Formal Security Analysis of the Authentication Protocol in Smart Cities using AVISPA

Hyewon Park and Yohan Park

School of Computer Engineering, Keimyung University, Daegu, Republic of Korea
wldnjsfuf@stu.kmu.ac.kr
yhpark@kmu.ac.kr

Abstract. Smart cities optimize traffic management and vehicle communication through Intelligent Transportation Systems (ITS), with Vehicular Ad-hoc Networks (VANET) serving as a core infrastructure. In these environments, security vulnerabilities can severely impact the smart city traffic system, leading to traffic congestion, blockage of emergency vehicle routes, and disruption of autonomous driving systems. Unfortunately, identifying security vulnerabilities of the system in VANET is complex due to various attack types and the dynamic nature of the network, requiring systematic verification techniques for effective analysis. Recently, Nath et al. proposed an authentication protocol for VANETs using LWE-based lattice signatures and tokens, however the protocol has not been sufficiently validated. This study utilizes AVISPA (Automated Validation of Internet Security Protocols and Applications) to analyze Nath et al.'s protocol. AVISPA, an automated security verification tool based on the Dolev-Yao(DY) attacker model, is effective in assessing various threats such as replay attacks and man-in-the-middle attacks, making it ideal for evaluating security in the VANET environment. The security analysis reveals that the protocol is vulnerable to multiple attacks due to the lack of message freshness verification and user authentication. To address these vulnerabilities, we propose countermeasures to enhance message freshness verification and user authentication mechanisms, and validate the improved protocol's security through AVISPA simulation. Finally, we verify the security of the authentication scheme which is applied the countermeasures through AVISPA. The result shows that the security of smart city vehicular networks can be strenghtened through AVISPA-based security verification and this can provide valuable insights for desining security protocols in smart cities.

Keywords: Intelligent Transportation System · Vehicular Ad-hoc Networks · Lattice-based cryptography · security protocol analysis · AVISPA simulation · sybil attack · eclipse attack

1 Introduction

Smart cities achieve efficient urban operations through Intelligent Transportation Systems (ITS), which optimize traffic management and facilitate seamless

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

vehicle communication[6]. Vehicular Ad-hoc Networks (VANET) play a critical role in the transportation infrastructure of smart cities, enabling real-time data exchange between vehicles (V2V) and between vehicles and infrastructure (V2I). This allows for the optimization of traffic flow, accident prevention, and the enhancement of emergency response systems[14]. Furthermore, VANET is also employed in various user-centric services such as road traffic load balancing, 3D navigation, and in-vehicle entertainment services[9].

However, if security vulnerabilities exist in VANET, a smart city's traffic management system could be significantly affected by malicious actors. For instance, replay attack or sybil attack[4] could disseminate false traffic information, leading to large-scale congestion or malfunctioning emergency vehicle priority systems. Additionally, an eclipse attack[8] that isolates a specific vehicle from the network could mislead autonomous vehicles into making incorrect decisions, potentially resulting in severe accidents. Thus, ensuring the security of VANET in smart cities is not only essential for traffic management but also crucial for maintaining overall urban stability and trust. To address these security challenges, robust authentication mechanisms are required, and it is critical to assess whether existing approaches provide sufficient security in VANET environments.

The previously proposed authentication methods in the VANET environment provide security using cryptographic algorithms based on mathematical challenges such as the discrete logarithm problem or integer factorization, like Rivest-Shamir-Adleman (RSA) and Elliptic-Curve Cryptography (ECC)[5, 18, 21]. However, with advancements in quantum algorithms, their vulnerabilities have become increasingly apparent[7, 19]. To address this issue, Nath et al. (2024) [15] proposed an authentication scheme that integrates a lattice-based signature mechanism based on the Learning With Errors (LWE) problem and a token-based authentication approach. This scheme claims to enhance anonymity using pseudo-identities and to enable fast authentication via a token-based mechanism. However, our analysis reveals that this approach is vulnerable to various attacks, including replay attack, impersonation attack, insider attack, table leakage attack, denial-of-service (DoS) attack, eclipse attack, and sybil attack.

In this paper, we review the lattice-based authentication protocol proposed by Nath et al. and conduct a formal simulation analysis using the Automated Validation of Internet Security Protocols and Applications (AVISPA)[1] version 1.6 tool. This formal analysis method is effective for evaluating protocol security [10, 11, 16, 17, 20]. Finally, we demonstrate that this scheme is not suitable for real-world VANET environments and provide recommendations to address its vulnerabilities and enhance security.

2 Preliminaries

2.1 System model

In a VANET environment, authentication and message exchange involve trusted entities such as the Trusted Authority (TA), Road Side Unit (RSU), and Onboard Unit (OBU). The TA oversees information exchange and ensures secure

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

communication over wireless media after vehicle registration. The RSU serves as infrastructure, managing vehicle authentication and parameters, while the OBU utilizes these functions to communicate with RSUs or other vehicles. In a high-speed vehicular environment, authentication must occur with minimal delay, necessitating advanced security designs. The overall system model is shown in Figure 1.

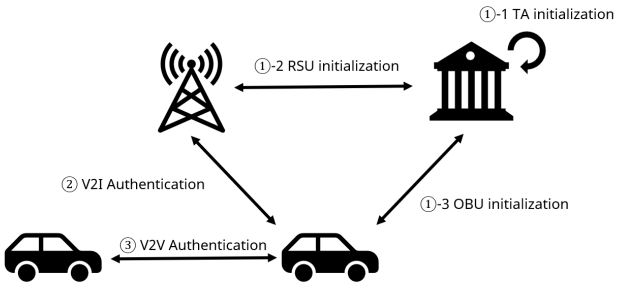

Fig. 1: System model

2.2 Threat Model

Nath et al. conducted a security analysis using the Dolev-Yao (DY) [3] and Canetti-Krawczyk (CK) [2] attacker models. The attackers in the DY and CK models possess the following capabilities:

- The attacker can eavesdrop, intercept, modify, delete, and manipulate messages transmitted over public channels.
- The attacker may guess the legitimate user's identity or password but cannot accurately guess both simultaneously.
- The attacker can compromise security by launching impersonation attack, replay attack, and man-in-the-middle attack.
- These models assume perfect cryptographic primitives, meaning encrypted messages are infeasible to decrypt without the corresponding key.
- The attacker can leak certain secret information, such as temporary nonces and session keys.

2.3 Lattice-Based Cryptography

Lattice-based cryptography provides strong resistance against quantum attacks due to the computational hardness of solving problems like the Shortest Vector Problem (SVP) and Learning With Errors (LWE) [12]. These problems are

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

computationally hard even for quantum computers, making lattice-based cryptography a promising alternative to classical public-key systems. A widely used encryption method relies on the LWE problem, where messages are encoded using a random matrix A, a secret key sek, and an error term e [13].The encryption process follows:

$$c = (A, \lfloor A \cdot sek + e + \text{encode}(m) \rceil) . \tag{1}$$

For decryption:

$$m = \text{decode}(\lfloor g \rceil - A \cdot sek). \tag{2}$$

Here, $\lfloor . \rceil$ denotes a rounding function that maps a point $a \in \mathbb{R}^n$ onto a lattice code $(\wedge, \lfloor . \rceil_\wedge)$:

$$a = \lfloor a \rceil_\Lambda + [a]_\Lambda, \tag{3}$$

where $\lfloor a \rceil_\Lambda \in \wedge$ is the quantized lattice point and $[a]_\Lambda \in \mathcal{V}_\wedge$ represents the rounding error. This transformation ensures efficient message encoding with minimal decoding errors.

2.4 AVISPA

To formally verify the proposed protocol, we utilize AVISPA[1], a widely used tool for security assessment. AVISPA is a well-known simulation tool for verifying the security of internet protocols and applications, particularly effective in evaluating vulnerabilities such as replay attack and man-in-the-middle attack. This tool employs code written in the High-Level Protocol Specification Language (HLPSL) and utilizes four backend models: Constraint-Logic-Based Attack Searcher (CL-AtSE), On-the-Fly Model Checker (OFMC), SAT-Based Model Checker (SATMC), and Tree-Automata-Based Protocol Analyzer (TA4SP). To assess the security properties of the protocol, the HLPSL code is first converted into an intermediate format using the HLPSL2IF translator, after which the backend models perform the verification.

3 Review of Nath et al.'s Scheme

This section provides an overview of the model proposed by Nath et al. The scheme is divided into three processes: the initialization phase, authentication in V2I (authentication for token issuance), and authentication in V2V and V2I(authentication for communication).

3.1 System initialization

This section outlines the initialization process for each entity involved in the system. Before mutual authentication and message exchange, every entity must undergo an initialization phase. Once a vehicle completes the registration process with the TA, the TA generates and periodically updates a Reg_{PID} table containing the vehicle's master public key and pseudo-ID. This table is maintained in the RSU. By maintaining this table, the need for digital certificate exchange between the RSU and OBU is eliminated. The details are as follows.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

TA initialization The TA, equipped with high computational capabilities, selects and distributes the necessary parameters for secure key exchange among the entities. First, it selects two distinct matrices, $A \in \mathbb{Z}_q^{m \times n}$ and $A' \in \mathbb{Z}_q^{m \times n}$. Then, it chooses a secret key sek from the distribution $\mathbb{Z}_q^{x^m}$ and computes the public key $Pub = A'.sek$. Subsequently, the TA selects a lattice code pair $X, (\lfloor . \rceil)_X$ for error correction and another lattice code pair $Y, (\lfloor . \rceil)_Y$ for quantization. Next, it selects two hash functions: $H_1 : \mathbb{Z}_q^m \rightarrow \mathbb{Z}_q$ and $H_2 : 0, 1^{\rightarrow} \mathbb{Z}_q$. Finally, it sets the public parameters as $m, n, q, A, Pub, H_1, H_2$.

RSU initialization The r-th RSU is identified by its unique identity, RSU_r. Under the supervision of the TA, the RSU selects its secret key $rsek_r$ from $x^m \in \mathbb{Z}_q$ and computes its public key as $rpub_r = A'.rsek_r$. The RSU_r receives from the TA the identities and public keys of nearby RSUs (e.g., RSU_s, RSU_t), along with the lattice code pairs $X, (\lfloor . \rceil)_X$, $Y, (\lfloor . \rceil)_Y$, and the hash functions H_1 and H_2. Finally, the RSU stores its key pair $(rsek_r, rpub_r)$, the error term $(err \rightarrow x^n)$, and the lattice codes $X, (\lfloor . \rceil)_X$, $Y, (\lfloor . \rceil)_Y$ in inactive memory. The public parameters are then set as $\{RSU_r, A, rpub_r, H_1, H_2\}$.

OBU initialization The i-th vehicle, or OBU_i, must undergo a registration process through the TA. The OBU_i receives the following parameters from the TA: the real identity $real_i$, the master secret key $msek_{vi}$, the master public key $mpub_{vi} = A.msek_{vi}$, the secret key sek_{vi}, the public key $pub_{vi} = A'.sek_{vi}$, the pseudo-identity $PID_i = \langle PID_{i1}, PID_{i2} \rangle$, where $PID_{i1} = H_1(real_i)$ and $PID_{i2} = real_i \oplus H_1(sek)$, and the list of active RSUs within a specific region, $List_{RSU}$. Next, the OBU_i stores the hash functions H_1, H_2 and the lattice codes $X, (\lfloor . \rceil)_X$, $Y, (\lfloor . \rceil)Y$ in non-volatile memory, and sets $\{PID_i, A, pub_{vi}, H_1, H_2\}$ as the public parameters.

3.2 Authentication in V2I

Before communicating with other vehicles, a fully registered OBU must receive a token from the RSU. The authentication and token exchange process between the RSU and OBU is illustrated in Figure 2, and the detailed procedure is as follows.

First, the j-th RSU, denoted as RSU_j, broadcasts a beacon message containing its identity RSU_j and public key $rpub_j$. Upon receiving the broadcasted message, OBU_i verifies whether RSU_j is listed in $List_{RSU}$. If the verification is successful, OBU_i generates a timestamp t_i and constructs a message $M1 = \{PID_i, t_i, RSU_j\}$, where PID_i is its pseudo-identity, and sends it as a response.

Next, RSU_j checks whether PID_i exists in the Reg_{PID} table. If a match is found, it retrieves the master public key of the corresponding OBU_i, denoted as $mpub_{vi}$, and computes $h_0 = H_1(mpub_{vi})$. Then, using the identities and public keys of neighboring RSUs aligned in a straight path, it generates the token as $\rho = H_2(RSU_j || rpub_j || RSU_k || rpub_k || RSU_l || rpub_l || \Delta t)$.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

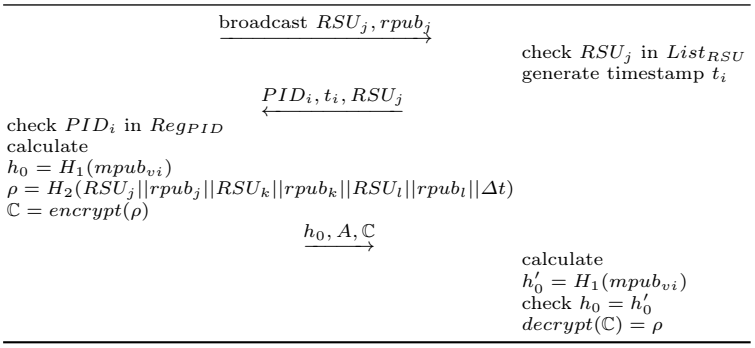

Fig. 2: V2I Authentication phase of Nath et al.'s scheme.

The RSU then encrypts the generated token using the master public key of OBU_i, computing $\mathbb{C} = encrypt(\rho)$, and transmits $\{h_0, A, \mathbb{C}\}$ to OBU_i.

Upon receiving $\{h_0, A, \mathbb{C}\}$, OBU_i computes $h_0' = H_1(mpub_{vi})$ using its own master public key and verifies whether $h_0 = h_0'$. If the verification succeeds, OBU_i decrypts $\mathbb{C}$ using its master secret key to obtain $\rho = decrypt(\mathbb{C})$.

3.3 Authentication in V2V and V2I

Once OBU_j obtains the token ρ from a nearby RSU, it becomes eligible to communicate with other vehicles. To initiate communication with another vehicle, OBU_j follows the process outlined below.

First, OBU_j computes $\gamma_j = mesk_{vj}.\rho$ and generates a timestamp t_j. It then performs a hash operation on the message msg, ρ, and t_j to compute $\beta_j = H_2(msg||\rho||t_j||PID_{j1})$. Subsequently, it generates the signature as $sign_j = \beta_j + \gamma_j$.

Next, OBU_j constructs the message containing the destination information $dest$ and the message $MSG = \{msg, PID_j, sign_j, \beta_j, mpub_{vj}, t_j\}$. This message is then broadcasted over the wireless communication medium to be received by OBU_k.

Upon receiving $\{dest, MSG\}$, OBU_k verifies the timestamp t_j and computes $\beta_k = H_2(msg||\rho||t_j||PID_{j1})$ using the received information. It then checks whether $\beta_k = \beta_j$. If the values match, OBU_k verifies OBU_j's signature using the equation $A.sign_j = A.\beta_k + mpub_{vj}.\rho$. If the signature verification is successful, OBU_k considers the message from OBU_j to be valid. Otherwise, it discards the message.

This authentication process is not limited to a single vehicle. It is performed dynamically during V2V communication, allowing multiple vehicles to exchange messages simultaneously. Additionally, vehicles can transmit messages to the RSU, which can verify the authenticity of these messages using the previously described verification process.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

4 Security weakness of Nath et al.'s Scheme

This section demonstrates that Nath et al.'s scheme is vulnerable to replay attack, impersonation attack, insider attack, table leakage attack, DoS attack, eclipse attack, and sybil attack. The vulnerabilities are analyzed using both formal and informal methods.

4.1 Formal Analysis Using AVISPA tool

HLPLS Specifications To verify the security of the protocol, HLPSL code was written for each entity involved in the communication process. The code is categorized based on the roles of the entities: TA (Figure 3 (a)), RSU (Figure 3 (b)), and OBU (Figures 3 (c)). Finally, the session and environment details are shown in Figure 3 (d).

The registration process for RSUs and OBUs corresponds to transitions 1 and 2 in the code. To ensure that the registered public keys can be effectively utilized in the protocol, each entity (TA, RSU, and OBU) receives its public key as a parameter. The public keys of TA, RSU, and OBU are denoted as PK_t, PK_r, PK_i, respectively, while the master public key of OBU is represented as MPK_i. In transition 1 of the RSU, $SND(RSU_j, PK_r)$ signifies the stage in the V2I authentication process where the RSU broadcasts its information. After receiving this message, the OBU sends an authentication request message PID_i, t_i, RSU_j to the RSU, which corresponds to $SND(Ti', PIDi1, PIDi2, RSUj')$ in transition 3. Upon receiving this message, the RSU generates a token, encrypts it, and transmits it to the OBU. This process occurs in transition 4, and the transmission to the OBU is represented as $SND(H1(MPK_i), Token', Token_{MPK_i})$. Finally, the process in which the OBU receives the encrypted message is described in transition 5. To evaluate the protocol's security, authentication is performed using witness() and request() functions during the message transmission and reception process. The protocol identifiers $auth_1, auth_2$, and $auth_3$ are used in conjunction with witness() and request() to verify whether the authentication process is correctly executed. Lastly, the code defined in the Goal section is used to verify that the protocol's overall authentication process is properly conducted.

Simulation Results The results of executing the AVISPA simulation on the previously defined HLPSL code are shown in Figure 4. The simulation results confirm that the scheme proposed by Nath et al. is not secure against replay attacks and man-in-the-middle (MITM) attacks. The specific points where issues arise can be identified in the Goal section. This analysis allows for the identification of authentication vulnerabilities within the protocol and provides valuable insights for developing security enhancements.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

```
role
role_TA(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_RSU:(symmetric_key) set,Key_set_TA_OBU:(symmetric_key) set,SND,RCV:channel(dy))
played_by TA
def=
  local

State:nat,Reali:text,PIDi1:text,A:text,NN:text,MM:text,QQ:text,PIDi2:text,Key_2:symmetric_key,Key_1:symmetric_key
  init
    State := 0
  transition
    1. State=0 /\ RCV(start) =|> State':=1
    /\ A':=new() /\ QQ':=new() /\ NN':=new() /\ MM':=new()
    /\ Key_1':=new() /\
Key_set_TA_RSU':=cons(Key_1',Key_set_TA_RSU)
    /\ SND({MM'.NN'.QQ'.A'.PKt}_Key_1')

    /\ Reali':=new() /\ PIDi2':=new() /\ PIDi1':=new() /\ Key_2':=new()
    /\ Key_set_TA_OBU':=cons(Key_2',Key_set_TA_OBU)
    /\ SND({MM'.NN'.QQ'.A'.PKt.MPKi.PKi.PIDi1'.PIDi2'.Reali'}_Key_2')
end role
```

(a) TA's role.

```
role
role_RSU(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_RSU:(symmetric_key) set,SND,RCV:channel(dy))
played_by RSU
def=
  local

State:nat,QQ:text,MM:text,NN:text,A:text,RSUj:text,PIDi1:text,Ti:text,PIDi2:text,Token:text,H:hash_func,Key_1:symmetric_key
  init
    State := 0
  transition
    1. State=0 /\ in(Key_1',Key_set_TA_RSU)
    /\ RCV({MM'.NN'.QQ'.A'.PKt}_Key_1') =|> State':=1
    /\ Key_set_TA_RSU':=delete(Key_1',Key_set_TA_RSU)

    /\ SND(RSUj.PKr)
    /\ witness(RSU,OBU,auth_1,RSUj)

    4. State=1 /\ RCV(Ti'.PIDi1'.PIDi2'.RSUj) =|> State':=2
    /\ request(RSU,OBU,auth_2,PIDi1')
    /\ Token':=H(RSUj.PKr.Ti')
    /\ SND(H(MPKi).Token'.{Token'}_MPKi)
    /\ witness(RSU,OBU,auth_3,Token')
end role
```

(b) RSU's role.

```
role
role_OBU(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_OBU:(symmetric_key) set,SND,RCV:channel(dy))
played_by OBU
def=
  local

State:nat,Reali:text,A:text,NN:text,MM:text,QQ:text,RSUj:text,PIDi1:text,Ti:text,PIDi2:text,Token:text,H:hash_func,Key_1:symmetric_key
  init
    State := 0
  transition
    2. State=0 /\ in(Key_1',Key_set_TA_OBU)
    /\ RCV({{MM'.NN'.QQ'.A'.PKt.MPKi.PKi.PIDi1'.PIDi2'.Reali'}_Key_1')
=|> State':=1
    /\ Key_set_TA_OBU':=delete(Key_1',Key_set_TA_OBU)

    3. State=1 /\ RCV(RSUj'.PKr) =|> State':=2
    /\ request(OBU,RSU,auth_1,RSUj)
    /\ Ti':=new()
    /\ SND(Ti'.PIDi1.PIDi2.RSUj')
    /\ witness(OBU,RSU,auth_2,PIDi1)

    5. State=2 /\ RCV(H(MPKi).Token'.{Token'}_MPKi) =|> State':=3
    /\ request(OBU,RSU,auth_3,Token')
end role
```

(c) OBU's role.

```
role
session(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_RSU:(symmetric_key) set,Key_set_TA_OBU:(symmetric_key) set)
def=
  local
    SND3,RCV3,SND2,RCV2,SND1,RCV1:channel(dy)
  composition

role_OBU(TA,RSU,OBU,PKt,MPKi,PKi,PKr,Key_set_TA_OBU,SND3,RCV3) /\
role_RSU(TA,RSU,OBU,PKt,MPKi,PKi,PKr,Key_set_TA_RSU,SND2,RCV2) /\
role_TA(TA,RSU,OBU,PKt,MPKi,PKi,PKr,Key_set_TA_RSU,Key_set_TA_OBU,SND1,RCV1)
end role


role environment()
def=
  const

pki:public_key,pkt:public_key,trust:agent,hash_0:hash_func,obu:agent,rsu:agent,mpki:public_key,pkr:public_key,auth_1:protocol_id,auth_2:protocol_id,auth_3:protocol_id
  intruder_knowledge = {trust,rsu,obu,pkt,mpki,pki,pkr}
  composition
    session(trust,rsu,obu,pkt,mpki,pki,pkr,{},{})
end role

goal
  authentication_on auth_1
  authentication_on auth_2
  authentication_on auth_3
end goal

environment()
```

(d) Session, Environment, and Goal.

Fig. 3: HLPSL code

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

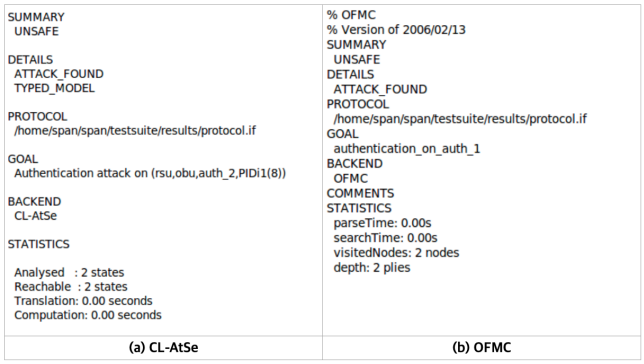

Fig. 4: AVISPA execution results

4.2 Informal analysis

This section demonstrates, through logical security analysis, that the scheme proposed by Nath et al. is vulnerable to replay attack, impersonation attack, insider attack, table leakage attack, DoS attack, eclipse attack, and sybil attack.

Replay attack

1. An attacker may capture and replay broadcast messages originating from RSU_j. OBU_i cannot verify the generation time of the broadcast message or authenticate its sender. As a result, the attacker can impersonate RSU_j through a replay attack.
2. An attacker can record the message M_1 sent by OBU_i, modify only the timestamp, and replay it. RSU_j does not verify the sender of message M_1. Consequently, the attacker can impersonate OBU_i using a replay attack.

Impersonation attack

1. An attacker can generate a broadcast message for RSU_j using publicly available information, such as RSU_j and $rpub_j$. OBU_i cannot verify the sender of the broadcast message, making it possible for the attacker to impersonate RSU_j.
2. An attacker can generate message M_1 using publicly available information, such as PID_i and RSU_j. RSU_j is unable to distinguish the actual sender of message M_1, allowing the attacker to impersonate OBU_i.

Insider attack A malicious user who has completed the legitimate registration process can obtain $mpub_{vi}$ through V2V communication. Using the acquired master public key of another vehicle, the attacker can generate $h_0, A, \mathbb{C}$ and send them to the OBU_i. Since OBU_i cannot verify the sender of the received message, the attacker can impersonate RSU_j.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

Table leakage attack An attacker can compromise the stored table of RSU_j and obtain critical parameters. Using the extracted values, including $mpub_{vi}$, RSU_j, and $rpub_j$, the attacker can generate $h_0, A, \mathbb{C}$ and send them to OBU_i. Since OBU_i cannot verify the sender of the received message, the attacker can impersonate RSU_j.

DoS attack

1. An attacker can record and replay the broadcast message from RSU_j or generate a new broadcast message using publicly available information, then repeatedly send it to OBU_i. Since OBU_i does not perform message verification, it is vulnerable to DoS attack.
2. An attacker can record and replay message M_1 sent by OBU_i or generate a new one, then repeatedly send it to RSU_j. Since RSU_j does not verify incoming messages, it is susceptible to DoS attack.

Eclipse attack An attacker capable of impersonating RSU_j can generate and transmit a manipulated token ρ to OBU_i. Since OBU_i cannot detect the alteration, it proceeds with V2V communication using the compromised token. As a result, OBU_i with the manipulated token is unable to communicate with legitimate vehicles and becomes isolated within the network. In smart city environments, this type of attack could have severe consequences by isolating emergency vehicles, preventing them from receiving critical updates on optimal routes, or delaying first responders. This could directly impact urban safety and traffic management efficiency.

Sybil attack

1. An OBU that has received a manipulated token from an attacker impersonating RSU_j may also receive a V2V message $dest, MSG$ from a fake vehicle created by the attacker. In this case, the OBU authenticates the fake vehicle's message using the compromised token. Since OBU_i cannot distinguish between legitimate and fake vehicles, it accepts $dest, MSG$ from the fake vehicle as valid information.
2. A malicious user who has completed legitimate registration can utilize a token obtained through the V2I process along with fabricated vehicle information to generate $dest, MSG$ and send it to OBU_i. Since OBU_i verifies messages solely based on the received information and token, it cannot determine whether the sender is a real or fake vehicle. Consequently, OBU_i accepts $dest, MSG$ from the fake vehicle as legitimate.

In large-scale smart cities, a sybil attack can significantly disrupt intelligent traffic management systems by allowing attackers to manipulate traffic data, leading to artificial congestion, inefficient routing, and delays in automated traffic flow optimization.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

5 Security Fixes

The scheme proposed by Nath et al. demonstrates significant security vulnerabilities primarily due to the lack of message freshness verification and absence of user authentication mechanisms. As a result of these weaknesses, an attacker can ultimately isolate users from the legitimate network and force them to accept only manipulated information. These critical security flaws are discussed in detail in Section 4.

Nath et al. claimed that their scheme enables fast and secure communication using tokens. However, the absence of message freshness checks, and the lack of a user authentication mechanism leaves the entire network highly vulnerable. Therefore, it is essential to design a method that ensures message freshness and incorporates a user authentication process to enhance network security.

To address the security issues identified in the protocol by Nath et al., we propose the following key guidelines.

Solution 1. According to Section 3.2, RSU_j transmits only its identity and public key during broadcasting. To prevent RSU_j impersonation attack, a timestamp and a signature mechanism can mitigate various attack. The detailed process is as follows.

RSU_j generates a timestamp t_j and signs it using its secret key $rsek_j$ to create the signature value $tsign_j = t_j + rsek_j$. Then, RSU_j constructs the broadcast message as $\{RSU_j, rpub_j, tsign_j, t_j\}$.

Upon receiving the broadcasted message, OBU_i first checks the freshness of the message using t_j. It then verifies the signature by computing $A.tsign_j = A.t_j + rpub_j$. Through this process, OBU_i can defend against replay attack, RSU impersonation attack, and DoS attack. The generated signature value remains valid for a certain period during RSU broadcasting and is updated when the token is refreshed.

Solution 2. In the process described in Section 3.2, when OBU_i sends message M_1, the sender cannot be authenticated, and RSU_j does not verify the timestamp upon receiving M_1. As a result, the freshness of the message cannot be ensured. To address this issue, a timestamp signing mechanism similar to the previous solution can be incorporated. The detailed method is as follows.

OBU_i selects a timestamp t_i and applies a digital signature using its master secret key $msek_{vi}$, producing $tsign_i = t_i + msek_{vi}$. It then constructs the message $M_1 = \{PID_i, t_i, tsign_i, RSU_j\}$ and transmits it to RSU_j.

Upon receiving the message, RSU_j verifies the freshness of the message by checking the timestamp t_i and then validates the signature by computing $A.tsign_i = A.t_i + mpub_{vi}$. Through this process, RSU_j can resist replay attack, OBU impersonation attack, and DoS attack.

Solution 3. In Section 3.2, RSU_j generates the message $\{h_0, A, C\}$ and sends it to OBU_i. Upon receiving this message, OBU_i decrypts it to obtain the token. However, there is no procedure to verify the freshness of the message, and there is a lack of validation to ensure that RSU_j is the actual sender. This results

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

in vulnerabilities to insider attack and table leakage attack. Additionally, since there is no mechanism to confirm that the sender is a legitimate RSU, there is a risk that OBU_i might receive a manipulated token without detecting it. As a result, the network becomes highly susceptible to severe security threats such as sybil attack and eclipse attack. To mitigate this risk, a procedure for OBU_i to verify RSU_j is necessary. The process is as follows:

First, RSU_j generates the token ρ, then encrypts it using OBU_i's master public key $mpub_{vi}$ to produce $\mathbb{C} = encrypt(\rho)$. Next, RSU_j generates a timestamp t_{j2} and calculates the signature for the token as $rsign_j = H_1(\rho||t_{j2}) + rsek_j$. Then, RSU_j sends $\{A, \mathbb{C}, rsign_j, t_{j2}\}$ to OBU_i.

Upon receiving the message, OBU_i first verifies the freshness of the message using the timestamp t_{j2}. Then, using its master secret key $msek_{vi}$, OBU_i decrypts $\mathbb{C}$ to obtain the token ρ. Afterward, to confirm that the token indeed came from RSU_j, the signature is verified using $A.rsign_j = A.H_1(\rho||t_{j2}) + rpub_j$. If the signature verification is successful, OBU_i can be confident that the token it received is the latest token sent by RSU_j, thus ensuring resistance against Sybil and Eclipse attacks.

Figure 5 presents the HLPSL code incorporating the proposed solutions. *Solution* 1 is applied in Transition 1 of the RSU, where an additional component, $\{Tj1'\}\_inv(PKr)$, is included in the broadcast message. Unlike the original protocol, this addition enables the OBU to verify that the broadcast message was indeed generated by the RSU. *Solution* 2 is implemented in Transition 3 of the OBU. In contrast to the original protocol, the message sent to the RSU now includes $\{Ti'\}\_inv(PKi)$. This modification allows the RSU to confirm that the message was generated by the OBU. *Solution* 3 is applied in Transition 4 of the RSU. Unlike the original approach, this solution utilizes a timestamp and a token to generate the signature $\{H(Tj2'.Token')\}\_inv(PKr)$ before transmission. This ensures that the OBU can verify that both the message and the token originated from the RSU. Figure 6 presents the AVISPA simulation results for the HLPSL code incorporating these solutions. The results confirm that the enhanced protocol is resistant to replay attacks and man-in-the-middle attacks.

6 Conclusion and future works

This study analyzes the security of VANET, a core infrastructure in smart city traffic systems. To evaluate the security of the lattice-based authentication protocol proposed by Nath et al., both formal analysis using AVISPA and informal logical analysis were conducted.

The analysis revealed that the protocol is vulnerable to various security threats, including replay attacks, impersonation attacks, and insider attacks, due to the lack of message freshness verification and absence of user authentication. In particular, through AVISPA verification, the study analyzed and validated how these vulnerabilities could be exploited in real attack scenarios.

To address these issues, this study proposes security improvements that strengthen message freshness verification and user authentication to ensure se-

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

```
role
role_TA(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_RSU:(symmetric_key) set,Key_set_TA_OBU:(symmetric_key) set,SND,RCV:channel(dy))
played_by TA
def=
        local

State:nat,Reali:text,PIDi1:text,A:text,NN:text,MM:text,QQ:text,PIDi2:text,Key_2:symmetric_key,Key_1:symmetric_key
init
    State := 0
transition
    1. State=0 /\ RCV(start) =|> State':=1
    /\ A':=new() /\ QQ':=new() /\ NN':=new() /\ MM':=new()
    /\ Key_1':=new() /\
Key_set_TA_RSU':=cons(Key_1',Key_set_TA_RSU)
    /\ SND({MM'.NN'.QQ'.A'.PKt}_Key_1')

    /\ Reali':=new() /\ PIDi2':=new() /\ PIDi1':=new() /\ Key_2':=new()
    /\ Key_set_TA_OBU':=cons(Key_2',Key_set_TA_OBU)
    /\ SND({MM'.NN'.QQ'.A'.PKt.MPKi.PKi.PIDi1'.PIDi2'.Reali'}_Key_2')
end role
```

(a) TA's role.

```
role
role_RSU(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_RSU:(symmetric_key) set,SND,RCV:channel(dy))
played_by RSU
def=
        local

State:nat,QQ:text,MM:text,NN:text,A:text,Tj2,Tj1,RSUj:text,PIDi1:text,Ti:text,PIDi2:text,Token:text,H:hash_func,Key_1:symmetric_key
        init
    State := 0
        transition
    1. State=0 /\ in(Key_1',Key_set_TA_RSU)
    /\ RCV({MM'.NN'.QQ'.A'.PKt}_Key_1') =|> State':=1
    /\ Key_set_TA_RSU':=delete(Key_1',Key_set_TA_RSU)

    /\ Tj1':=new()
    /\ SND(Tj1'.RSUj.PKr.{Tj1'}_inv(PKr))
    /\ witness(RSU,OBU,auth_1,Tj1')

    4. State=1 /\ RCV(Ti'.PIDi1.PIDi2.RSUj.{Ti'}_inv(PKi)) =|> State':=2
    /\ request(RSU,OBU,auth_2,Ti')
    /\ Tj2':=new()
    /\ Token':=H(RSUj.PKr.Ti')
    /\ SND({H(Tj2'.Token')}_inv(PKr).Tj2'.{Token'}_MPKi)
    /\ witness(RSU,OBU,auth_3,Token')
end role
```

(b) RSU's role.

```
role
role_OBU(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_OBU:(symmetric_key) set,SND,RCV:channel(dy))
played_by OBU
def=
        local

State:nat,Reali:text,A:text,NN:text,Tj2,Tj1,MM:text,QQ:text,RSUj:text,PIDi1:text,Ti:text,PIDi2:text,Token:text,H:hash_func,Key_1:symmetric_key
        init
    State := 0
        transition
    2. State=0 /\ in(Key_1',Key_set_TA_OBU)
    /\ RCV({MM'.NN'.QQ'.A'.PKt.MPKi.PKi.PIDi1'.PIDi2'.Reali'}_Key_1')
=|> State':=1
    /\ Key_set_TA_OBU':=delete(Key_1',Key_set_TA_OBU)

    3. State=1 /\ RCV(Tj1'.RSUj.PKr.{Tj1'}_inv(PKr)) =|> State':=2
    /\ request(OBU,RSU,auth_1,Tj1')
    /\ Ti':=new()
    /\ SND(Ti'.PIDi1.PIDi2.RSUj.{Ti'}_inv(PKi))
    /\ witness(OBU,RSU,auth_2,Ti')

    5. State=2 /\ RCV({H(Tj2'.Token')}_inv(PKr).Tj2'.{Token'}_MPKi) =|
> State':=3
    /\ request(OBU,RSU,auth_3,Token')
end role
```

(c) OBU's role.

```
role
session(TA:agent,RSU:agent,OBU:agent,PKt:public_key,MPKi:public_key,PKi:public_key,PKr:public_key,Key_set_TA_RSU:(symmetric_key) set,Key_set_TA_OBU:(symmetric_key) set)
def=
        local
    SND3,RCV3,SND2,RCV2,SND1,RCV1:channel(dy)
        composition

role_OBU(TA,RSU,OBU,PKt,MPKi,PKi,PKr,Key_set_TA_OBU,SND3,RCV3) /\
role_RSU(TA,RSU,OBU,PKt,MPKi,PKi,PKr,Key_set_TA_RSU,SND2,RCV2)
/\
role_TA(TA,RSU,OBU,PKt,MPKi,PKi,PKr,Key_set_TA_RSU,Key_set_TA_OBU,SND1,RCV1)
end role

role environment()
def=
        const

pki:public_key,pkt:public_key,trust:agent,hash_0:hash_func,obu:agent,rsu:agent,mpki:public_key,pkr:public_key,auth_1:protocol_id,auth_2:protocol_id,auth_3:protocol_id
    intruder_knowledge = {trust,rsu,obu,pkt,mpki,pki,pkr}
    composition
    session(trust,rsu,obu,pkt,mpki,pki,pkr,{},{})
end role

goal
    authentication_on auth_1
    authentication_on auth_2
    authentication_on auth_3
end goal

environment()
```

(d) Session, Environment, and Goal.

Fig. 5: HLPSL code reflecting the solutions

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

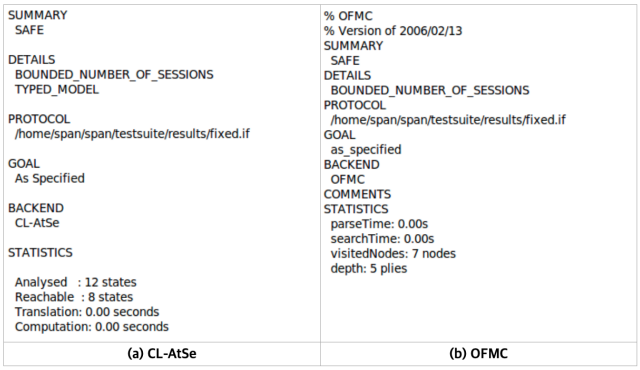

Fig. 6: AVISPA execution results reflecting the solutions

cure authentication in the smart city VANET environment. AVISPA-based validation confirmed that the proposed improvements effectively resolved existing vulnerabilities and demonstrated security against major attacks. This demonstrates the effectiveness of AVISPA verification in addressing security issues in smart city VANET environments.

Future work will focus on optimizing the proposed authentication framework for high mobility smart city environments to ensure faster and more secure communication. Additionally, performance and security validation under various network conditions will be conducted to develop a VANET authentication framework that is practically applicable.

Acknowledgments. This work was supported by the Bisa Research Grant of Keimyung University in 2024 under Project 20240618.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Armando, A., Basin, D., Boichut, Y., Chevalier, Y., Compagna, L., Cuéllar, J., Drielsma, P.H., Héam, P.C., Kouchnarenko, O., Mantovani, J., et al.: The avispa tool for the automated validation of internet security protocols and applications. In: Computer Aided Verification: 17th International Conference, CAV 2005, Edinburgh, Scotland, UK, July 6-10, 2005. Proceedings 17. pp. 281–285. Springer (2005)
2. Canetti, R., Krawczyk, H.: Universally composable notions of key exchange and secure channels. In: Advances in Cryptology—EUROCRYPT 2002: International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam, The Netherlands, April 28–May 2, 2002 Proceedings 21. pp. 337–351. Springer (2002)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1

3. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Transactions on information theory **29**(2), 198–208 (1983)
4. Douceur, J.R.: The sybil attack. In: International workshop on peer-to-peer systems. pp. 251–260. Springer (2002)
5. Dwivedi, S.K., Amin, R., Vollala, S., Das, A.K.: Design of blockchain and ecc-based robust and efficient batch authentication protocol for vehicular ad-hoc networks. IEEE Transactions on Intelligent Transportation Systems (2023)
6. Elassy, M., Al-Hattab, M., Takruri, M., Badawi, S.: Intelligent transportation systems for sustainable smart cities. Transportation Engineering p. 100252 (2024)
7. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996)
8. Heilman, E., Kendler, A., Zohar, A., Goldberg, S.: Eclipse attacks on {Bitcoin's}{peer-to-peer} network. In: 24th USENIX security symposium (USENIX security 15). pp. 129–144 (2015)
9. Hussein, N.H., Yaw, C.T., Koh, S.P., Tiong, S.K., Chong, K.H.: A comprehensive survey on vehicular networking: Communications, applications, challenges, and upcoming research directions. IEEE Access **10**, 86127–86180 (2022)
10. Ju, S., Park, H., Son, S., Kim, H., Park, Y., Park, Y.: Blockchain-assisted secure and lightweight authentication scheme for multi-server internet of drones environments. Mathematics **12**(24), 3965 (2024)
11. Kim, M., Park, K., Park, Y.: A reliable and privacy-preserving vehicular energy trading scheme using decentralized identifiers. Mathematics **12**(10), 1450 (2024)
12. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Post-quantum cryptography, pp. 147–191. Springer (2009)
13. Micciancio, D., Schultz-Wu, M.: Error correction and ciphertext quantization in lattice cryptography. In: Annual International Cryptology Conference. pp. 648–681. Springer (2023)
14. Naeem, M.A., Chaudhary, S., Meng, Y.: Road to efficiency: V2v enabled intelligent transportation system. Electronics **13**(13), 2673 (2024)
15. Nath, H.J., Choudhury, H.: Lbpv: Lattice-based privacy-preserving mutual authentication scheme for vanet. Computers and Electrical Engineering **120**, 109765 (2024)
16. Park, H., Son, S., Park, Y., Park, Y.: Provably quantum secure three-party mutual authentication and key exchange protocol based on modular learning with error. Electronics (2079-9292) **13**(19) (2024)
17. Park, K., Lee, J., Das, A.K., Park, Y.: Bpps: Blockchain-enabled privacy-preserving scheme for demand-response management in smart grid environments. IEEE Transactions on Dependable and Secure Computing **20**(2), 1719–1729 (2022)
18. Shayea, G.G., Mohammed, D.A., Abbas, A.H., Abdulsattar, N.F.: Privacy-aware secure routing through elliptical curve cryptography with optimal rsu distribution in vanets. Designs **6**(6), 121 (2022)
19. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. pp. 124–134. Ieee (1994)
20. Yu, S., Lee, J., Sutrala, A.K., Das, A.K., Park, Y.: Laka-uav: Lightweight authentication and key agreement scheme for cloud-assisted unmanned aerial vehicle using blockchain in flying ad-hoc networks. Computer networks **224**, 109612 (2023)
21. Zhu, D., Guan, Y.: Secure and lightweight conditional privacy-preserving identity authentication scheme for vanet. IEEE Sensors Journal (2024)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97573-8_1