Formalization and Verification of VANET

Ran Li[1], Huibiao Zhu*[1], Lili Xiao[1], Jiaqi Yin[1], Yuan Fei[2], Gang Lu*[1]
[1]Shanghai Key Laboratory of Trustworthy Computing,
East China Normal University, Shanghai, China
[2]School of Information, Mechanical and Electrical Engineering,
Shanghai Normal University, Shanghai, China

***Abstract*—Vehicular Ad Hoc Network (VANET) is a subclass of Mobile Ad Hoc Network (MANET) types. As a key part of the Intelligent Transportation Systems (ITSs) framework, it can be used not only to provide value added services, but also to guarantee the security of ITS. Since VANET is extensively applied, its security is of great significance.**

In this paper, we model the architecture of VANET using process algebra Communicating Sequential Processes (CSP). By utilizing model checker Process Analysis Toolkit (PAT), we verify five properties (deadlock freedom, divergence freedom, data leakage, vehicle faking and RSU faking) of the model and find that the proposed architecture may cause data leakage. Hence, we improve the model by encrypting the messages with receiver's public key to cope with this problem. The new verification results show that our study can guarantee the security of VANET.

***Index Terms*—VANET; Security; CSP; Modeling; Verification**

I. INTRODUCTION

While the rapid growth of driverless vehicles has been fueled by the development of vehicle industry and wireless communication technology, VANET is actually the supporting infrastructure and paves the way for driverless vehicles.

VANET is the application of traditional MANET in the field of intelligent traffic [1]. The basic architecture of VANET is demonstrated in Fig. 1. Each vehicle has an On Board Unit (OBU) and one or more Application Units (AUs). Road Side Unit (RSU) is the device installed along road side and communicates with OBUs using Dedicated Short Range Communication (DSRC) technology. Moreover, RSU can also get in touch with the gateway to the Internet.

Fig. 1 illustrates that communication in VANET can be divided into three domains: in-vehicle domain, ad hoc domain and infrastructure domain [1]. In-vehicle domain contains an OBU and one or more AUs. Infrastructure domain communication is between RSUs and infrastructure. Ad hoc domain is composed of two types of communication. One is Vehicle-to-Vehicle (V2V) communication which means a vehicle can connect with other vehicles, and the other is Vehicle-to-Infrastructure (V2I) communication which represents that a vehicle can exchange information with infrastructure.

Since VANET is an emergent technology, it has massive challenges of security issues. Many studies have been carried out on the topic of communications security in VANET [2]–[4]. However, the previous solutions cannot comfirm the confidentiality-integrity-availability (CIA) property [5]. To secure communications in VANET, a newly proposed architecture uses end-to-end authentication to avoid intrusion in VANET and considers VANET as a hierarchical model to decrease the number of message exchanges [5]. However, the proposed architecture is not verified formally.

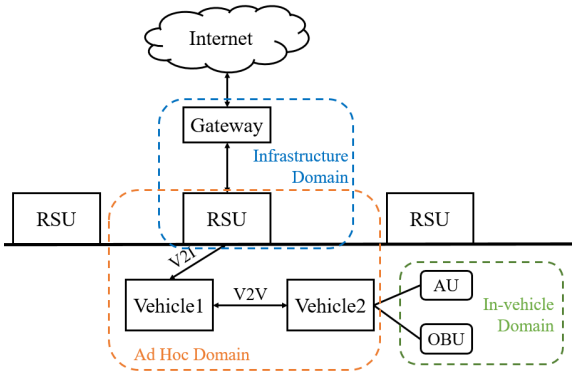

Fig. 1. Basic Architecture of VANET

In this paper, we formalize the model of the proposed architecture of VANET [5] using process algebra CSP [6], [7]. Additionally, we use model checking tool PAT [8], [9] to verify some related properties, including deadlock freedom, divergence freedom, data leakage, vehicle faking and RSU faking. From the verification results, we find that the original proposed architecture [5] is not safe and it may cause data leakage problem. Hence, we modify the original model by means of encrypting messages with receiver's public key to solve this problem. The verification results of the improved model show that the modification is truly effective.

The remainder of this paper is organized as follows. In Section II, we give a brief introduction to VANET which contains the proposed architecture [5] and the flow of communications between entities. Besides, the process algebra CSP is also introduced briefly in this section. Section III presents the formalized model of VANET [5]. Furthermore, Section IV is about the verification results of the original model and we also put forward some improvements of the model. We conclude our work and propose the future work in Section V.

II. BACKGROUND

In this section, we briefly explain the proposed architecture of VANET [5] and focus on the communication flow of it. In addition, an introduction to CSP is given as well.

*Corresponding authors: hbzhu@sei.ecnu.edu.cn (H. Zhu), glu@cs.ecnu.edu.cn (G. Lu).

DOI reference number: 10.18293/SEKE2020-011

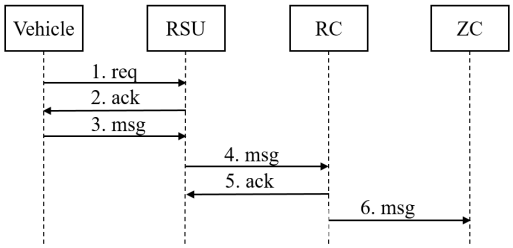

Fig. 2. In Ad Hoc and Infrastructure Domain

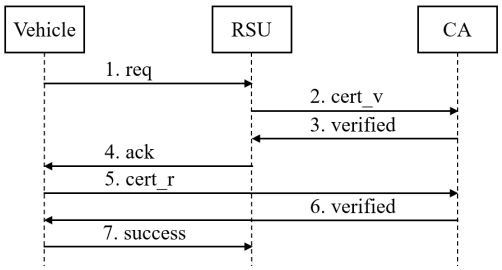

Fig. 3. In-vehicle Domain (Adapted from[5])

A. VANET

In order to ensure communications security, a newly proposed architecture adopts end-to-end authentication to avoid intrusion in VANET and considers VANET as a hierarchical model to decrease the number of message exchanges.

The proposed architecture covers five entities: Vehicle, RSU, RSU Controller, Zone Controller and CA. Vehicle and RSU are the same as the entities in the basic architecture mentioned above, while RSU Controller, Zone Controller and CA are newly introduced entities in VANET.

- **RSU Controller:** It is in charge of the communications in an area with a number of RSUs.
- **Zone Controller:** It controls multiple RSU Controllers.
- **CA (Certification Authority):** It is responsible for distributing the certificates to RSUs, RSU controllers and Zone controllers. Each certificate contains the entity's ID, public key and expiry time.

To ensure security applications in ad hoc domain communication and infrastructure domain communication, the proposed architecture emphasizes end-to-end authentication process using certificates from CA. Fig. 2 shows the flow of this solution.

1. The vehicle sends a request to the RSU.

2. Once a RSU receives this request, it checks the expiry time and the threshold value of the message control. If they are both valid, it returns its ID and the timestamp. If not, it broadcasts an alarm notification.

3. Then, the vehicle executes Deffie-Hellman key exchange [10] with the RSU and uses the shared key to encrypt the message with Elliptic Curve Cryptography [11].

4. Afterwards, the RSU decrypts the message and passes it along with timestamp forward to the RSU controller.

5. If the timestamp is valid, the RSU controller returns its certificate with an ack to the RSU. It also proceeds to encrypt and sends the message along with its certificate to the Zone controller. Otherwise, the RSU controller broadcasts an alarm notification, i.e., it fails.

6. When the Zone controller receives the message, it checks the timestamp at first. If the timestamp is valid, the Zone controller decrypts the message and sends the message to other Zone controllers and base stations. Otherwise, it fails.

On the other hand, to ensure security in in-vehicle communication, the proposed architecture emphasizes the verification of the certificates. The flow is shown in Fig. 3. Here, RSU in Fig. 3 can also be a service provider. Due to the similarity in flows of them, we list the flow of the communication among the vehicle, the RSU and CA.

1. The OBU of a vehicle sends a service request which includes the certificate of the vehicle.

2. A RSU sends the certificate of the vehicle to CA to check its validity.

3. CA returns the verification result.

4. Once the RSU receives the verification result which shows the certificate is valid, it sends its own certificate to the vehicle along with an ack. If not, it refuses the request.

5. Then, the vehicle sends the received certificate to CA to check its validity.

6. CA returns the verification result.

7. Once the vehicle receives the verification result which shows the certificate is valid, the vehicle can use the service successfully and safely in AU. Otherwise, it rejects the service.

B. CSP

CSP, the abbreviation of Communicating Sequential Processes, is a process algebra proposed by C. A. R. Hoare [6]. The syntax of CSP provides many operators to express the actions of processes and their interactions. We give a brief definition of the syntax used in this paper as below.

$$P, Q \ ::= \ Skip \mid a \rightarrow P \mid c?x \rightarrow P \mid c!u \rightarrow P \\ \mid P||Q \mid P \Box Q \mid P \lhd B \rhd Q \mid P[[a \leftarrow b]]$$

- $Skip$ denotes that a process terminates successfully.
- $a \rightarrow P$ describes an object which first engages in the event a and then behaves exactly as described by P.
- $c?x \rightarrow P$ represents that a process receives a message through a channel called c and assigns the value of the message to x, and then behaves like process P.
- $c!u \rightarrow P$ means that a process sends message u through a channel called c and then behaves like process P.
- $P||Q$ indicates process P executes in parallel with process Q.
- $P \Box Q$ stands for external choice, which means that a process performs like P or Q and the choice is determined by the environment.
- $P \lhd B \rhd Q$ expresses conditional choice. If B is true, then the process behaves like P, otherwise behaves like Q.
- $P[[a \leftarrow b]]$ is the syntax of renaming and signifies an event a is replaced by b in process P.

III. MODELING VANET

In this section, we give the formalized model of VANET. First, we introduce the definitions of sets, messages and channels. On this basis, we formalize the model of VANET using CSP.

A. Sets, Messages and Channels

First, some sets are introduced in our formalized model for convenience. For communicators, we define the set **Vehicles** of the Vehicle components, **RSUs** of the RSUs and **Controllers** of the RSU Controller and the Zone Controller components.

Entity set involves entities mentioned above. Further, each entity has its own certificates, ID and keys. **Cert** is the set of certificates. **Id** is the set of IDs and **Key** is the set of keys. $Key = PUBK \cup PRIK$, where $PUBK$ denotes the public keys and $PRIK$ represents the private keys.

Moreover, we give definitions of **Content** of the content, **T** of the time and **State** of the states which contain true state and false state.

Based on the sets above, the messages are further abstracted as follows.

$$MSG_{cert} =_{df} \{msg_{reqv}.a.b.cert, msg_{rspv}.a.b.state| \\ a,b \in Entity \cup CA, cert \in Cert, state \in State\}$$
$$MSG_{prdc} =_{df} \{msg_{prdc}.a.b.E(k,c)| \\ a,b \in Entity \cup CA, k \in Key, c \in Content\}$$
$$MSG_{reqb} =_{df} \{msg_{reqb}.a.b.id.t|a,b \in Entity, id \in Id, t \in T\}$$
$$MSG_{reqs} =_{df} \{msg_{reqs}.a.b.rid.vid.cert| \\ a,b \in Entity, rid, vid \in Id, cert \in Cert\}$$
$$MSG_{ack} =_{df} \{msg_{ackr}.a.b.rid.ack.cert,\ msg_{ackrc}.a.b.ack.cert| \\ a,b \in Entity, rid \in Id, ack \in State, cert \in Cert\}$$
$$MSG_{data} =_{df} \{msg_{d1}.a.b.E(k,c), msg_{d2}.a.b.E(k,c).cert.t| \\ a,b \in Entity, k \in Key, c \in Content, cert \in Cert, t \in T\}$$
$$MSG_{c1} =_{df} MSG_{cert} \cup MSG_{prdc} \cup MSG_{reqb} \\ \cup MSG_{data} \cup MSG_{ack}$$
$$MSG_{c2} =_{df} MSG_{cert} \cup MSG_{reqs} \cup MSG_{ack}$$

MSG_{c1} is the set of messages in ad hoc and infrastrusture domain and MSG_{c2} consists of messages communicated in in-vehicle domain.

Besides, we use the symbols **E** and **D** to represent encryption function and decryption function respectively.

- **E(**k**,**msg**)** indicates that k is the key which is used to encrypt the message msg.
- **D(**k^{-1}**,**$E(k, msg)$**)** means that the corresponding decryption key k^{-1} can decrypt the message which is encrypted with k.

Then, we give the definitions of channels.

- channels of processes between legal entities, denoted by **COM_PATH**:
  $ComVC$, $ComRC$, $ComRcC$, $ComZcC$, $ComVR$, $ComRRc$, $ComRcZc$
- channels of intruders faking/intercepting processes, represented by **INTRUDER_PATH**:
  $FakeVR$, $InterceptRV$, $FakeRV$, $InterceptVR$, $FakeRRc$, $InterceptRcR$, $FakeRcR$, $InterceptRRc$, $FakeRcZc$, $InterceptZcRc$, $FakeZcRc$, $InterceptRcZc$
- channel of synchronization time: $Time$

The declarations of the channels are as follows.

Channel COM_PATH, $INTRUDER\_PATH$: **MSG**

B. Overall Modeling

We formalize the whole model as below. $VANET_0$ represents the system without considering intruders, while $VANET$ takes account of the attacks from intruders. The channels of our model are shown in Fig. 4.

$$VANET =_{df} VANET_0 \,[|INTRUDER\_PATH|]\, Intruder$$
$$VANET_0 =_{df} Vehicle \,||RSU\, ||RC\, ||ZC\, ||CA\, ||Clock$$

$Vehicle$, RSU, RC, ZC and CA describe the performance of vehicles, RSUs, RSU controllers, Zone controllers and CA respectively. The process called $Clock$ is used to realize the synchronization of time. Additionally, we use the process $Intruder$ to simulate intruders' actions, such as intercepting or faking messages.

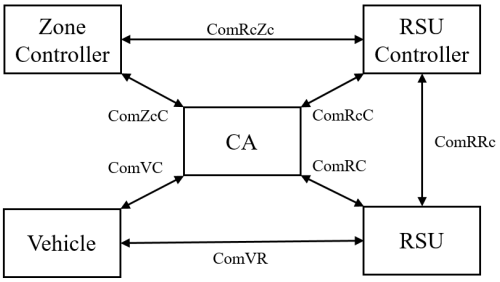

Fig. 4. Channels of VANET

C. Clock Modeling

When entities deliver messages, they need to check whether the timestamp is valid. So we define $Clock$ in order to realize the synchronization of all entities. The $Clock$ process is responsible for recording time and returning the current time whenever some entities want it.

$$Clock(t) =_{df} tick \rightarrow Clock(t+1) \,\Box\, Time!t \rightarrow Clock(t)$$

D. CA Modeling

CA provides the identities and certificates for all RSUs, RSU controllers and Zone controllers. Besides, CA can verify the entities by certificate verification. The model of CA is given as follows.

$$CA =_{df} ComRC!msg_{prdc}.C.R.E(ca_{kpri}, cert_r) \rightarrow CA \\ \Box ComRcC!msg_{prdc}.C.Rc.E(ca_{kpri}, cert_{rc}) \rightarrow CA \\ \Box ComZcC!msg_{prdc}.C.Zc.E(ca_{kpri}, cert_{zc}) \rightarrow CA \\ \Box ComRC?msg_{reqv}.R.C.cert_v \rightarrow \\ verified := valid(cert_v) \rightarrow \\ ComRC!msg_{rspv}.C.R.cert_v.verified \rightarrow CA \\ \Box ComVC?msg_{reqv}.V.C.cert_r \rightarrow \\ verified := valid(cert_r) \rightarrow \\ ComVC!msg_{rspv}.C.V.cert_r.verified \rightarrow CA$$

$valid(cert)$ is a function to verify whether the certificate is valid. If the certificate is valid, it returns $true$ and then the variable $verified$ is set to true.

E. Vehicle Modeling

We formalize the process $Vehicle$ using general choice $\Box$ to describe the two types of communication mentioned above. The model of $Vehicle$ is shown as below.

$$
\begin{aligned}
Vehicle_0 =_{df}\ & ComVR!msg_{reqb}.V.R.vid.t_{exp} \rightarrow \\
& ComVR?msg_{reqb}.R.V.rsu_{id}.t_s \rightarrow \\
& dh\_key\_change \rightarrow \\
& ComVR!msg_{data1}.V.R.E(k_{dh}, msg) \rightarrow Vehicle_0 \\
\Box\ & ComVR!msg_{reqs}.V.R.rsu_{id}.v_{id}.cert_v \rightarrow \\
& ComVR?msg_{ackr}.R.V.rsu_{id}.ack.cert_r \rightarrow \\
& \left(\begin{array}{l} \left(\begin{array}{l} ComVC!msg_{reqv}.V.C.cert_r \rightarrow \\ ComVC?msg_{rspv}.C.V.cert_r.verified \rightarrow \\ (success \rightarrow Vehicle_0) \end{array} \right) \\ \lhd (verified == true) \rhd (\ fail \rightarrow Vehicle_0\ ) \end{array} \right) \\
& \quad \lhd (ack == true) \rhd (\ fail \rightarrow Vehicle_0)
\end{aligned}
$$

In the first half of the model, we describe the behaviors of the vehicle in ad hoc or infrastructure domain and they correspond to Steps 1-3 in Fig. 2. t_{exp} is the expiry timestamp and t_s records when the RSU sends the reply. We also define an event called dh_key_change which denotes that the vehicle executes Deffie-Hellman key exchange with the RSU. After the execution, the vehicle and the RSU have a shared key k_{dh}, which is used to encrypt the message with Elliptic Curve Cryptography. The remaining actions correspond to Steps 1, 4, 5 and 6 in Fig. 3 and represent the actions of the vehicle in in-vehicle communication.

Since we have given the model without intruders, then we need to consider the existence of intruder actions. For example, we should allow intruders to fake or intercept messages. We describe the behavior of the intruder via renaming as follows and the channels that intruders involved are shown in Fig. 5.

$$
\begin{aligned}
Vehicle =_{df}\ & Vehicle_0[[\\
& ComVR!\{|ComVR|\} \leftarrow ComVR!\{|ComVR|\}, \\
& ComVR!\{|ComVR|\} \leftarrow InterceptVR!\{|ComVR|\}, \\
& ComVR?\{|ComVR|\} \leftarrow ComVR?\{|ComVR|\}, \\
& ComVR?\{|ComVR|\} \leftarrow FakeRV?\{|ComVR|\}]]
\end{aligned}
$$

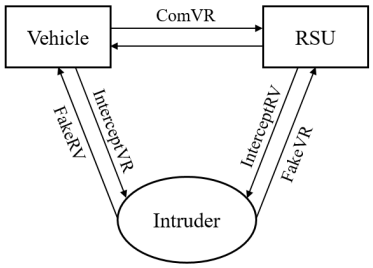

Fig. 5. Channels with an Intruder

$\{|c|\}$ is the symbol which denotes that the set of all communications over the channel c. The first two formulas represent that the process performs either a $ComVR!msg$ or $InterceptVR!msg$ event whenever $Vehicle_0$ performs a corresponding $ComVR!msg$ event. Similarly, the last two formulas imply that when the $ComVR!msg$ event occurs in $Vehicle_0$, the process $Vehicle$ behaves like $ComVR?msg$ or $FakeRV?msg$.

F. RSU Modeling

We formalize RSU process as below.

$$
\begin{aligned}
RSU_0 =_{df}\ & ComRC?msg_{prdc}.R.C.E(ca_{kpri}, cert_r) \rightarrow \\
& ComVR?msg_{reqb}.V.R.vid.t_{exp} \rightarrow Time?t \rightarrow \\
& \left(\begin{array}{l} \left(\begin{array}{l} ComVR!msg_{reqb}.R.V.rsu_{id}.t_s \\ \{msg_{cnt} := msg_{cnt} + 1\} \rightarrow dh\_key\_change \rightarrow \\ ComVR?msg_{data1}.V.R.E(k_{dh}, msg) \rightarrow \\ \left(\begin{array}{l} ComRRc!msg_{data2}.R.Rc. \\ E(r_{kpri}, msg).cert_{rsu}.t_v \rightarrow \\ ComRRc?msg_{ackrc}.Rc.R.ack.cert_{rc} \\ \{msg_{cnt} := msg_{cnt} - 1\} \rightarrow RSU_0 \end{array} \right) \\ \lhd (D(k, E(k_{dh}, msg))) \rhd \\ (fail\{msg_{cnt} := msg_{cnt} - 1\} \rightarrow RSU_0) \end{array} \right) \\ \lhd (msg_{cnt} < m_{threshold} \wedge t < t_{exp}) \rhd (fail \rightarrow RSU_0) \end{array} \right) \\
\Box\ & ComVR?msg_{reqs}.V.R.rsu_{id}.v_{id}.cert_v \rightarrow \\
& ComRC!msg_{reqv}.R.C.cert_v \rightarrow \\
& ComRC?msg_{rspv}.C.R.cert_v.verified \rightarrow \\
& \left(\begin{array}{l} \left(\begin{array}{l} ack := true \rightarrow \\ ComVR!msg_{ackr}.R.V.rsu_{id}.ack.cert_r \rightarrow RSU_0 \end{array} \right) \\ \lhd (verified == true) \rhd (fail \rightarrow RSU_0) \end{array} \right)
\end{aligned}
$$

msg_{cnt} is the current number of messages in this RSU. $m_{threshold}$ denotes the threshold value of message control and t_v indicates the validity time duration. For the RSU, in ad hoc or infrastructure domain, the actions on channels $ComVR$ and $ComRRc$ correspond to Steps 2-5 in Fig. 2. While, in in-vehicle communication, the actions on channels $ComVR$ and $ComRC$ correspond to Steps 1-4 in Fig. 3.

The corresponding process RSU is formalized using renaming which is similar to the process $Vehicle$, so we leave out details here.

G. Controller Modeling

Controllers are only used in ad hoc or infrastructure domain. The actions of the RSU controller on channels $ComRRc$ and $ComRcZc$ correspond to Steps 4-6 in Fig. 2 and the action of the Zone controller on channel $ComRcZc$ corresponds to Step 6 in Fig. 2. The models of $RSUController$ and $ZoneController$ are shown as below.

$$
\begin{aligned}
RC_0 =_{df}\ & ComRcC?msg_{prdc}.C.Rc.E(ca_{kpri}, cert_{rc}) \rightarrow \\
& ComRRc?msg_{data2}.R.Rc.E(r_{kpri}, msg).cert_{rsu}.t_v \rightarrow \\
& Time?t \rightarrow \\
& \left(\begin{array}{l} \left(\begin{array}{l} ComRRc!msg_{ackr}.Rc.R.ack.cert_{rc} \rightarrow \\ ComRcZc!msg_{data1}.Rc.Zc. \\ E(rc_{kpri}, E(r_{kpri}, msg)).t_r \rightarrow RC_0 \end{array} \right) \\ \lhd (t < t_v) \rhd (fail \rightarrow RC_0) \end{array} \right)
\end{aligned}
$$

$$
\begin{aligned}
ZC_0 =_{df}\ & ComZcC?msg_{prdc}.C.Zc.E(ca_{kpri}, cert_{zc}) \rightarrow \\
& ComRcZc?msg_{data1}.Rc.Zc. \\
& E(rc_{kpri}, E(r_{kpri}, msg)).t_r \rightarrow Time?t \rightarrow \\
& \left(\begin{array}{l} \left(\begin{array}{l} \left(\begin{array}{l} (success \rightarrow ZC_0) \\ \lhd (D(r_{kpub}, E(r_{kpri}, msg))) \rhd \\ (fail \rightarrow ZC_0) \end{array} \right) \\ \lhd (D(rc_{kpub}, E(rc_{kpri}, E(r_{kpri}, msg)))) \rhd \\ (fail \rightarrow ZC_0) \end{array} \right) \\ \lhd (t < t_r) \rhd (fail \rightarrow ZC_0) \end{array} \right)
\end{aligned}
$$

Similarly, the processes RC and ZC are built using renaming and due to the space limit, we omit the details here.

H. Intruder Modeling

We formalize the intruder as a process which can carry out attacks such as intercepting and faking messages from valid communications. Firstly, we have a set **Fact**, containing all facts an intruder can learn.

$$Fact =_{df} \{Entity,\ CA\} \cup Cert \cup T \cup \{I_{kpri},\ I_{kpub}\} \cup \{k, c | k \in Key, c \in Content\} \cup \{E(k,c) | k \in Key, c \in Content\}$$

Then we define how the intruder deduces new fact f from given fact set F. We use the symbol $F \mapsto f$ to represent that f can be deduced from the set F. The detailed definition is given as follows. The first two rules indicate encryption and decryption. The last rule means that if the intruder can deduce the fact f from the fact set F and F is the subset of F', the f can also be derived from F'.

$$\{k, c\} \mapsto E(k,c) \quad \{k^{-1}, E(k,c)\} \mapsto c$$
$$F \mapsto f \wedge F \subseteq F' \Rightarrow F' \mapsto f$$

We also introduce a function $Info(m)$, which implies the facts learned by the intruders if they intercept messages.

$$Info(msg_{reqv}.a.b.cert) =_{df} \{a, b, cert\}$$
$$Info(msg_{rspv}.a.b.state) =_{df} \{a, b, state\}$$
$$Info(msg_{prdc}.a.b.E(k,c)) =_{df} \{a, b, E(k,c)\}$$
$$Info(msg_{reqb}.a.b.id.t) =_{df} \{a, b, id, t\}$$
$$Info(msg_{d1}.a.b.E(k,c)) =_{df} \{a, b, E(k,c)\}$$
$$Info(msg_{d2}.a.b.E(k,c).cert.t) =_{df} \{a, b, E(k,c), cert, t\}$$
$$Info(msg_{reqs}.a.b.rid.vid.cert) =_{df} \{a, b, rid, vid, cert\}$$
$$Info(msg_{ackr}.a.b.rid.ack.cert) =_{df} \{a, b, rid, ack, cert\}$$
$$Info(msg_{ackrc}.a.b.ack.cert) =_{df} \{a, b, ack, cert\}$$

Then, we add a definition of $Deduce$ channel which is used to deduce new facts.

$$Channel\ Deduce : FACT.P(FACT)$$

Based on this, we give the formalization of $Intruder$ which is parameterized by the facts he knows.

$$Intruder(I) =_{df}$$
$$\Box_{msg \in MSG} Intercept.msg \rightarrow Intruder_0(I \cup Info(msg))$$
$$\Box\Box_{msg \in MSG \cap Info(msg) \in I} Fake.msg \rightarrow Intruder_0(I)$$
$$\Box\Box_{f \in Fact, f \notin F, F \mapsto f} Init\{data\_leakage\_success := false\}$$
$$\rightarrow Deduce.f.F \rightarrow \left(\begin{array}{l} \left(\begin{array}{l} data\_leakage\_success = true \\ \rightarrow Intruder_0(F \cup \{f\}) \end{array} \right) \\ \triangleleft (f == Data) \triangleright \\ \left(\begin{array}{l} data\_leakage\_success = false \\ \rightarrow Intruder_0(F \cup \{f\}) \end{array} \right) \end{array} \right)$$

If the intruder intercepts the message msg, then it adds $info(msg)$ into his facts. Also, if the intruder knows $info(msg)$, then he can pretend as a legal entity and fake the message msg. Further, as introduced before, the intruder can deduce new facts from the known facts as well.

For $Intruder_0$, we define its parameter as IK, which stands for the intruder's initial knowledge.

$$Intruder =_{df} Intruder_0(IK),\ IK =_{df} \{Entity, I_{kpub}, I_{kpri}\}$$

IV. VERIFICATION AND IMPROVEMENT

In this section, we use model checker PAT to verify the properties of the above constructed model. Moreover, we propose an improved model according to the verification results.

A. Properties Verification

The descriptions and the corresponding assertions of specific security properties are given below.

Property 1: Deadlock Freedom

Deadlock is a situation in which nothing further can happen. This property means that we need to ensure the model cannot get stuck in a deadlock state. In the tool PAT, we use a primitive to describe this situation.

$$\#assert\ VANET\ deadlockfree;$$

Property 2: Divergence Freedom

Divergence is a phenomenon in which a process has an infinite loop or unguarded recursion. To ensure that the model is well defined, we need to check if the model is divergence free. We complete the check by means of a primitive in PAT.

$$\#assert\ VANET\ divergencefree;$$

Property 3: Data Leakage

We also verify whether the intruder can obtain the message successfully since this property is relevant to the security of VANET. The assertion is set to check it.

$$\#define\ Data\_Leakage\_Success\ data\_leakage\_success == true;$$
$$\#assert\ VANET\ reaches\ Data\_Leakage\_Success;$$

Property 4: Vehicle Faking

This property means that the system is unaware that an intruder has succeeded in posing as a legal Vehicle successfully. The assertions are listed as follows, where $vehicle\_fake\_success1$ and $vehicle\_fake\_success2$ are boolean variables defined to verify whether the intruder succeeded in in-vehicle communication and in ad hoc or infrastructure domain respectively.

$$\#define\ Vehicle\_Fake\_Success\ vehicle\_fake\_success1 || vehicle\_fake\_success2 == true$$
$$\#assert\ VANET\ reaches\ Vehicle\_Fake\_Success;$$

Property 5: RSU Faking

Analogously, this property means that an intruder has disguised as a legal RSU successfully. $rsu\_fake\_success1$ and $rsu\_fake\_success2$ are used to check whether the intruder succeeded in in-vehicle communication and in ad hoc or infrastructure domain. The related assertions are shown as below.

$$\#define\ RSU\_Fake\_Success\ rsu\_fake\_success1 || rsu\_fake\_success2 == true;$$
$$\#assert\ VANET\ reaches\ RSU\_Fake\_Success;$$

Verification Results

The verification results are shown in Fig. 6.

- Property 1 and Property 2 are valid, indicating that our model can never run into a deadlock state and is well defined.
- Property 3 is valid. It means the intruder has acquired the message successfully, i.e., data security of the system is not guaranteed. Therefore, we put forward the improvement next.
- Property 4 and Property 5 are invalid, which represents that the intruder can never pretend as a legal vehicle or a legal RSU successfully.

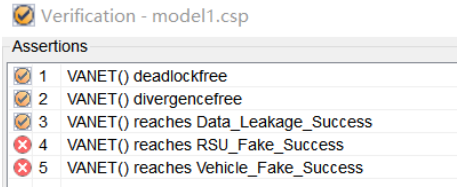

Fig. 6. Verification Results of the Model

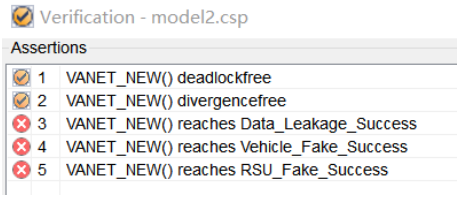

Fig. 7. Verification Results of the Improved Model

B. Attack and Improvement

As illustrated in the verification results above, Property 3 is valid. It indicates that the system still has security risk in spite of the usage of digital signature and encryption. After executing Deffie-Hellman key exchange with the RSU, the vehicle sends the message encrypted with the key shared with the RSU. Later, the RSU uses the shared key to decrypt the message. After decrypting, the RSU encrypts the decrypted message with its own private key. Then the RSU sends the encrypted message along with its certificate to the RSU controller. Once the intruder gets the RSU's public key, he can decrypt the message and obtain it. An example which causes data leakage is presented as below.

$$\begin{aligned} & ComVR.msg_{data1}.V.R.E(k_{dh}, msg) \\ \rightarrow\ & InterceptRRc.msg_{data2}.R.Rc.E(r_{kpri}, msg).cert_{rsu}.t_v \\ \rightarrow\ & FakeRcR.msg_{ackrc}.Rc.R.ack.cert_{rc} \end{aligned}$$

First, the vehicle sends the encrypted message to RSU through the channel $ComVR$. Then the intruder intercepts this message through the channel $InterceptRRc$. Since every entity in the system knows the public key of CA, the intruder can decrypt the certificate of the RSU with CA's public key. Also, as the certificate contains the entity's public key, the intruder can access the public key and obtain the message.

In order to overcome the above problem, we modify the architecture by encrypting the original encrypted message with the receiver's public key. Thus, we replace MSG_{data} defined before with the new following definition.

$$\begin{aligned} MSG_{data} =_{df} \{ & msg_{data1}.a.b.E(k1, E(k2, c)), \\ & msg_{data2}.a.b.E(k1, E(k2, c)).cert.t \mid \\ & a, b \in Entity, k1 \in PUBK, k2 \in PRIK, \\ & c \in Content, cert \in Cert, t \in T \} \end{aligned}$$

In this case, the intruder may intercept msg_{data} without the ability to decrypt it. Since the intruder cannot get the receiver's private key and it cannot decrypt the intercepted message consequently.

C. Improved Model and Verification

We formalize the following model to verify whether the improved model $VANET_{NEW}$ can solve data leakage problem based on the analyses above.

$$\begin{aligned} & VANET\_NEW =_{df} \\ & \quad VANET\_NEW1\ [|INTRUDER\_PATH|]\ Intruder \\ & VANET\_NEW1 =_{df} Vehicle||RSU||RC||ZC||CA||Clock \end{aligned}$$

The new verification results are shown in Fig. 7. Property 3 is invalid, which indicates that the intruder cannot get the message, i.e., data security is ensured in our new model.

V. CONCLUSION AND FUTUREWORK

Kumar et al. proposed a new architecture to confirm the security of VANET using end-to-end authentication and hierarchical structure [5]. In this paper, we have modeled this proposed architecture of VANET using CSP. With the aid of model checking tool PAT, we have verified five properties of this model, including *deadlock freedom, divergence freedom, data leakage, vehicle faking and RSU faking*. The verification results show that the proposed architecture may cause data leakage. Aiming to handle this problem, we improved the above proposed model by encrypting the messages with receiver's public key. The new verification results indicate that the improved model is truly secure. We will dive into more security issues over VANET and explore how to verify other security properties with formal methods in the future.

Acknowledgements. This work was partly supported by National Key Research and Development Program of China (grant no. 2018YFB2101300), National Natural Science Foundation of China (grant no. 61872145), Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (grant no. ZF1213), the Fundamental Research Funds for the Central Universities of China and the Opening Project of Shanghai Trusted Industrial Control Platform (grant no. TICPSH202003007-ZC).

REFERENCES

[1] Saif Al-Sultan, Moath M. Al-Doori, Ali H. Al-Bayatti, Hussein Zedan: A comprehensive survey on vehicular Ad Hoc network. J. Network and Computer Applications 37: 380-392 (2014)
[2] H. Hasrouny, C. Bassil, A.E. Samhat, A. Laouiti, "Security Risk Analysis of a Trust model for Secure Group Leader-based communication in VANET", Ad-hoc Networks for Smart Cities Book IWVSC Malaysia, pp. 71-83, 2016.
[3] H. Hasrouny, A.E. Samhat, C. Bassil, A. Laouiti, "VANET Security Challenges and Solutions: A Survey" in Vehicular Communications journal, Elsevier, vol. 7, pp. 7-20, January 2017.
[4] Hamssa Hasrouny, Abed Ellatif Samhat, Carole Bassil, Anis Laouiti: VANet security challenges and solutions: A survey. Vehicular Communications 7: 7-20 (2017)
[5] Gulshan Kumar, Rahul Saha, Mritunjay Kumar Rai, Tai-Hoon Kim: Multidimensional Security Provision for Secure Communication in Vehicular Ad Hoc Networks Using Hierarchical Structure and End-to-End Authentication. IEEE Access 6: 46558-46567 (2018)
[6] C. A. R. Hoare: Communicating Sequential Processes. Commun. ACM 21(8): 666-677 (1978)
[7] Gavin Lowe, A. W. Roscoe: Using CSP to Detect Errors in the TMN Protocol. IEEE Trans. Software Eng. 23(10): 659-669 (1997)
[8] Ho T. Dung, Thang H. Bui, Tho T. Quan: Model Checking Control Flow Petri Nets Using PAT. ICCSA (6) 2013: 124-129
[9] Zhipeng Shao, HanYong Hao, Yuanyuan Ma, Chen Wang, Jiaxuan Fei: Modeling and Verifying Intelligent Unit Transmission Protocol Using CSP Model Checker PAT. QRS Companion 2016: 244-251
[10] Whitfield Diffie, Martin E. Hellman: New directions in cryptography. IEEE Trans. Information Theory 22(6): 644-654 (1976)
[11] Victor S. Miller: Use of Elliptic Curves in Cryptography. CRYPTO 1985: 417-426