November 2021

SECURE SMART TRAFFIC LIGHT SYSTEM USING AUXILIARY DATA IN COOPERATIVE AWARENESS MESSAGE

Jayashree Panda

Priyanka Bansal

Raghu Rajendra Arur

Ganesh Kondaveeti

Poornima Angaragatti

SECURE SMART TRAFFIC LIGHT SYSTEM USING AUXILIARY DATA IN COOPERATIVE AWARENESS MESSAGE

AUTHORS:
Jayashree Panda
Priyanka Bansal
Raghu Rajendra Arur
Ganesh Kondaveeti
Poornima Angaragatti

ABSTRACT

Current smart traffic lights (STLs) can provide access to a traffic signal, which a vehicle, such as an emergency vehicle, can manipulate through a Cooperative Awareness Message (CAM) object. However, researchers have demonstrated that STLs can be easily hacked using an abusive CAM object. To address such a weakness, techniques are presented herein that secure a STL with auxiliary data that is attached to a CAM object, which can be used by a vehicle on-board unit (OBU) and smartphones. Aspects of the presented techniques reduce the encryption and decryption overhead that is incurred when attaching a security header as part of a CAM message. Further aspects of the presented techniques ensure that the authenticity and integrity of a CAM message can be validated through a roadside infrastructure (e.g., Roadside Units (RSUs)) and STLs with the help of a central server.

DETAILED DESCRIPTION

The narrative that is presented below employs a number of abbreviations. For convenience, various abbreviations, along with their associated description, are presented in Table 1, below.

Abbreviation	Description
CA	Certificate Authority
CAM	Cooperative Awareness Message
ITL	Intelligent traffic light
ITS	Intelligent traffic system

MQTT	Message Queuing Telemetry Transport
RSU	Roadside Unit
STL	Smart traffic light
TS	Timestamp
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle

Table 1: List of Abbreviations

Additionally, as described herein a 'CAM packet' and a 'CAM object' may be understood to refer to the same artifact and, thus, may be referred to interchangeably.

Intelligent traffic lights (ITLs) are critical cyber-physical systems that help smart cities reduce road congestion and vehicle emissions. Current smart traffic lights (STLs) provide access to a traffic signal, which a vehicle, such as an emergency vehicle, can manipulate through a CAM object that is carried via the Message Queuing Telemetry Transport (MQTT) communication protocol so that, when there is an emergency, the vehicle will receive a priority green light. However, such a feature has been adopted by many applications that also allow road users (such as cyclists, bikers, and four wheelers) to operate a signal while crossing an intersection junction. Additionally, such a capability also opens up a new frontier of cybersecurity. For example, researchers have demonstrated that STLs can be easily hacked using CAM objects.

CAM objects inform road users and the roadside infrastructure about each other's position, dynamics, and attributes. For a secure CAM message, each packet should be attached with a header. To protect the message's authenticity and integrity, such a header should contain a CA, a digest, and a timestamp. However, a stronger digest necessitates a better processor and increases the processing time and cost. Moreover, such a secured CAM can only be generated by a vehicular ad hoc network (VANET). The applications that use CAM objects don't follow any security structure and employ CAM objects only for Vehicle to Vehicle (V2V) or Vehicle to Infrastructure (V2I) communications.

Android-based applications that employ CAM objects may be used to hack a traffic light system. As just one example, consider a cyclist paradigm. For such a context, two

Netherland researchers have demonstrated how a traffic light may be hacked using a cyclist application.

In particular, the researchers made use of the Frida dynamic instrumentation toolkit to decompile the CAM objects that were exchanged over the MQTT communication protocol. Every new MQTT connection was seen as a new cyclist. Multiple Android applications use only the speed, latitude, and longitude to open up a signal. The researchers discovered that a single Hypertext Transfer Protocol (HTTP) POST request in a loop can keep a signal green forever and lead to traffic congestion in a city. Additionally, the researchers claimed that there is no way to distinguish between two different digital cyclists. Accordingly, it has been recommended that some kind of authentication and authorization be added in a central server to recognize the messages that are sent from Android applications and to block those that are from abusive users.

Within the context of the cyclist example that was described above, Figure 1, below, depicts elements of an illustrative CAM object that is carried over the MQTT communication protocol in which each new MQTT connection is seen as a new cyclist.

```
10      camp.publish.implementation = function (a, b, c, d) {
11          console.log('Hooking publish event');
12          this.deInit();
13          this.init();
14          console.log("Inited: " + this.isInited());
15          console.log("clientId: " + this.clientId.value);
16          Thread.sleep(0.5);
17          console.log("Connected: " + this.isConnected());
18          this.publish(a, b, c, d);
19      };
```

Figure 1: Illustrative CAM Object

To address the types of challenges that were described above, techniques are presented herein that support attaching auxiliary data (containing identification and timestamp information) to CAM messages. Such auxiliary data may be generated on the fly with the help of Roadside Units (RSUs) and can be used by smart traffic lights to check the authenticity and uniqueness of an application or vehicle that is operating a signal.

Aspects of the techniques presented herein leverage a number of existing technologies or capabilities. To begin, RSUs are employed to monitor a vehicle, collect

data, and transfer data to a central traffic management server. Under aspects of the techniques presented herein, RSUs are used for capturing additional information. Additionally, the format or structure of a CAM message includes a Special Vehicle Container field. Under aspects of the techniques presented herein, that field is extended to support auxiliary data containing a certificate authorization and a timestamp. Finally, while a CAM message includes a security header, aspects of the techniques presented herein obviate STL hacking through applications without requiring a security header thus avoiding any processor dependency.

Detecting and defending traffic system attacks has been the subject of extensive research. Proposed herein are techniques through which traffic light hacks can be avoided through use of a smart vehicle and the existing ITS framework.

To obtain a clear visualization of traffic on a road, along with V2V connectivity, data is collected from infrastructure RSUs and from various other sensors that are placed on the road. The collected data is disseminated to an Internet of vehicles (IoV) cloud that helps in managing the city-level traffic. RSUs play an important role in a VANET environment for privacy preservation. To protect the privacy of a vehicle, the issued certificate (certificate authorization) must be updated frequently by the RSUs. Figure 2, below, presents a high-level overview of such an environment.

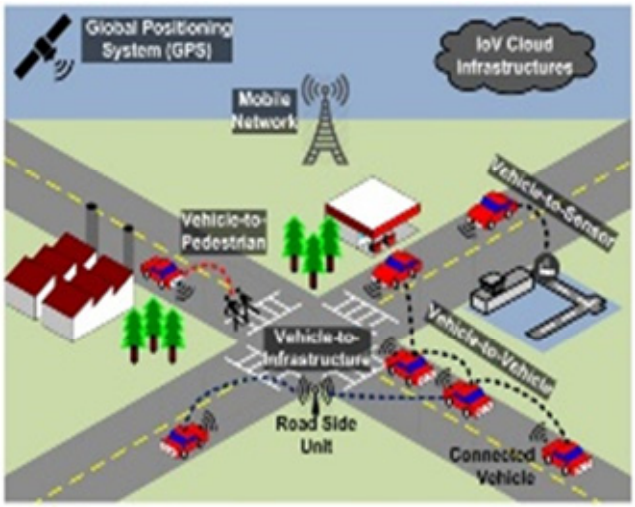

Figure 2: Illustrative Environment

Aspects of the techniques presented herein support the use of authorized and authenticated auxiliary data from an RSU that may be added to a CAM object. Android

applications can use the same object to interact with a smart traffic light system (TLS). An STL needs to validate the auxiliary data with a central server before opening up the signal. Use of aspects of the techniques presented herein incurs less encryption and decryption overhead (which is needed for a CAM security header). The same auxiliary data or metadata may also be used by VANETs if a security header is absent in CAM objects. Figure 3, below, portrays elements of the final request packet from a vehicle to a STL.

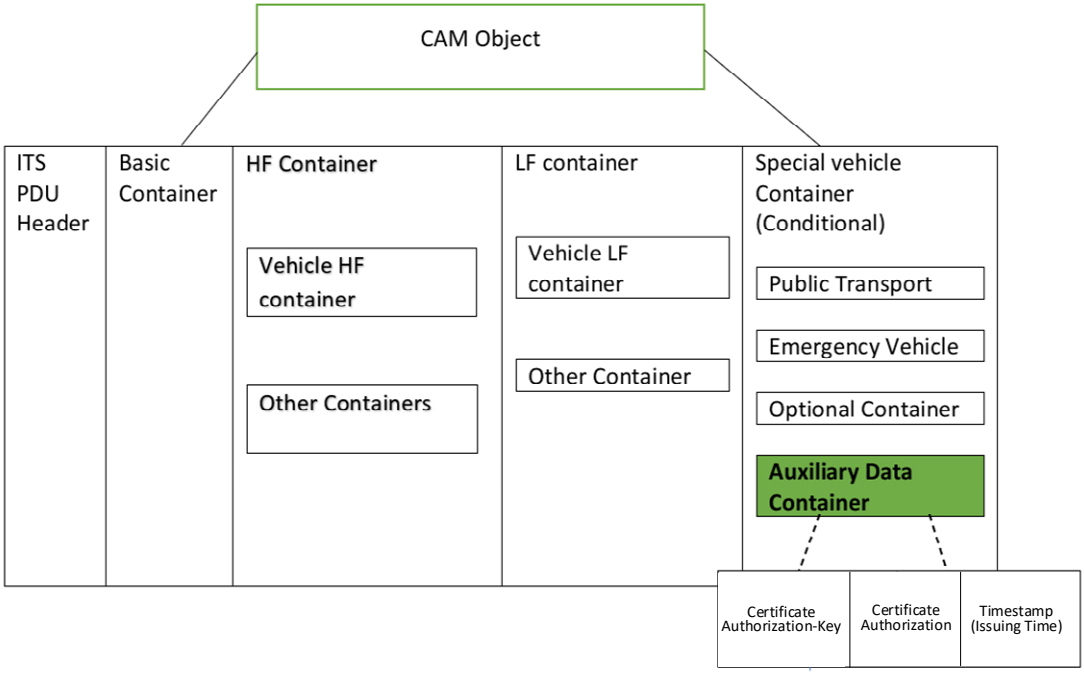

Figure 3: Exemplary Vehicle-to-STL Packet Request

Aspects of the techniques presented herein leverage various existing frameworks, such as RSUs, V2I communication, certificates, certificate authorization updates, etc. Further, aspects of the presented techniques encompass three major steps:

- The generation of auxiliary data with physical authentication.
- Updates to the auxiliary data (e.g., a certificate authorization and a timestamp).
- Auxiliary data validation in a STL before a signal is opened up.

Each of the above steps will be described and illustrated in the narrative that is presented below.

A first step includes the generation of auxiliary data with physical authentication. Physical authentication typically suggests a "seeing and believing" concept. An exemplary flow for such an approach, as illustrated in Figure 4, below, may include the following activities:

- It may be assumed that some of the RSUs will be equipped with a camera and an analyzer that can take a picture of the license plate of a moving vehicle and extract the license information from the picture.
- Vehicles without a license plate (such as, for example, a cyclist or an electric motor vehicle) should carry a radio-frequency identification (RFID) tag for physical authentication.
- Each interested vehicle may send a GET request with its vehicle number or RFID number to retrieve a key pair, a certificate authorization, and metadata.
- When an RSU receives the GET request, the RSU validates the vehicle number with captured license information. If the capture is available within an RSU, then the RSU generates a key-pair, a certificate authorization, a generation time stamp, and a certificate authorization expiry time.

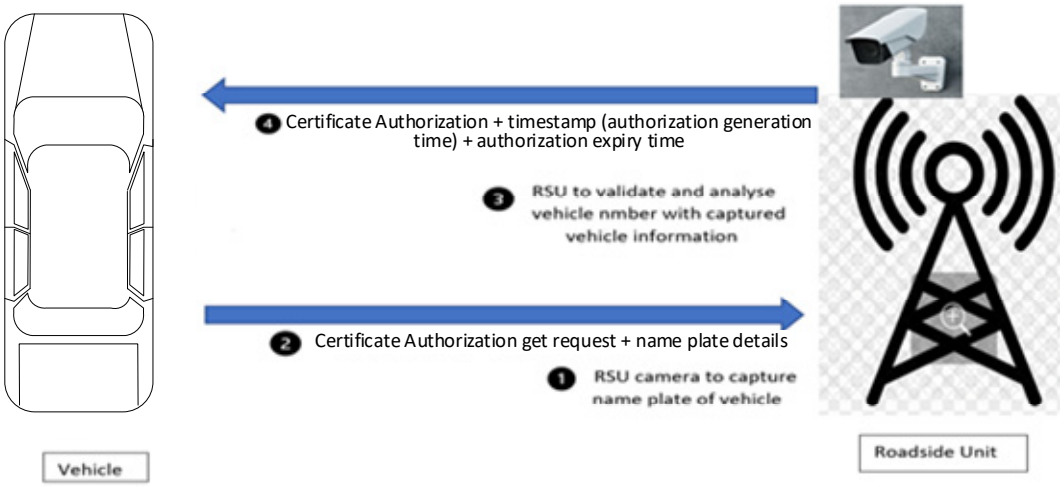

Figure 4: Illustrative Authentication Activities

A second step may include providing updates to the auxiliary data (e.g., a certificate authorization and a timestamp). For example, a generated certificate authorization may

only be valid only within a local area and it must be updated in the next RSU. The certificate authorization updating flow, as shown in Figure 5, below, may include the following activities:

- Once a vehicle obtains an initial valid key pair and certificate, the vehicle can refresh them in the next RSU.
- The local or next RSU can validate and re-issue a fresh certificate authorization.
- An application can send an UPDATE request with the existing data (e.g., a key pair, a certificate authorization, and timestamp details).
- Thereafter, the RSU can validate the key and certificate. If such a validation is successful, then the RSU can generate a new key, its own certificate authorization, and current time stamp details.

Such a process (of updating the certificate authorization and metadata) needs to continue until it hits the local area where traffic signals are available.

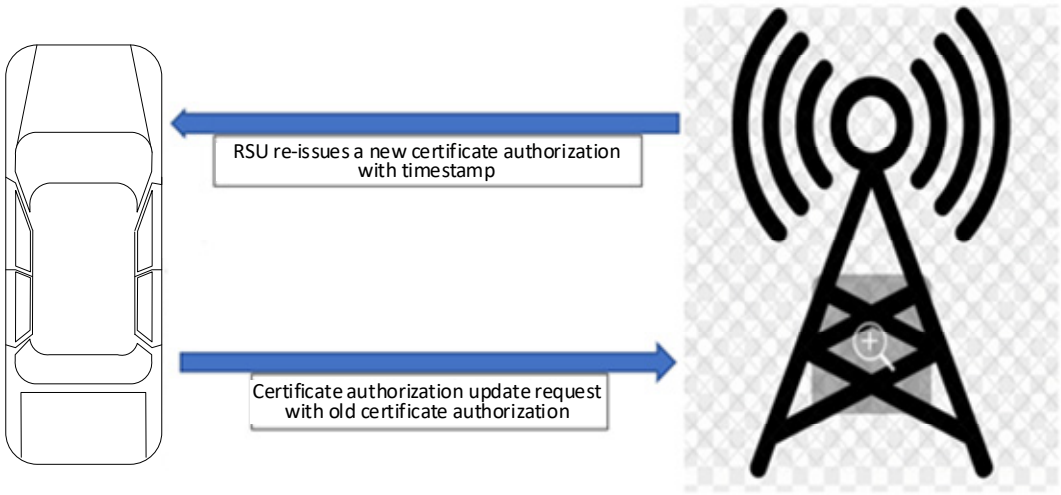

Figure 5: Illustrative Update Activities

A third step involves auxiliary data validation in a STL before a signal can be accessed by a vehicle. For example, a vehicle that is interested in accessing a signal automatically, without stopping at a junction, can send a CAM object with a certificate authorization and timestamp details to the traffic light. The validations that take place within a STL may include the following activities, which are illustrated in Figure 6, below:

- The STL sends the message to a central traffic system server. The central server confirms if there are any security header attached to the CAM object. According to aspects of the techniques presented herein, a CAM object will not include a security header.
- The CAM message will be opened to verify if a special vehicle container is available.
- If a special vehicle container is present then the server validates the certificate authorization and stores the certificate authorization and time stamp as a key-value pair in a database.
- If the same message again reaches the STL, it queries the database and confirms that the packet is already processed. Accordingly, the message can be dropped, which can avoid a replay of the same CAM message.

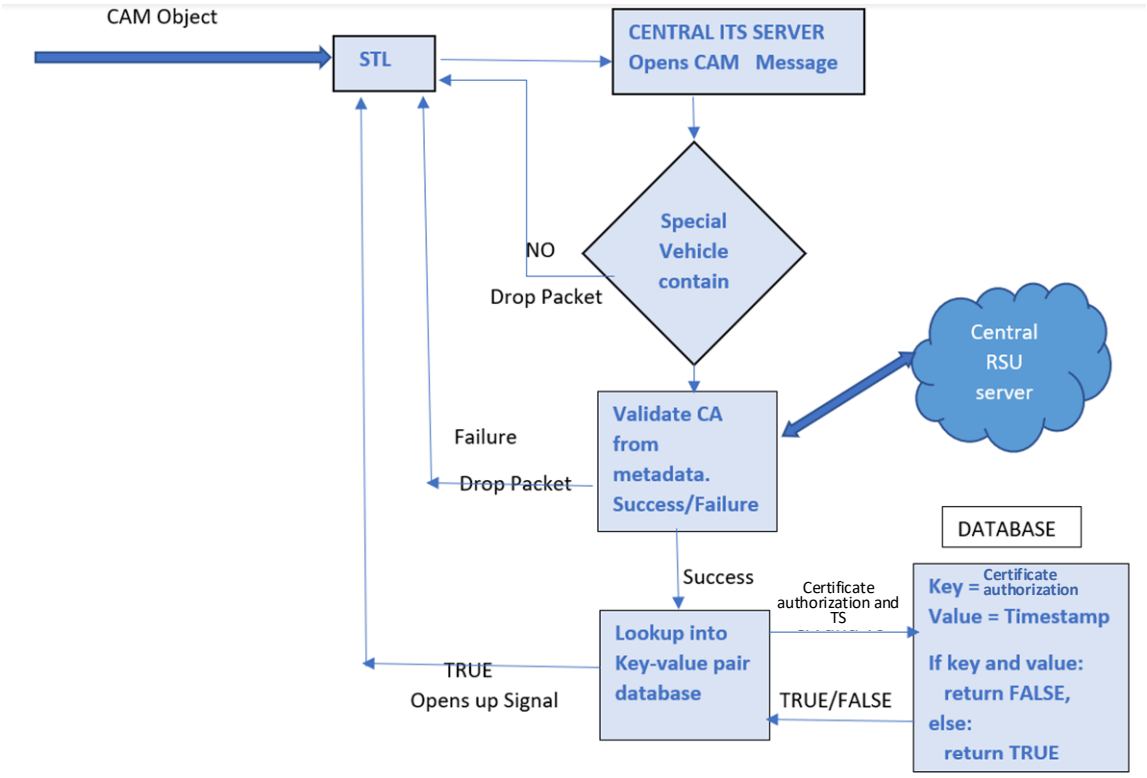

Figure 6: Illustrative Validation Activities

Figure 7, below, presents elements of an overall workflow according to aspects of the techniques presented herein.

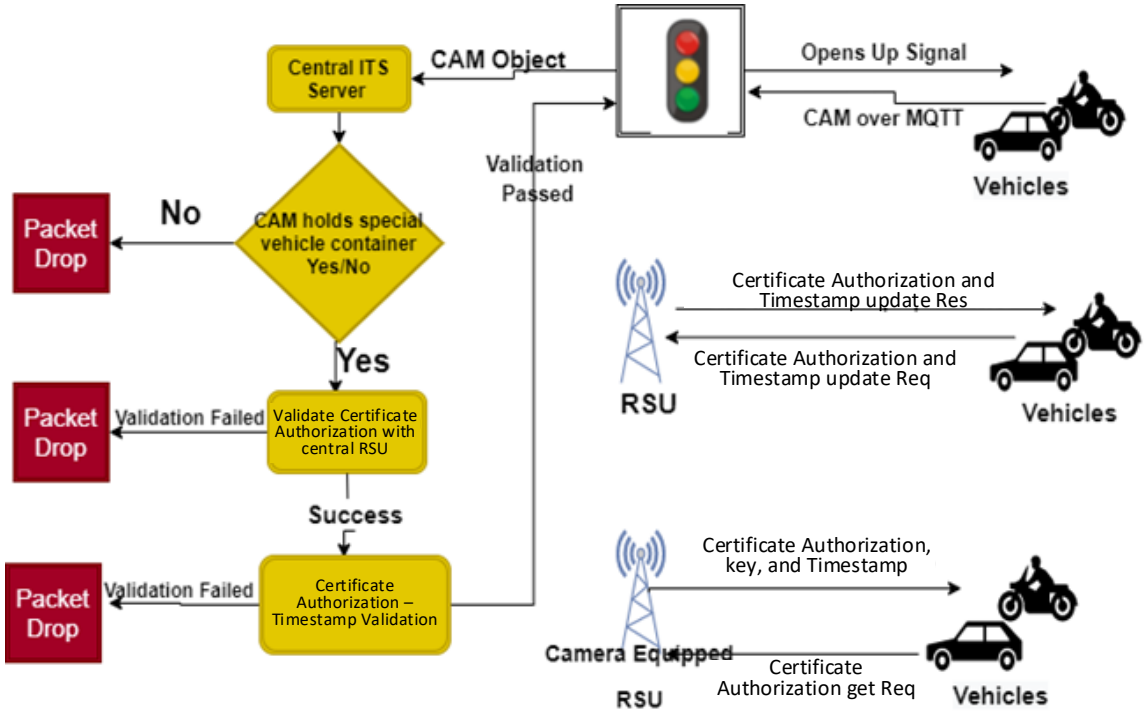

Figure 7: Exemplary Workflow

As described and illustrated in the narrative that was presented above, aspects of the techniques presented herein provide several features and advantages that can be realized within STL systems. For example, lightweight auxiliary data that contains a certificate authorization and a timestamp can be included in CAM messages sent by applications, which often cannot afford to attach a security header. Advantageously, the certificate authorization and the timestamp can be generated between an RSU and an application using physical verification (e.g., via license plates, RFID tags, etc.) provided by RSUs (e.g., using cameras, etc. at the RSUs). The auxiliary data can be refreshed at each RSU using the physical verification to avoid replay attacks. Finally, the authenticity and integrity of CAM messages can be validated for signal operation at a STL using the auxiliary data.

In summary, techniques have been presented herein that secure a STL with auxiliary data that is attached to a CAM object which can be used by a vehicle on-board unit (OBU) and smartphones. Aspects of the presented techniques reduce the encryption

and decryption overhead that is incurred when attaching a security header as part of a CAM message. Further, aspects of the presented techniques ensure that the authenticity and integrity of CAM messages can be validated through the roadside infrastructure (e.g., RSUs) and STLs with the help of a central server.