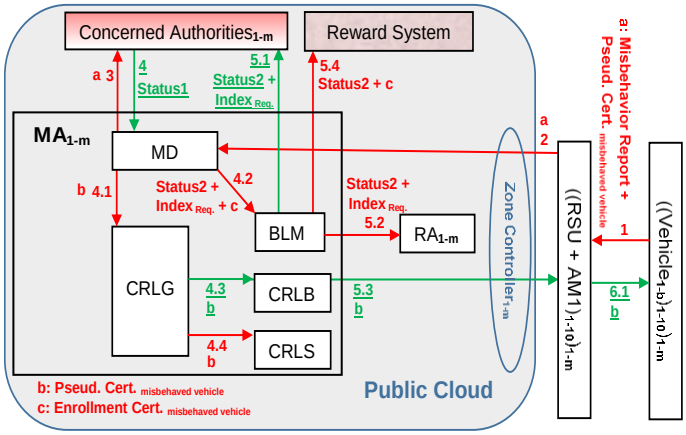

Figure 1. Functions of MA in Vehicle Certificate Revocation Process

A vehicle could not request certificate from multiple RAs. Note, each LBS Provider should register with same RA, where the vehicle that requested a service is registered. *For IG:* Generates and assigns unique independent Index value for each entity request. Stores this data in a table. Shuffles all requests received together.

6-Enrollment Certificate Authority (ECA): Responsible for issuing Enrollment Certificates (long-term) to vehicles and LBS Providers, which are used for Pseudonym Certificates (short-term) requests.

7- Pseudonym CA + Misbehavior Authority (PCA + MA): PCA includes MA unit. *For PCA:* Issues short-term Pseudonym Certificates to both; vehicles to request services, LBS Providers to provide services. *For MA:* Contains the following five subunits illustrated in Figure 1. Note, parameter *m* in Figure 1 is explained later in Section IV. Also, functions of MA in LBS Provider Certificate Revocation Process are similar to that in Vehicle (explained in Section IV), except that Application Manager needs to be added in Figure 1. **(a) Global Misbehavior Detection (MD):** i) Processes Misbehavior Reports received from vehicles, and LBS Provider Incident Reports received from Application Manager; to identify which vehicles or LBS Providers are misbehaving. Receives Misbehavior Reports from RSU, the CH, as mentioned before in Section I. ii) Checks that both Reporter and reported entity don't exist in its Blacklist. iii) Uses collected information from Concerned Authorities, e.g. police stations, to identify malicious entities and also check the trustfulness of Reporter. iv) Verifies that the Reporter's location is near to incident location. v) Computes both the number of true and false reported incidents for every Reporter and send it to Concerned Authorities, which helps them to know the level of trustfulness of a specific Reporter. vi) Computes the number of reported incidents occurred for a certain entity and send it to Concerned Authorities. **(b) CRL Generator (CRLG):** Revokes and adds pseudonym certificate of misbehaved vehicle to Certificate Revocation List (CRL) for Vehicles, and misbehaved LBS Provider to CRL for LBS Providers. Signs two CRLs. Note, CRL entries need to be ordered by priority, and CRL size needs to be upper-bounded. **(c) Blacklist Manager (BLM):** sends information to other entities needed for updating their Internal Blacklists. **(d) CRL Broadcast (CRLB):** Sends to vehicles the current CRL for Vehicles via corresponding RSU, to reject future messages from misbehaved vehicles. **(e) CRL Store (CRLS):** Stores two CRLs.

8- Concerned Authorities: Communicate with Cloud Manager. Include entities, such as: ECA, Intelligent Transportation System (ITS) Management, Transportation Authorities, and Governmental Agencies.

9- Root CA and Intermediate CA: components of PKI [14].

10- LBS Provider: Replies to requests sent from Application Server by offering needed service. Charges a vehicle according to type and duration of application used, after considering points in vehicle's balance (sent to it from Reward System).

11- Application Server: Contains LBS Providers (1-k) in its domain. i) Sends service requests to available trusted LBS Providers. ii) Sends regular reports (Application Reports) to Application Manager. iii) Sends to Application Manager list of available LBS Providers in its domain, to get the updated list of trusted ones. iv) If LBS Provider$_x$ deny to offer services for certain time period, Application Server$_y$ will send a report to Application Manager (LBS Provider Incident Report), to decrease its trust level. (Note, Application Manager will notify Reward System to decrease trust level of LBS Provider$_x$.)

12- Application Manager: Stores details of each service provided by an LBS Provider to a vehicle, which are included in Application Reports received from different Application Servers. Sends LBS Provider Incident Reports to MA$_y$, to decide whether to revoke LBS Provider$_x$' s certificate (as in LBS Provider Certificate Revocation Process in Section IV).

13- Reward System: When a vehicle requests an LBS application, either case could happen. If vehicle is eligible to receive application, Reward System will reward to it by adding points to its account balance. The amount of points added is according to type of application used and duration of usage. These points can be used later by vehicle when requesting another application as explained in Phase IV of Section IV. Else, if vehicle is not eligible, Reward System will penalize it by setting its balance to zero. Note, Reward System is also used for LBS Provider's trust level, as explained here in no.11.

Note, some system entities contain two Internal Blacklists: one for vehicles and one for LBS Providers, while other entities have only one for vehicles.

IV. Description of Phases of Proposed System

In this section, a detailed description of each phase of our proposed system is given.

After performing Vehicle Bootstrapping Phase, a vehicle may request an LBS application, where Vehicle Certificate Provisioning Process occurs for a vehicle to acquire a pseudonym certificate. Then, LBS Request Phase takes place, where a vehicle will use its (acquired) pseudonym certificate to send an application request. Consequently, LBS Provider of requested application will perform LBS Provider Certificate Provisioning Process to acquire a pseudonym certificate. After that, LBS Provider will use its pseudonym certificate to offer vehicle the needed service. This is the case if for every service request there is a new certificate generated for a vehicle or an LBS Provider. However, this could not be the case if the Certificate Provisioning Process for a vehicle or an LBS Provider takes place every three days. In this case, only the LBS Request Phase will run with every service request. Note, Certificate Revocation Process for Vehicle/LBS Provider occurs anytime during execution of other processes.

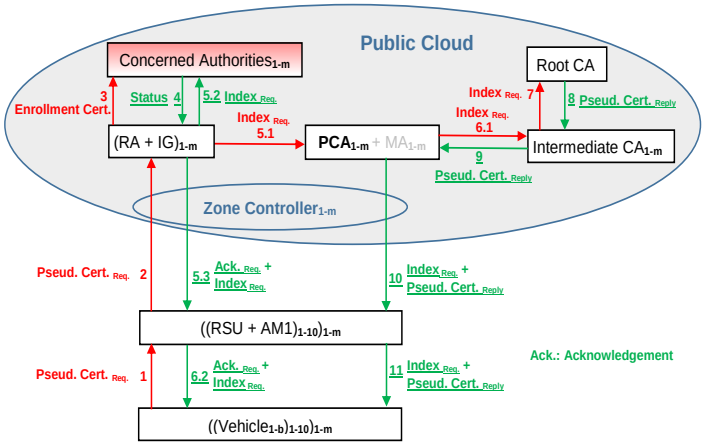

Figure 2. Vehicle Certificate Provisioning Process

Our system processes are performed for each Public Cloud. We assume that each Public Cloud contains the following entities: a Cloud Manager, an Application Manager, a Reward System, a Root CA, *m* Intermediate CA, *m* ECA, *m* PCA + MA, *m* RA + IG, *m* Zone Controller, *m* Concerned Authorities, *m* Application Server, and *k* (for each 1-*m*) LBS Providers. Each of these entities, except Zone Controller, has its own Pu/Pr key pair which is known to other entities. If a system entity includes a unit, a unit will have same Pu/Pr key pair as its entity, and share same information with its entity.

When a vehicle enters range of a RSU (that may exist in domain of a new zone), RSU will broadcast to it public keys needed to access Cloud in that zone, such as: RSU's Pu key, and PCA/MA's Pu key. RSU will discover that a new vehicle has entered its range from the "hello" message sent by vehicle.

Our proposed system consists of four main phases: Vehicle Bootstrapping Phase, Certificate Provisioning for Vehicle and LBS Provider Phase, Certificate Revocation for Vehicle and LBS Provider Phase, and finally LBS Request in VCC Phase. Note, figures below contain abstracted messages, their details are mentioned in text. Also, "red arrows" in figures are for "sending a message", while "green arrows" are for "receiving a reply" for message sent. Note, every message is sent with its timestamp, which is the exact time that it has been sent.

Phase I: Vehicle Bootstrapping Phase

In this phase, a secure channel is established with Cloud Manager via Device Configuration Manager (DCM). After that, in a secure environment, DCM communicates with vehicle for bootstrapping. This phase includes two main processes: Initialization Process, and Enrollment Process. In Initialization Process, a vehicle obtains certificates needed to trust messages it received. Whereas, in Enrollment Process, a vehicle obtains certificates needed to send messages and actively participate in network. During Enrollment Process, trustworthy information is received by ECA about vehicle to be enrolled, e.g. Electronic License Plate (ELP), from vehicle or DCM. Then, enrollment certificate is issued to vehicle by ECA, signed by ECA's private key, which is needed by vehicle to obtain pseudonym certificates that are used to request services. Different ECAs issue enrollment certificates for different geographic regions. Also, in this phase, a vehicle receives the following certificates: (1) DCM certificate, (2) ECA certificate, and (3) PCA/MA certificate.

Phase II: Vehicle and LBS Provider Certificate Provisioning Phase

Phase II of our system is divided into two main processes: Part A, which is Vehicle Certificate Provisioning Process. Part B, which is LBS Provider Certificate Provisioning Process. These processes are designed to maintain entity's privacy against insider attacks.

A) Vehicle Certificate Provisioning Process

Below, the details of each step in Vehicle Certificate Provisioning Process are given as shown in Figure 2.

Step 1: Vehicle sends to **RSU+AM1** in its range a pseudonym certificate request (Pseud. Cert.$_{\mathbf{Req.}}$) signed by its enrollment certificate (encrypted with vehicle's private key); along with hash of request and its enrollment certificate, and encrypts whole message with RSU public key. Pseud. Cert.$_{\mathbf{Req.}}$ contains details about vehicle as: location, speed and address. Vehicle's public key acts as Vehicle Identifier in network (VID).

Step 2: RSU decrypts the received message and verifies the vehicle's signature and enrollment certificate. Then, AM1 changes vehicle's address after checking its validity. If all checks are valid, then **RSU** will sign the message and encrypt it with RA's public key, and send it to **RA + IG** via corresponding Zone Controller.

Step 3: RA decrypts received message, and checks validity of RSU's signature and vehicle's enrollment certificate. Also, RA checks if vehicle is in its Internal Blacklist, and if it has sent more than one request for a specific time period. If all checks are valid, then **RA** will send a signed message containing vehicle's enrollment certificate to **Concerned Authorities**, to check that the vehicle is eligible to receive a certificate. Note, Concerned Authorities are notified of misbehaved vehicles from MA during Vehicle Certificate Revocation Process explained later.

Step 4: Concerned Authorities checks validity of RA's signature and vehicle's enrollment certificate. If all checks are valid, then **Concerned Authorities** will reply back to **RA** with the vehicle's status along with its enrollment certificate, and encrypt message with RA's public key.

Step 5: RA decrypts received message. If all checks are valid, and if vehicle is not eligible, then RA will send a deny message to it via corresponding Zone Controller and RSU, and will discard future requests sent from it as RA acts as a request filter for PCA. Else, if vehicle is eligible, then IG will generate and uniquely assign an Index value for each request. Also, IG will shuffle its request with other requests received, in order to prevent PCA from linking together requests sent from the same vehicle. After that, **RA** will:

5.1 Send Index value for each vehicle's request, along with vehicle's status and enrollment certificate, and hash of request (e.g. using HMAC-SHA1) to corresponding **PCA**.

5.2 Send Index value for each vehicle's request, along with vehicle's status and enrollment certificate to **Concerned Authorities**, that is needed in Vehicle Certificate Revocation Process. Note, for each vehicle, Concerned Authorities store

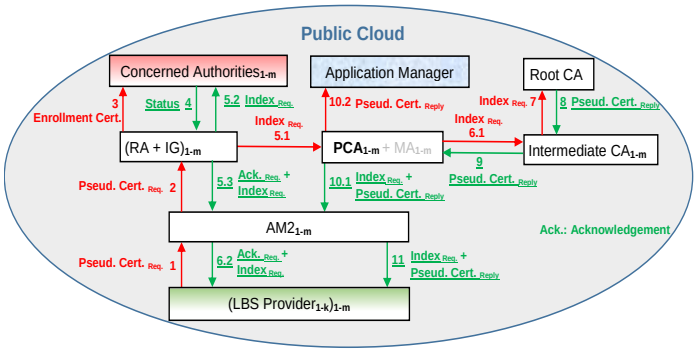

Figure 3. LBS Provider Certificate Provisioning Process

status and Index value against enrollment certificate.

5.3 Send Index value (encrypted with vehicle's public key included in enrollment certificate, so that RSU would not see it) along with an acknowledgement of reception of request and hash of request; sign message and encrypt it with RSU's public key, to **RSU** via corresponding Zone Controller.

Step 6:

6.1 PCA sends Index value and vehicle's status; signs message and encrypts it with Intermediate CA's public key, to corresponding **Intermediate CA**.

6.2 RSU decrypts received message and checks validity of RA's signature. If all checks are valid, then **RSU** will sign the message and broadcast it to **all vehicles in its range**. Note, when the target vehicle sees its hash value that has generated in Step 1, it decrypts the message and stores Index value sent. While other vehicles in RSU range will discard the received message as it doesn't contain their hash value. So, using hash value in this way saves the time of other vehicles, and thus enhances their performance.

Step 7: Intermediate CA decrypts received message and checks validity of PCA's signature. If all checks are valid, then **Intermediate CA** will sign the message and encrypt it with Root CA's public key, and send it to **Root CA**.

Step 8: Root CA decrypts message and checks validity of Intermediate CA's signature. If all checks are valid and if vehicle is eligible to receive a certificate, then Root CA will generate a new pseudonym certificate that contains a public key for vehicle and certificate's validity time period. After that, **Root CA** will sign the new certificate and generate its hash, include Index in message, then encrypt message with Intermediate CA's public key and reply to **Intermediate CA**.

Step 9: Intermediate CA decrypts message and checks validity of Root CA's signature. If all checks are valid, then **Intermediate CA** will sign the new certificate and generate its hash; include Index in message, then encrypt whole message with PCA's public key and reply back to **PCA**.

Step 10: PCA decrypts message and checks validity of Intermediate CA's signature. If all checks are valid, then PCA will sign the new certificate and generate its hash; include Index in reply, and encrypt whole reply with vehicle's public key so that RSU would not see it. After that, **PCA** includes the hash of request and the encrypted reply in a message, signs the message and encrypts it with RSU's public key, then reply back to **RSU** via corresponding Zone Controller. Note, PCA doesn't reply back or send to RA the new certificate, to mitigate vehicle tracking by the RA through correlation of vehicle's data stored at RA with new certificate.

Step 11: Similar to Step 6.2 here. RSU decrypts message and checks validity of PCA's signature. If all checks are valid, then **RSU** will sign the encrypted reply received and include the hash of request in message, then broadcast message to **all vehicles in its range.** Note, since the reply sent is encrypted with vehicle's public key that has sent the request, only that vehicle is able to decrypt the received reply and use the new certificate, and so avoids masquerade attack. Note, if target vehicle left RSU_x before receiving its reply, then RSU_x will send reply to RSU_y where vehicle may exist, using RSU-to-RSU communication in RSU Cloud explained in Phase IV.

B) LBS Provider Certificate Provisioning Process

Below, the details of each step in LBS Provider Certificate Provisioning Process are given as shown in Figure 3.

Step 1: Similar to Step 1 in Vehicle Provider Certificate Provisioning Process explained earlier, except that AM2 is used instead of RSU+AM1, where AM2 has its own public / private key pair. Also, Pseud. $Cert._{Req.}$ contains details about LBS Provider, e.g. service type(s) offered, organization / company's location, address, name, and type. LBS Provider's public key acts as LBS Provider Identifier in network (PID).

Step 2: Similar to Step 2 in Vehicle Certificate Provisioning Process, except that AM2 is used instead of RSU and no Zone Controller exist.

Step 3: Similar to Step 3 in Vehicle Certificate Provisioning Process, except that if all checks are valid, then **RA** will send LBS Provider's enrollment certificate; along with Pseud. $Cert._{Req.}$ and its hash to **Concerned Authorities**.

Step 4: Similar to Step 4 in Vehicle Certificate Provisioning.

Step 5: Similar to Step 5 in Vehicle Certificate Provisioning, except: AM2 is used instead of RSU, no Zone Controller exist.

Step 6: Similar to Step 6 in Vehicle Certificate Provisioning Process, except that in Step 6.2, AM2 is used instead of RSU. Also, if all checks are valid, then **AM2** will sign the received message and send it to **LBS Provider**.

Steps 7, 8 and 9: Similar to Steps 7, 8 and 9 in Vehicle Certificate Provisioning Process.

Step 10:

10. 1 Similar to Step 10 in Vehicle Certificate Provisioning Process, except that AM2 is used instead of RSU, and no Zone Controller exist.

10. 2 If all checks are valid, then **PCA** will encrypt the signed certificate and its hash with Application Manager's public key, and send it to **Application Manager**.

Step 11: AM2 decrypts received message and checks validity of PCA's signature. If all checks are valid, then **AM2** will sign the encrypted reply received and hash of request, and send them to **LBS Provider**.

Phase III: Vehicle and LBS Provider Certificate Revocation Phase

Misbehavior Detection and Reporting Scheme are used in Revocation Process to identify which vehicles / LBS Providers certificates to revoke. Phase III of our system is divided into two main processes: Part A, Vehicle Certificate Revocation

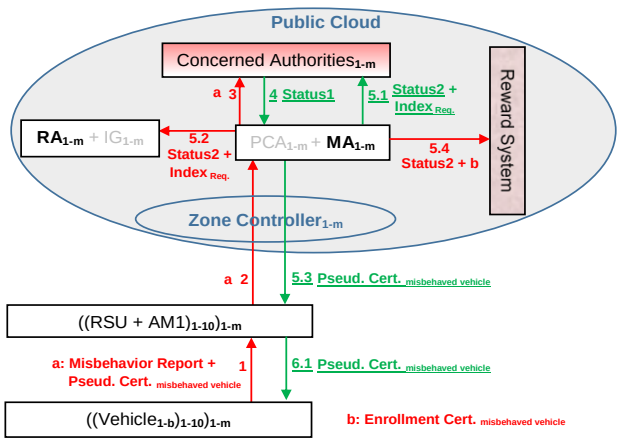

Figure 4. Vehicle Certificate Revocation Process

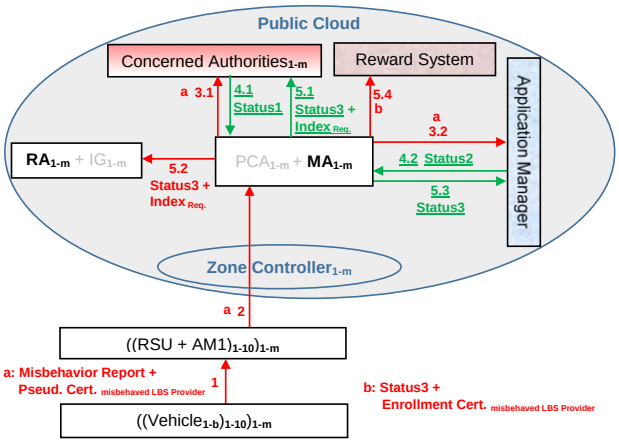

Figure 5. LBS Provider Certificate Revocation Process

Process; Part B, LBS Provider Certificate Revocation Process.

A) Vehicle Certificate Revocation Process

Below, the details of each step in Vehicle Certificate Revocation Process are given as shown in Figure 4. Note that, both Vehicle Certificate Provisioning (green arrows) and Revocation Processes (red arrows) are illustrated in Figure 6.

Step 1: Vehicle (Reporter) generates a Misbehavior Report that contains false beacon (VANET message) received, incident time, and Reporter's location; speed and address. Then, **Reporter** signs the Report and sends it to **RSU+AM1** in its coverage range, along with its hash, Pseud. Cert. of both Reporter and reported (misbehaved) vehicle. The whole message is encrypted with RSU's public key. Note, a vehicle has its own MD algorithms that work on local level.

Step 2: Similar to Step 2 in Vehicle Certificate Provisioning Process explained earlier. If all checks are valid, then AM1 will shuffle all messages received from multiple vehicles, to prohibit MA from reconstructing vehicle's route via messages sent from it. Also, RSU, the CH, will generate its own Reports as mentioned before in Section I. After that, **RSU** will sign each message, encrypt it with MA's public key, and send it to **MA** via corresponding Zone Controller.

Step 3: Similar to Step 3 in Vehicle Certificate Provisioning. If all checks are valid, then **MA** will sign the received message and send it to **Concerned Authorities**, to check trust status of Pseud. Cert. of both Reporter and reported vehicle(s). Also, MA will do the following operations: (1) MA performs some internal operations to detect trustfulness of received report(s), and determine which reported vehicle(s) its pseudonym certificate needs to be revoked. (2) MA checks if more than one Misbehavior Report points to the same vehicle or not. (3) MA checks how many times the same Reporter has reported the same vehicle, and if the time frame between every two consecutive Reports is within acceptable limits. For more details about MA internal operations, refer to Section III.

Step 4: Similar to Step 4 in Vehicle Certificate Provisioning.

Step 5: MA decrypts received message. If all checks are valid, and if the vehicle founded to be malicious, MA will generate Status2. Then, MA will add revoked Pseud. Cert. of this vehicle to its Internal Blacklist and to CRL for Vehicles. Also, CRLG will sign CRL, and CRLB will send it to RSU to broadcast it (every given time period). After that, **MA** will:

5. 1 and **5.2** Send a signed message to corresponding **Concerned Authorities,** and **RA** (encrypted with RA's public key); that includes: Status2, revoked Pseud. Cert., and Index value corresponding to Pseud. Cert.$_{\text{Req.}}$ of vehicle, to revoke Pseud. Cert. and resovle identity of vehicle, and add it to their Internal Blacklists to reject future requests sent from it.

5. 3 Send Status2 and revoked Pseud. Cert., sign message and encrypt it with RSU's public key, to **RSU** via corresponding Zone Controller; to avoid other vehicles waiting until new CRL is broadcasted and so enhancing their performance.

5. 4 Send a signed message to **Reward System** (encrypted with Reward System's public key); that includes: Status2, vehicle's Pseud. Cert. and enrollment certificate, so that Reward System penalize vehicle by setting its balance to zero.

Step 6: RSU decrypts received message and checks validity of MA's signature. If all checks are valid, **RSU** will sign received message, then it will:

6. 1 Broadcast message to **all vehicles in its range**.

6. 2 Send message to **neighboring RSUs**, using RSU-to-RSU communication in RSU Cloud explained later in Phase IV.

Step 7: Each of **neighboring RSUs** will broadcast received message to **all vehicles in their range**, to discard future messages sent from the same vehicle.

B) LBS Provider Certificate Revocation Process

Below, the details of each step in LBS Provider Certificate Revocation Process are given as shown in Figure 5.

Step 1: After a vehicle receives notification of service reply, as explained later in Phase IV, LBS Provider may stop at any time from providing needed service to it. So, vehicle would report (misbehaved) LBS Provider similar to Step 1 in Vehicle Certificate Revocation Process explained earlier, except that Misbehavior Report would contain: denied service type, incident time, and Reporter's location; speed and address.

Steps 2: Similar to Step 2 in Vehicle Certificate Revocation.

Steps 3:

3. 1 Similar to Step 3 in Vehicle Certificate Revocation.

3. 2 If all checks are valid, **MA** will sign received message, encrypt it with Application Manager's public key, send it to **Application Manager** to check reported LBS Provider status.

Steps 4:

4. 1 Similar to Step 4 in Vehicle Certificate Provisioning.

4. 2 Similar to Step 3 in Vehicle Certificate Provisioning. If all

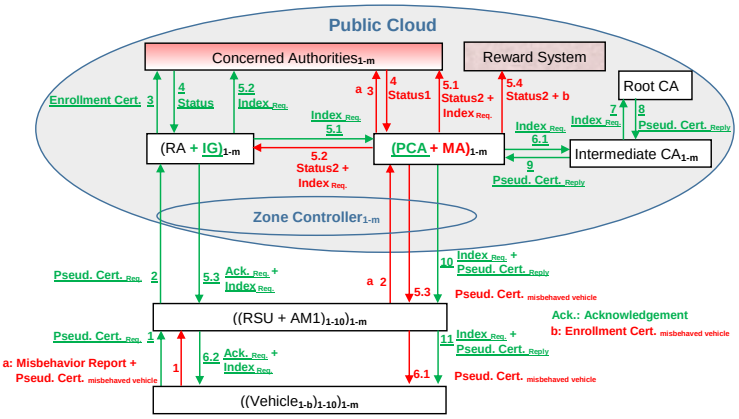

Figure 6. Vehicle Certificate Provisioning and Revocation Processes

checks are valid, then Application Manager will check trustfulness of Misbehavior Report(s) received, e.g.: (1) If Reporter requested that service type or not. If yes, did Reporter received requested service or not. (2) Which LBS Provider, if any, denied to provide the requested service to Reporter, using LBS Provider Incident Report sent to Application Manager from Application Server, as explained in Phase IV. (3) Status (Status3) of reported LBS Provider, e.g. if it denied to provide a certain service for a certain time period. (4) How many times and duration of each time, if any, reported LBS Provider provided a service to Reporter, using Index value corresponding to a certain service request; used in Phase IV. Note, if Application Manager discovered that Reporter sent false Misbehavior Report; i.e. if it found Index value in its list for another LBS Provider, it will report this to MA to revoke Reporter's certificate as in Vehicle Certificate Revocation. Also, note, Application Manager sends to corresponding MA LBS Provider Incident Reports; to decide whether to revoke LBS Provider's certificate, regardless whether MA has sent to it Misbehavior Report or not. After **Application Manager** performs previous operations, it will reply back to **MA** with Status3 along with received message; sign whole message and encrypt it with MA's public key.

Step 5: Similar to Step 5 in Vehicle Certificate Revocation.

5. 1 , **5.2** Similar to 5.1, 5.2 in Vehicle Certificate Revocation.

5. 3 MA will send a signed message to **Application Manager** (encrypted with Application Manager's public key); that includes: Status2, Status3, and revoked Pseud. Cert., so that Application Manager adds that vehicle to its Internal Blacklist.

5. 4 Similar to Step 5.4 in Vehicle Certificate Revocation, except that, Reward System will penalize LBS Provider by setting its trust level to zero.

Phase IV: LBS Request in Vehicular Cloud Computing Phase

Below, the details of each step in LBS Request in Vehicular Cloud Computing Phase are given as shown in Figure 7.

Step 1: Vehicle sends to **RSU + AM1** in its range a service request ($Service_{Req.}$) signed by its Pseud. Cert.; along with hash of $Service_{Req.}$; and its Pseud. Cert., and encrypts whole message with RSU's public key. $Service_{Req.}$ contains: service type and estimated duration, vehicle's location, speed, address.

Steps 2: Similar to Step 2 in Vehicle Certificate Provisioning Process explained earlier.

Step 3: Similar to Step 3 in Vehicle Certificate Provisioning. If all checks are valid, then **RA** will send to **MA** vehicle's Pseud. Cert. and hash of $Service_{Req.}$, sign message and encrypt it with MA's public key, to check Pseud. Cert. status.

Step 4: Similar to Step 3 in Vehicle Certificate Provisioning. If all checks are valid, then **MA** will:

4. 1 Reply back to **RA** with signed $Status_{Pseud.\ Cert.\ vehicle}$ along with received message, and encrypt whole message with RA's public key.

4. 2 Send to **Reward System** signed message; that includes: vehicle's enrollment certificate and Pseud. Cert.; hash of $Service_{Req.}$; along with signed $Status_{Pseud.\ Cert.\ vehicle}$, and encrypt whole message with Reward System's public key.

Step 5: Similar to Step 5 in Vehicle Certificate Provisioning. Note, if vehicle is not eligible to receive requested service, RA will discard future requests from it as RA acts as a request filter for Application Server. Also, note, if vehicle is eligible, IG will shuffle its request with other requests received, to prevent Application Server from reconstructing vehicle's path based on requests sent from it. After that, **RA** will:

5.1 and **5.2** Send a signed message to corresponding **Application Server** (encrypted with Application Server's public key) and **Reward System** (encrypted with Reward System's public key); that includes: Index corresponding to $Service_{Req.}$, $Service_{Req.}$ and its hash, vehicle's Pseud. Cert., and $Status_{Pseud.\ Cert.\ vehicle}$ (received from and signed by MA).

5. 3 Similar to Step 5.3 in Vehicle Certificate Provisioning.

Step 6: After decrypting received message and checking validity of sender's signature, and if all checks are valid, then:

6. 1 Application Server will verify that the vehicle has an access privilege to requested service. If vehicle doesn't have an access privilege, then Application Server will reply back to RA, so that RA will send deny message to vehicle and will reject future requests from it for same application. Also, if this incident occurred frequently within a certain time period, RA will inform Reward System to penalize vehicle. Else, if vehicle has an access privilege, then **Application Server** will send to **Application Manager** a list of available LBS Providers in its domain and $List_{request}$, sign message and encrypt it with Application Manager's public key.

6. 2 If vehicle has an access privilege, then **Reward System** will (reward) increase its account balance in Step 10.2, after LBS Provider notifies it with service delivery in Step 9.2. Else, if vehicle doesn't have an access privilege, then Reward System will (penalize) set its balance to zero.

6. 3 Similar to Step 6.2 in Vehicle Certificate Provisioning.

Step 7: Application Manager decrypts received message and checks validity of Application Server's signature. If all checks are valid, then **Application Manager** will reply back to **Application Server** with list of trusted available LBS Providers (using its Internal Blacklist) that contains only trusted LBS Provider(s) available in list sent by Application Server; along with Pseud. Cert. and trust level of each, sign message and encrypt it with Application Server's public key.

Step 8: Application Server decrypts received message and checks validity of Application Manager's signature. If all checks are valid, then **Application Server** will send to

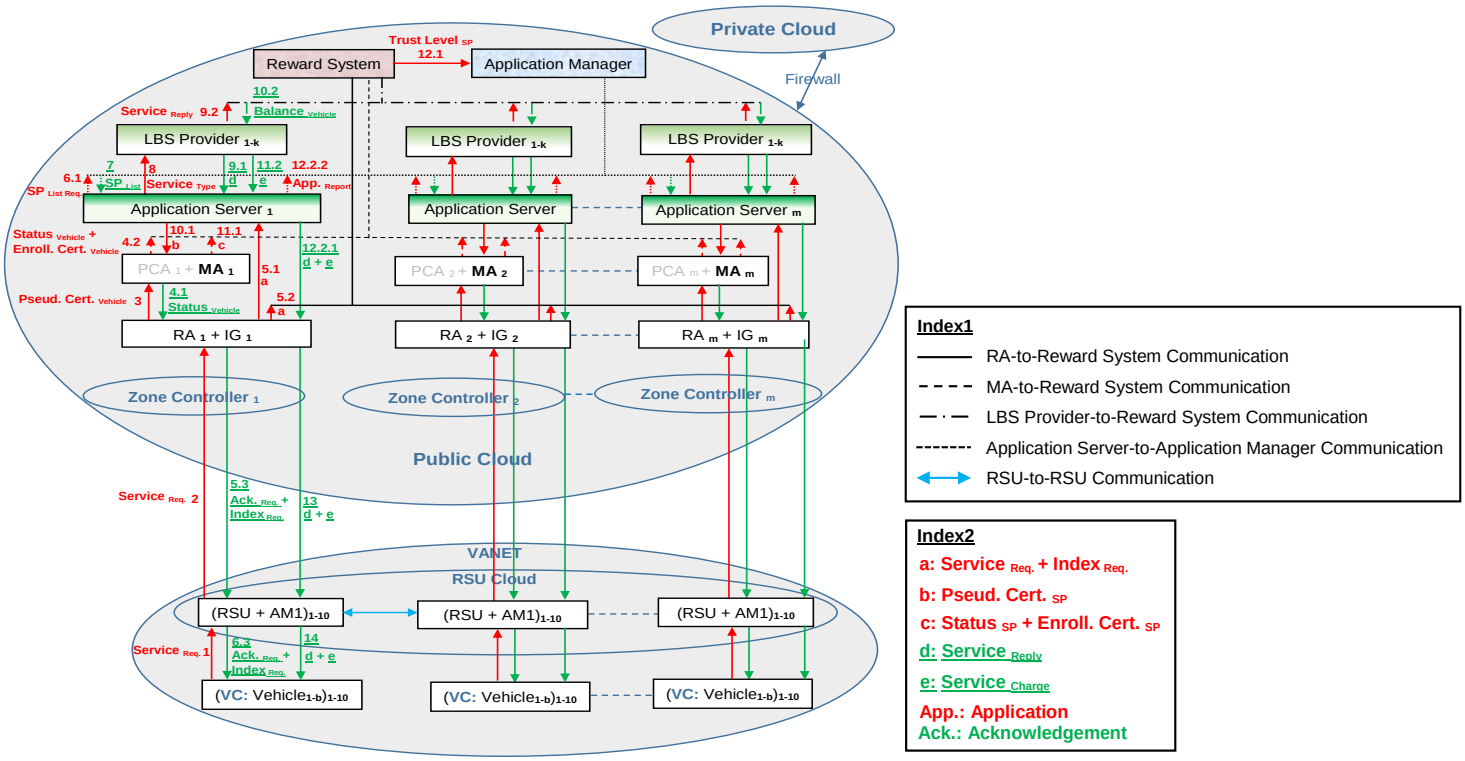

Figure 7. LBS Request in Vehicular Cloud Computing

available **LBS Provider** that has highest trust level and could provide needed service a message; that includes: service type and estimated duration, vehicle's location and speed (before service) and Status$_{\text{Pseud. Cert. vehicle}}$, Index value, sign whole message and encrypt it with LBS Provider's public key.

Step 9: LBS Provider decrypts received message and checks validity of Application Server's signature. If all checks are valid, then **LBS Provider** will:

9. 1 and **9.2** Reply back to **Application Server** (encrypt message with Application Server's public key), and send to **Reward System** (encrypt message with Reward System's public key) a signed Service$_{\text{Reply}}$; that includes: service needed and duration, vehicle's location and speed (after service), and Index value; along with its (LBS Provider's) Pseud. Cert. Note, if LBS Provider didn't reply Application Server within a certain time period, Application Server will send request to another available trusted LBS Provider, and will notify Application Manager with this incident *(LBS Provider Incident Report)*; so that Application Manager decreases LBS Provider's trust level. Also, Application Manager will in turn inform Reward System to decrease LBS Provider's trust level. However, if the same LBS Provider repeated again this incident, Application Manager will forward all Incident Reports of LBS Provider to corresponding MA, so that MA decides whether to revoke its Pseud. Cert. as in LBS Provider Certificate Revocation. LBS Provider Incident Report contains details about LBS Provider, such as: Pseud. Cert., organization's location; address; name; and type, service type(s) failed to be provided, total number of times denied to provide requested service, and total failure time.

Step 10: Similar to Step 6 here.

10. 1 Application Server will send to **MA** the LBS Provider's Pseud. Cert. and hash of Service$_{\text{Req.}}$, sign message and encrypt it with MA's public key.

10. 2 Reward System will increase vehicle's balance; then, reply to **LBS Provider** with new balance and Index value, sign message and encrypt it with LBS Provider's public key.

Step 11: Similar to Step 6 here.

11. 1 MA checks that LBS Provider is not in its internal Blacklist. If all checks are valid, then **MA** will send to **Reward System** a signed message that contains: received message (in Step 10.1); along with LBS Provider's enrollment certificate and Status$_{\text{Pseud. Cert. LBS Provider}}$, and encrypt whole message with Reward System's public key.

11. 2 LBS Provider will reply back to **Application Server** with a signed message that contains: Index value and Service Charge (= Amount Due – vehicle's balance), along with its Pseud. Cert. and received message (in Step 10.2 signed by Reward System), and encrypt whole message with Application Server's public key.

Step 12: Similar to Step 6 here.

12. 1 Reward System will increase trust level of LBS Provider, then it will send to **Application Manager** a signed message that includes: LBS Provider's Pseud. Cert. and trust level; and Index value, and encrypt message with Application Manager's public key.

12. 2 Application Server checks that LBS Provider has considered points in vehicle's balance when calculating Service Charge, to guarantee fair charge for vehicle. If all checks are valid, then **Application Server** will:

12. 2. 1 Reply back to **RA** with a signed message that includes: Service$_{\text{Reply}}$ (received in Step 9.1 signed by LBS Provider); Index and Service Charge (received in Step 11.2 signed by LBS Provider); LBS Provider's Pseud. Cert. and hash of Service$_{\text{Req.}}$, and encrypt whole message with RA's public key.

12. 2. 2 Send a signed *Application Report* to **Application Manager** that contains details of each service; such as: $Service_{Reply}$ (received in Step 9.1 signed by LBS Provider); Index and Service Charge (received in Step 11.2 signed by LBS Provider); LBS Provider's Pseud. Cert.; and message received from RA (in Step 5.1 signed by RA), and encrypt whole message with Application Manager's public key.

Step 13: RA decrypts received message (Reply) and checks validity of Application Server's signature. If all checks are valid, RA will encrypt Reply with vehicle's public key. Then, **RA** will reply back to **RSU** via corresponding Zone Controller with a signed message that includes encrypted Reply and hash of $Service_{Req.}$, and encrypt message with RSU's public key. Note, Reply is encrypted with vehicle's public key, to thwart another vehicle (malicious) from using same service without paying charge for it (that is without sending service request).

Step 14: Similar to Step11 in Vehicle Certificate Provisioning.

RSU Clouds

We propose to use RSU Cloud in LBS Request to guarantee service delivery for a vehicle. RSU Cloud allows RSU-to-RSU communication where each RSU forwards data to its nearest RSU. This is needed if a vehicle gets out of coverage range of a certain RSU_x before receiving a reply for its request. In this case, RSU_x will perform certain calculations depending on the vehicle's location, direction, speed, and destination; in order to determine RSU_y that the vehicle has entered its range. After that, RSU_x will forward vehicle's reply to RSU_y. Thus, using RSU Cloud in this way, is efficient as it removes the burden from RSU_x to forward the reply to all its nearest or neighboring RSUs, which may cause delay in receiving the reply. In addition, this saves the vehicle's time and reduces the communication overhead, as it doesn't need to send another new service request for the same service. Note, this procedure is also valid if RSU_y exists in a new zone, that is if the vehicle entered the domain of a new zone.

V. Security and Privacy Analysis of Proposed System

This section provides analysis of the privacy and security of the proposed system. We highlight the main role of our system entities to achieve privacy and security during the execution of the different system phases. Then, we provide a detailed description of security and privacy considerations of system.

A) Security and Privacy Analysis of System Entities

- **Address Mask (AM1):** To maintain vehicle's location privacy and mitigate its tracking; AM1 before sending: Pseud. $Cert._{Req.}$ or $Service_{Req.}$ to RA and Misbehavior Reports to MA, masks locations of sender by changing its source addresses; to avoid linkage of locations to source addresses (known as vehicle tracking). Note, sometimes the IP address enables an adversary to estimate the location of an entity [16]. Also, AM1 shuffles all Misbehavior Reports received together, to prohibit MA from determining paths of Reporters, and so providing sender's route privacy.
- **Registration Authority (RA):** It links Index value to vehicle's / LBS Provider's data, to reject future Pseud. $Cert._{Req.}$ or $Service_{Req.}$ from misbehaved entities. So, RA using its Internal Blacklist, acts as a request filter for both PCA and Application Server, and thus prevents Denial of Service (DoS) attack. Also, RA sends to PCA an Index instead of vehicle's / LBS Provider's Pseud. $Cert._{Req.}$, to avoid linking of request with generated Pseud. Cert. by PCA, and so providing sender's anonymity and preventing identification. **Index Generation (IG):** It shuffles Pseud. $Cert._{Req.}$ and $Service_{Req.}$ received with other requests, to prohibit PCA and Application Server from determining vehicle's path, so maintaining sender's route privacy. Also, IG generates a new independent Index value for each vehicle's (or LBS Provider's) Pseud. $Cert._{Req.}$ and $Service_{Req.}$, which prevents correlation of requests of same vehicle and mitigates traceability. Note, Indices assigned to same entity are not serial to prevent linkage of entity's requests together, and so maintains sender's anonymity and avoids traceability. Index value is sent to an LBS Provider instead of vehicle's Pseud. Cert., to avoid linking of Pseud. Cert. with service request by an LBS Provider, and so achieving sender's anonymity and preventing traceability. Since independent Index is assigned for each vehicle request, request of a certain vehicle is not be easily distinguished from other vehicles' requests. So, using Index values increases size of anonymity set of message sent, and also guarantees Perfect Backward / Forward Privacy; i.e. resolution of one Index doesn't reveal other Indices (or messages). In addition to previous security services, Index values also provide other security services, such as: increasing sender's confidence that the message was sent to correct receiver, increasing receiver's trust that the message was sent from correct sender, providing a challenge / response means between communicating entities.
- **Pseudonym CA (PCA):** To preserve sender's privacy and mitigate traceability by RA, PCA sends new Pseud. Cert. to requested entity via: RSU (for a vehicle) or AM2 (for an LBS Provider), without RA being involved unlike [3]. Also, PCA encrypts new Pseud. Cert. with requested entity's public key to prevent Man-in-the-Middle (MitM) and masquerade attacks. **Misbehavior Authority (MA):** It acts as a firewall for Application Sever, as it checks that the vehicle's Pseud. Cert. that has sent $Service_{Req.}$, using its Internal Blacklist, is not revoked. Thus, avoiding having too many ineligible requests sent to Application Sever that may cause DoS attack. Also, CRL generated by MA should contain only the certificates of an entity after it started misbehaving. This would decrease size of CRL, and also provides unlinkability between certificates used before entity started misbehaving and after entity misbehaved; so preventing entity tracking.
- **Application Server:** Acts as a firewall for LBS Providers, as it verifies that the vehicle that has sent $Service_{Req.}$ has an access privilege to requested application. So, this avoids having too many ineligible requests sent to LBS Provider that may result in DoS attack. Also, if vehicle doesn't have an access privilege to requested application, Application Server will reply back to RA, to reject future requests sent from the same vehicle for the same application. In addition, Application Server acts as a firewall for vehicles, as it sends legitimate

requests to trusted LBS Providers, whose certificates are not revoked. Besides that, Application Server doesn't send to LBS Provider the vehicle's Pseud. Cert. to prevent linkability between vehicle's Pseud. Cert. and request, and so maintaining sender's anonymity and preventing traceability.

- **Application Manager:** It sends to Application Servers the updated list of trusted LBS Providers in each Application Server's domain, using its Internal Blacklist.

B) Security and Privacy Considerations of Proposed System

1)To maintain privacy against attackers from **inside** system: **(a)** There is **organizational separation** between our system entities; e.g. Cloud Manager, RA, and PCA are run completely by separate different trusted organizations. **(b)** Our system design ensures that no single entity owns enough information that may allow tracking of a vehicle, known as **Distributed Resolution,** e.g. vehicle sends Pseud. Cert.$_{\mathbf{Req.}}$ to RA; but RA doesn't see new Pseud. Cert. Also, PCA issues Pseud. Cert., yet it doesn't know the receiver (vehicle) of Pseud. Cert. This is similar to [17], where a blind signature scheme is used and multiple authorities need to cooperate to resolve identity. **(c) Preventing Data Correlation**, e.g. vehicle's data included in Pseud. Cert.$_{\mathbf{Req.}}$ stored by RA cannot be correlated with vehicle's public keys included in Pseud. Cert. stored by PCA, and so avoiding identification.

2)To preserve privacy against attackers from **outside** system: **(a)** Our system uses **multiple encryption** for confidentiality similar to The Onion Router (TOR) [18], where messages are encrypted multiple times. **(b)** Frequent **certificate changes** should take place, to thwart a malicious entity from masquerading as different identities via compromising a single entity (known as Sybil Attack). However, vehicles cannot store too many certificates, or have continuous connectivity to system to receive certificates when needed, so we use a certificate validity time period of three days. **(c)** Other techniques used in our system include: **i) Message Signing** for sender authentication (also for message non-repudiation, accountability and trust), **ii) Hash Function** for message integrity, and **iii) Timestamps** to avoid replay attacks.

VI. Conclusion

In this paper, we proposed a secure architecture for VCC that maintains security and privacy of drivers while accessing LBS applications. The proposed system issued pseudonym certificates to both vehicles and LBS Providers, which are used to sign messages, where different system entities needs to cooperate to revoke these certificates. Our system used Index values for secure efficient Certificate Provisioning, Certificate Revocation and LBS Request. Unlike other works, the proposed architecture integrated all processes, such as: Certificate Provisioning and Revocation for both a Vehicle and an LBS Provider as well as LBS Request, into one system to be used in VCC. Also, the proposed system considered certificate revocation for an LBS Provider, which is according to our knowledge was not addressed in other works for VCC. Moreover, in our system, we used a novel Reward System, and a novel idea that allows RSU Cluster Formation where RSU acts as CH. Besides that, we introduced to use RSU Clouds to guarantee service delivery for a vehicle. A main challenge was to provide real-time services while maintaining location privacy of driver. In the future, we aim to provide cost and performance analysis for our system to provide more practical solution. Also, we need to evaluate security and privacy of proposed system on different internal and external attacks.

References

[1] M. Mansour, A. Fahmy, and M. Hashem, "Maintaining location privacy and anonymity for vehicle's drivers in VANET," in *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, issue 11, pp. 8-40, 2012.

[2] Artail, Hassan, and Noor Abbani, "A pseudonym management system to achieve anonymity in vehicular Ad hoc networks," in *IEEE Transactions on Dependable and Secure Computing*, vol. 13, issue 1, pp. 106-119, 2016.

[3] Whyte, William, et al., "A Security Credential Management System for V2V Communications," in *IEEE Vehicular Networking Conference, VNC*, 2013.

[4] Dietzel, S., Petit, J., Heijenk, G., and Kargl, F., "Graph-based metrics for insider attack detection in VANET multihop data dissemination protocols," in *IEEE Transactions on Vehicular Technology*, vol. 62, issue 4, pp. 1505–1518, 2013.

[5] B. Glas, and C. Gebauer, "Safety & Security: Synergies and Challenges of Integrity-protected Bus Communication," in *Embedded Security in Cars Conference, ESCAR*, Berlin, Germany, 2015.

[6] M.B. Mansour, C. Salama, H.K. Mohamed, and S.A. Hammad, "SEVECLOUD: A Secure Privacy-Preserving Robust Protocol for Vehicular Cloud Computing," 2016, unpublished.

[7] Kumar, Neeraj, et al., "An intelligent clustering scheme for distributed intrusion detection in vehicular cloud computing," in *Cluster Computing*, vol. 18, issue 3, pp. 1263-1283, 2015.

[8] Abumuhfouz, Ismail M., and Kiho Lim, "Protecting Vehicular Cloud Against Malicious Nodes Using Zone Authorities," in *IEEE SoutheastCon*, 2015.

[9] Lim, Kiho, Ismail M. Abumuhfouz, and D. Manivannan, "Secure Incentive-Based Architecture for Vehicular Cloud," in *14th International Conference on Ad-Hoc, Mobile and Wireless Networks, ADHOC-NOW*, Springer International Publishing, pp. 361-374, 2015.

[10] Kang, W. M., Lee, J. D., Jeong, Y. S., and Park, J. H., "VCC-SSF: Service-Oriented Security Framework for Vehicular Cloud Computing," in *Sustainability*, vol. 2, issue 7, pp. 2028-2044, 2015.

[11] Zhou, J., Dong, X., Cao, Z., and Vasilakos, A. V., "Secure and Privacy Preserving Protocol for Cloud-Based Vehicular DTNs," in *IEEE Transactions on Information Forensics and Security*, vol. 10, issue 6, pp. 1299-1314, 2015.

[12] Hussain, R., Rezaeifar, Z., Lee, Y. H., and Oh, H., "Secure and privacy-aware traffic information as a service in VANET-based Clouds," in *Pervasive and Mobile Computing*, vol. 24, pp. 194-209, 2015.

[13] Sedjelmaci, H., Senouci, S. M., "An accurate and efficient collaborative intrusion detection framework to secure vehicular networks," in *Computers & Electrical Engineering*, vol. 43, pp. 33-47, 2015.

[14] N. BiBmeyer, H. Stiibing, E. Schoch, S. Gotz, J. P. Stolz, and B. Lonc, "A generic public key infrastructure for securing Car-to-X communication," in *18th ITS World Congress*, 2011.

[15] Rawshdeh, Z.Y., and Mahmud, S.M., "Toward strongly connected clustering structure in vehicular ad hoc networks," in *70th IEEE Vehicular Technology Conference Fall, VTC 2009-Fall*, pp. 1–5, 2009.

[16] Internet Engineering Task Force (IETF) Geopriv Working Group, "Geopriv Requirements". [Online]. Available: https://datatracker.ietf.org/doc/rfc3693/

[17] F. Schaub, F. Kargl, Z. Ma, and M. Weber, "V-tokens for conditional pseudonymity in VANETs," in *IEEE Wireless Communications & Networking Conference, WCNC*, pp. 1-6, Sydney, Australia, April 2010.

[18] "Tor: Anonymity Online". [Online]. Available: http://tor.eff.org/index.html.en