A Light Reconstruction on CPP-BAT in VANETs

Eko Fajar Cahyadi*, Anggun Fitrian Isnawati*, Karisma Trinanda Putra†, Heri Wijayanto‡
*Faculty of Telecomm. & Electrical Engineering, Institut Teknologi Telkom Purwokerto, Purwokerto 53147, Indonesia
†Dept. of Electrical Engineering, Universitas Muhammadiyah Yogyakarta, Yogyakarta 55183, Indonesia
‡Faculty of Engineering, University of Mataram, Mataram 83115, Indonesia
Corresponding email: ekofajarcahyadi@ittelkom-pwt.ac.id

2022 IEEE International Conference on Consumer Electronics - Taiwan | 978-1-6654-7050-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCE-TAIWAN55306.2022.9869141

***Abstract*—In 2013, Shim proposed a conditional privacy-preserving authentication (CPP-BAT) scheme for vehicular ad hoc networks (VANETs). Their scheme was designed to improve Jiang *et al.*'s binary authentication tree (BAT) scheme that claimed insecure against forgery attacks, replay attacks, and Sybil attacks. Unfortunately, we also found out that Shim's CPP-BAT is potentially insecure against non-repudiation attacks. In this short article, we address the issue and give our improvement to withstand the threat.**

***Index Terms*—authentication, BAT, CPP-BAT, non-repudiation attacks, VANETs**

I. Introduction

Vehicular ad hoc networks (VANETs) comprises three main entities, *i.e.*, trusted authority (TA), roadside units (RSUs), and onboard units (OBUs), as depicted in Fig. 1. In this network, vehicles could communicate to each other, as well as to the infrastructure, via vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications, respectively [1]. To support the security and privacy in VANETs, an authentication scheme has a critical role to make sure all entities and the information are valid and verified [2]. In 2009, Jiang *et al.* [3] proposed a binary authentication tree (BAT) scheme for VANETs. The scheme efficiently diminish the bottleneck issue in batch verification performance and so significantly reduced computational overhead. In BAT, RSUs can quickly distinguish illegal signatures from all the authentic ones, allowing them to withstand message flooding attacks. However, in 2013, Shim [4] discovered that the BAT cannot resist the forgery attacks, replay attacks, and Sybil attacks. Therefore, to improve the BAT, Shim [4] proposed a conditional privacy-preserving authentication (CPP-BAT) for VANETs. Unfortunately, we also found that the batch verification phase in CPP-BAT potentially suffered from non-repudiation attacks. A malicious user $\mathcal{M}$ can broadcast false messages to deceive other vehicles, resulting TA unable to trace $\mathcal{M}$ by signature. Therefore, this article proposes a light modification of CPP-BAT to withstand the attack.

II. Weakness on CPP-BAT

This section briefly review the CPP-BAT scheme and discuss its vulnerabilities.

A. Brief Review of CPP-BAT

Shim's CPP-BAT scheme consists of four phases: **setup**, **pseudo-identity generation / private key extraction**, **message signing**, and **verification**. Due to page limitation, we briefly review the scheme. Meanwhile, for a comprehensive explanation, please refer to [4].

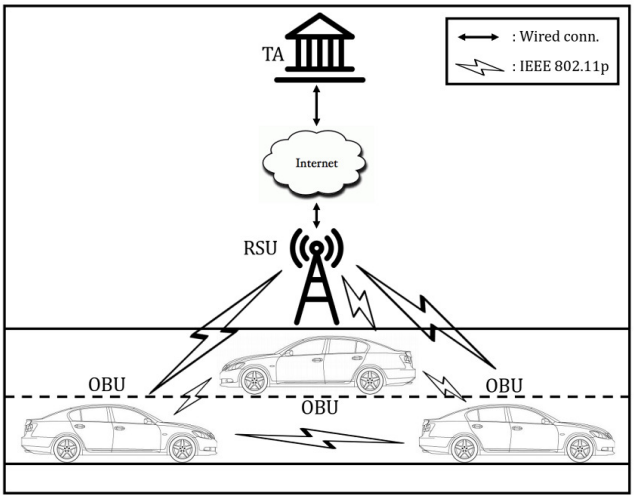

Fig. 1. The topology of VANETs.

1) Setup: TA comprises private key generator (PKG) and trace authority (TRA). After constructing a bilinear map $\hat{e} : G_1 \times G_1 \rightarrow G_2$, PKG computes its public key $P_{pub} = sP$, and picks two hash functions $H_1 \in G_1$ and $H_2 \in Z_q^*$. Meanwhile, TRA computes its public key $T_{pub} = tP$. TA publishes public parameters $params = \{q, \hat{e}, G_1, G_2, P, P_{pub}, T_{pub}, H_1, H_2\}$ to vehicles and RSUs.

2) Pseudo-identity generation / private key extraction: TRA generates $PK_i^* = \{z_{i,1}P, z_{i,2}P, \cdots, z_{i,n}P\}$ and vehicles' pseudo-identity $PID_i^* = \{PID_{i,k} | k = 1, 2, \cdots, n\}$. Then, it sends PID_i^* to PKG for processing the corresponding private key $SK_i^* = \{SK_{i,k} | k = 1, 2, \cdots, n\}$. TA sends $params$ and $(PID_i^*, SK_i^*, PK_i^*)$ to vehicles through a secure channel.

3) Message signing: Vehicle picks $r_i \in Z_q^*$ and computes $U_i = r_iP$, $h_i = H_2(PID_i, M_i, U_i)$, and $V_i = SK_i + h_i r_i P_{pub}$, to produces a signature $\sigma = (U_i, V_i)$ on $M_i = \{PID_i \parallel v_iP \parallel m_i \parallel tt_i\}$. Finally, vehicle sends the final message (M_i, σ_i) to nearby RSU.

4) Verification: After receiving (M_i, σ_i) from vehicles, RSU performs a single verification: $\hat{e}(V_i, P) = \hat{e}([Q_i + h_iU_i], P_{pub})$ or batch verification: $\hat{e}(\sum_{i=k_1}^{k_2} V_i, P) = \hat{e}(\sum_{i=k_1}^{k_2} [H_1(PID_i) + h_iU_i], P_{pub})$ mechanism.

B. Non-repudiation attacks in CPP-BAT

Referring to [5]–[7], since the batch verification phase in CPP-BAT did not employ a random vector to distinguish every signature, $\mathcal{M}$ can deny his/her signatures. Yoon *et al.* [8] classify batch verification into three types: *Type 1* (multiple signatures on a single message by multiple signers), *Type 2* (multiple signatures on multiple messages by a single signer), and *Type 3* (multiple signatures on multiple messages by multiple signers). By considering the *Type 2*, $\mathcal{M}$ can forge individual signatures and make a false batch verification valid. We assume that $\mathcal{M}$ sends three pairs of messages $\{(M_{k_1}, \sigma_{k_1}), (M_{k_1+1}, \sigma_{k_1+1}), (M_{k_2}, \sigma_{k_2})\}$ to RSU.

1) First method: $\mathcal{M}$ can swap signatures in $\{(M_{k_1}, \sigma_{k_1}), (M_{k_1+1}, \sigma_{k_1+1}), (M_{k_2}, \sigma_{k_2})\}$ to become $\{(M_{k_1}, \sigma_{k_1+1}), (M_{k_1+1}, \sigma_{k_2}), (M_{k_2}, \sigma_{k_1})\}$. When RSU receives the pairs, it will prove the correctness of the signature summation by checking:

$$\begin{aligned}\hat{e}(\sum_{i=k_1}^{k_2} V_i^{'}, P) &= \hat{e}(V_{k_1+1} + V_{k_2} + V_{k_1}, P) \\ &= \hat{e}(V_{k_1} + V_{k_1+1} + V_{k_2}, P) \\ &= \hat{e}(\sum_{i=k_1}^{k_2} [H_1(PID_i) + h_i U_i], P_{pub})\end{aligned} \tag{1}$$

From (1), RSU will consider those changes are legal, since their sum remains the same, even though the orders of those signatures have been changed. In fact, $\mathcal{M}$ can deny if he/she had sent these messages to RSU, because $V_i \neq V_i^{'}$.

2) Second method: Let $V_i^{'} = a_i \times V_i$ and $\sum_{i=k_1}^{k_2} a_i = 1$. $\mathcal{M}$ sends three messages $\{(M_{k_1}, \sigma_{k_1}), (M_{k_1+1}, \sigma_{k_1+1}), (M_{k_2}, \sigma_{k_2})\}$ to RSU and let $V_{k_1}^{'} = 0.5V_{k_1}$, $V_{k_1+1}^{'} = 0.3V_{k_1+1}$, $V_{k_2}^{'} = 0.2V_{k_2}$. It will prove the correctness of the signatures summation by checking:

$$\begin{aligned}\hat{e}(\sum_{i=k_1}^{k_2} V_i^{'}, P) &= \hat{e}(0.5V_{k_1} + 0.3V_{k_1+1} + 0.2V_{k_2}, P) \\ &= \hat{e}(V_{k_1} + V_{k_1+1} + V_{k_2}, P) \\ &= \hat{e}(\sum_{i=k_1}^{k_2} [H_1(PID_i) + h_i U_i], P_{pub})\end{aligned} \tag{2}$$

From (2), RSU considers the signatures $V_{k_1}^{'}$, $V_{k_1+1}^{'}$, and $V_{k_2}^{'}$ are legal to $\{(M_{k_1}, \sigma_{k_1}), (M_{k_1+1}, \sigma_{k_1+1}), (M_{k_2}, \sigma_{k_2})\}$, since their sum remains the same, even though $\mathcal{M}$ has forged all the signatures by gives it a particular value. In this case, $\mathcal{M}$ also can deny his/her signatures, because $V_i \neq V_i^{'}$.

III. Our Improvement

To resolve the non-repudiation attacks, RSU should generate a random vector v_i, where $i = 1, 2, \cdots, n$. This concept is obtained from the small exponent test conducted in [9] and [10]. The value v_i ranges between 1 and 2^l, where l is a security parameter with a small value and does not make any computational overhead. Parameter l is set to the maximum probability of 2^{-l}, so even with a single signature in the batch is wrong, it still can be detected, except with the possibility 2^{-l}. Therefore, the batch verification process become $\hat{e}(\sum_{i=k_1}^{k_2} v_i[H_1(PID_i) + h_i U_i], P_{pub})$.

By this improvement, if $\mathcal{M}$ wants to deny his/her signatures, it will result in the batch failing. Givenly P is a generator in G_1, we have (V_{k_1}, y_{k_1}), (V_{k_1+1}, y_{k_1+1}), $\cdots$, (V_{k_2}, y_{k_2}), with $V_i \in Z_q^*$ and $y_i \in G_1$, check if $\forall i \in \{k_1, k_1+1, \cdots, k_2\}$ satisfy $\hat{e}(V_i, P) = \hat{e}(y_i, Q)$, by doing the following steps:

- Selects random parameter $l_{k_1}, l_{k_1+1}, \cdots, l_{k_2} \in \{0, 1\}^l$
- Computes $A = \sum_{i=k_1}^{k_2} l_i y_i$ and $B = \sum_{i=k_1}^{k_2} l_i V_i$
- If $\hat{e}(B, P) = \hat{e}(A, Q)$, then accepts, otherwise rejects.

The batch instance will become $(V_{k_1}, y_{k_1}), (V_{k_1+1}, y_{k_1+1}), \cdots, (V_{k_2}, y_{k_2})$, with $y_i = [H_1(PID_i) + h_i U_i], P_{pub}$. The verification of the signature consists of checking operation that $\hat{e}(V_i, P) = \hat{e}(y_i, Q)$. If $\mathcal{M}$ wants to make some false multiple digital signatures V_i valid, he/she must make those operation holds. Since $\mathcal{M}$ did not know the values of l that leads to the value of v_i, it is difficult for $\mathcal{M}$ to make $\hat{e}(V_i, P) = \hat{e}(y_i, Q)$ holds.

IV. Conclusion

VANETs can be regarded as the future of our road transportation systems. In this paper, we have shown that CPP-BAT for VANETs is vulnerable to non-repudiation attacks. To counteract the threat, we proposed an improvement in its batch verification scheme by using a vector parameter v_i to identify each message without incurring any computational cost. As a result, we have added an extra feature of security to the method by keeping the original paper's efficiency.

References

[1] E. F. Cahyadi and M-S. Hwang, "A comprehensive survey on certificateless aggregate signature in vehicular ad hoc networks," *IETE Technical Review*, 2022. DOI: 10.1080/02564602.2021.2017800.

[2] E. F. Cahyadi and M-S. Hwang, "An improved efficient anonymous authentication with conditional privacy-preserving scheme for VANETs," *PLoS ONE*, vol. 16, no. 9, e0257044, 2021.

[3] Y. Jiang, M. Shi, X. Shen, and C. Lin, "BAT: a robust signature scheme for vehicular networks using binary authentication tree," *IEEE Transactions on Wireless Communications*, vol. 8, no. 4, pp. 1974–1983, 2009.

[4] K-A. Shim, "Reconstruction of a secure authentication scheme for vehicular ad hoc networks using a binary authentication tree," *IEEE Transactions on Wireless Communications*, vol. 12, no. 11, pp. 5386–5393, 2013.

[5] C-C. Lee and Y-M. Lai, "Toward a secure batch verification with group testing for VANET," *Wireless Networks*, vol. 9, pp. 1441–1449, 2013.

[6] M-S. Hwang, I-C. Lin, and K-F. Hwang, "Cryptanalysis of the batch verifying multiple RSA digital signatures," *Informatica*, vol. 11, no. 1, pp. 15–19, 2000.

[7] M-S. Hwang, C-C. Lee, and Y-L. Tang, "Two simple batch verifying multiple digital signatures', in *Qing, S., Okamoto, T. and Zhou, J. (Eds.): Information and Communications Security, ICICS 2001*, 2001, LNCS, vol. 2229, pp. 233–237.

[8] H. Yoon, J. H. Cheon, and Y. Kim, "Batch verification with ID-based signature," in *Park, C. and Chee, S. (Eds.): Information Security and Cryptology – ICISC 2004*, 2005, LNCS, vol. 3506, pp. 233–248.

[9] M. Bellare, J. A. Garay, and T. Rabin, "Fast batch verification for modular exponentiation and digital signatures," in *Nyberg, K. (Ed.): Advances in Cryptology – Eurocrypt'98*, 1998, LNCS, vol. 2139, pp. 236–250.

[10] S-J. Horng, S-F. Tzeng, Y. Pan, P. Fan, X. Wang, T. Li, and M. K. Khan, "b-SPECS+: batch verification for secure pseudonymous authentication in VANET," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1860–1875, 2013.