This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2923156, IEEE Transactions on Information Forensics and Security

SER.

MSOC.Eval($PPR, F, sk_{f,ser}, sk_{f,csp}, pbk_{CSP}, pvk_{CSP}, C_{Sen_i}(i = 1, 2, \cdots, n_S)$): The cloud server SER firstly decrypts $r_i = f^{-1}_{sk_{f,ser}}(C_{i,ser})$ by using its secret key $sk_{f,ser}$ and checks whether $C^{'}_{i,ser} = H_0(r_i \parallel C_{i,1} \parallel \cdots \parallel C_{i,n_i})$ holds. If it fails, SER halts the protocol; otherwise, it sends $C_{i,i'}, C_{i,csp}, C^{'}_{i,csp}$ to the CSP.

The CSP firstly decrypts $N_i = f^{-1}_{sk_{f,csp}}(C_{i,csp})$ by using its secret key $sk_{f,csp}$ and checks whether $C^{'}_{i,csp} = H_0(N_i \parallel C_{i,1} \parallel \cdots \parallel C_{i,n_i})$ holds. If it fails, CSP halts the protocol; otherwise, it randomly selects $r_{i,csp} \in_R \{0,1\}^{2\lambda}$, and re-encrypts the blinded inputs

$$
\begin{aligned}
&C^{'}_{i,i'} = C_{i,i'} \ mod \ N_i = r_i m_{i,i'} \ mod \ N_i,\\
&C^{'}_{i,i',q} = C^{'}_{i,i'} \ mod \ q, C^{'}_{i,i',p} = C^{'}_{i,i'} \ mod \ p,\\
&C^{"}_{i,i'} = (p^{-1}p(C^{'}_{i,i',q})^q + q^{-1}q(C^{'}_{i,i',p})^p\\
&\qquad +r_{i,csp}N) \ mod \ T,\\
&C^{'}_{rec,csp} = H_1(C^{'}_{1,csp} \parallel \cdots \parallel C^{'}_{n_S,csp} \parallel C^{"}_{1,1} \parallel \cdots \parallel\\
&\qquad C^{"}_{1,n_1} \parallel \cdots \parallel C^{"}_{n_S,1} \parallel \cdots \parallel C^{"}_{n_S,n_{n_S}}),
\end{aligned} \tag{4}
$$

where $p^{-1}p \equiv 1 \ mod \ q$ and $q^{-1}q \equiv 1 \ mod \ p$. Finally, the CSP sends $C_{CSP} = (C^{"}_{i,i'}, C^{'}_{rec,csp})$ to the cloud server SER.

Without loss of generality, it is assumed that the degree of the j-th item $Item_j = a_j \prod_{l=1}^{n} x_l^{t_{l,j}} (j = 1, 2, \cdots, K)$ of the outsourced multivariate polynomial F is deg_j. After receiving C_{CSP}, the cloud server SER firstly checks whether $C^{'}_{rec,csp} = H_1(C^{'}_{1,csp} \parallel \cdots \parallel C^{'}_{n_S,csp} \parallel C^{"}_{1,1} \parallel \cdots \parallel C^{"}_{1,n_1} \parallel \cdots \parallel C^{"}_{n_S,1} \parallel \cdots \parallel C^{"}_{n_S,n_{n_S}})$ holds. If it fails, the SER halts the protocol; otherwise it randomly selects $r \in_R \{0,1\}^{2\lambda}$ with the condition that $r \in \mathbb{Z}^*_T$ and computes

$$
C^{"}_{i,i',ser} = r_i^{-1} C^{"}_{i,i'}, C^{"}_{i,i',SER} = r C^{"}_{i,i',ser}. \tag{5}
$$

Then, it computes

$$
\begin{aligned}
&C_{Item_j} = r^{deg_F - deg_j} a_j \prod_{l=1}^{n} (C^{"}_{l,SER})^{t_{l,j}},\\
&C^{bld}_F = \sum_{j=1}^{K} C_{Item_j}, C^{bld,'}_F = H_1(C^{'}_{rec,csp} \parallel C^{bld}_F),
\end{aligned} \tag{6}
$$

where $n = \sum_{i=1}^{n_S} n_i$, $\cup_{l=1}^{n}\{C^{"}_{l,SER}\} = \cup_{i=1,i'=1}^{n_S,n_i}\{C^{"}_{i,i',SER}\}$, $\cup_{l=1}^{n}\{t_{l,j}\} = \cup_{i=1,i'=1}^{n_S,n_i}\{t_{i,i',j}\}$ and sends $C_{SER} = (C^{bld}_F, C^{'}_{rec,csp}, C^{bld,'}_F)$ to the CSP.

After receiving C_{SER}, the CSP checks whether $C^{bld,'}_F = H_1(C^{'}_{rec,csp} \parallel C^{bld}_F)$ holds. If it fails, CSP halts the protocol; otherwise, it computes

$$
\begin{aligned}
&C^{CSP,1}_F = C^{bld}_F \ mod \ N,\\
&C^{CSP,2}_F = f_{pk_{f,rec}}(C^{CSP,1}_F),\\
&C^{CSP,3}_F = H_1(C^{CSP,1}_F \parallel C^{CSP,2}_F),
\end{aligned} \tag{7}
$$

and sends $C^{CSP}_F = (C^{CSP,2}_F, C^{CSP,3}_F)$ back to the cloud server SER.

Finally, the SER computes

$$
C_{rec,ser} = f_{pk_{f,rec}}(r), C^{'}_F = H_1(r \parallel C^{CSP,2}_F), \tag{8}
$$

and sends $C_F = (C^{CSP}_F, C_{rec,ser}, C^{'}_F)$ to the receiver REC.

MSOC.Dec($PPR, sk_{f,rec}, C_F$): The receiver REC firstly decrypts

$$
\begin{aligned}
&r = f^{-1}_{sk_{f,rec}}(C_{rec,ser}),\\
&C^{CSP,1}_F = f^{-1}_{sk_{f,rec}}(C^{CSP,2}_F),
\end{aligned} \tag{9}
$$

by using its secret key $sk_{f,rec}$. Then, it checks whether both $C^{CSP,3}_F = H_1(C^{CSP,1}_F \parallel C^{CSP,2}_F)$ and $C^{'}_F = H_1(r \parallel C^{CSP,2}_F)$ hold. If it fails, the REC halts the protocol; otherwise, it decrypts the result of outsourced computation

$$
F(m_1, m_2, \cdots, m_n) = r^{-deg_F} C^{CSP,1}_F. \tag{10}
$$

5 THE PROPOSED LIGHTWEIGHT PRIVACY-PRESERVING AUTHENTICATION PROTOCOL LPPA

In this section, a lightweight privacy-preserving authentication protocol LPPA for LBS in VANETs is proposed. Before giving the description in detail, an efficient and secure comparison protocol LSCP is firstly devised as the cornerstone of our final design.

5.1 The Proposed LSCP

In this subsection, based on our proposed MSOC, a lightweight and secure comparison protocol LSCP is proposed, which comprises the following algorithms **LSCP.Setup**, **LSCP.Enc**, **LSCP.Comp** and **LSCP.Dec**. The proposed **LSCP.Setup** and **LSCP.Enc** are the same as **MSOC.Setup** and **MSOC.Enc**.

LSCP.Comp($PPR, f_{jud}, sk_{f,ser}, sk_{f,csp}, C_{m_i}, C_{m_j}$): Taking the ciphertexts $C_{m_i} = MSOC.Enc(PPR, pbk_i, pvk_i, m_i)$ and $C_{m_j} = MSOC.Enc(PPR, pbk_j, pvk_j, m_j)$ of two integers m_i and m_j with $\lambda^{'} = 2\lambda$-bit long as input, the issue of deciding whether m_i is larger than m_j in the encrypted domain can be transformed into evaluating a corresponding judging polynomial $f_{jud}(C_{m_i}, C_{m_j})$ in the encrypted domain.

We show how to construct the judging polynomial f_{jud} in the plaintext as follows. Firstly, without loss of generality, it is noted that m_i can be represented in the following form (i.e. C_{m_j} can be performed the same)

$$
m_i = m_{i,\lambda'-1} 2^{\lambda^{'}-1} + m_{i,\lambda'-2} 2^{\lambda^{'}-2} + \cdots + m_{i,0}. \tag{11}
$$

Therefore, the binary representation of m_i, namely $m_{i,\lambda'-1} m_{i,\lambda'-2} \cdots m_{i,0}$ can be straightforwardly derived by bit decomposition.

Then, it is observed that for single bit comparison between $m_{i,s}(s = 0, 2, \cdots, \lambda^{'} - 1)$ and $m_{j,s}$, we can

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2923156, IEEE Transactions on Information Forensics and Security

obtain the judging polynomial $f_{jud,l}(m_{i,s}, m_{j,s})(s = 1, 2, \cdots, \lambda' - 1; l = 1, 2, 3)$ by the truth table method such that

$$\begin{aligned} f_{jud,1}(m_{i,s}, m_{j,s}) &= m_{i,s} - m_{i,s}m_{j,s} = 1 \\ &\quad if\ and\ only\ if\ m_{i,s} > m_{j,s}, \\ f_{jud,2}(m_{i,s}, m_{j,s}) &= 2m_{i,s}m_{j,s} - m_{i,s} - m_{j,s} + 1 = 1 \\ &\quad if\ and\ only\ if\ m_{i,s} = m_{j,s}, \\ f_{jud,3}(m_{i,s}, m_{j,s}) &= m_{j,s} - m_{i,s}m_{j,s} = 1 \\ &\quad if\ and\ only\ if\ m_{i,s} < m_{j,s}. \end{aligned} \tag{12}$$

By Eqn. (12) we can transfer the single bit comparison to checking whether the corresponding judging polynomial $f_{jud,l}(m_{i,s}, m_{j,s})$ equals 0 or 1. Therefore, the comparison between two integers m_i and m_j of size λ' can be achieved by sequentially performing the bit comparison from the most significant bit to the least significant one and a binary chopping method would enhance the efficiency of comparison. Let $L = \lceil \frac{\lambda'}{2} \rceil$, we have

$$\begin{aligned} m_i &= \underbrace{m_{i,\lambda'-1}, \cdots, m_{i,L}}_{m_{i,h}} \underbrace{m_{i,L-1}, \cdots, m_{i,0}}_{m_{i,l}}, \\ m_j &= \underbrace{m_{j,\lambda'-1}, \cdots, m_{j,L}}_{m_{j,h}} \underbrace{m_{j,L-1}, \cdots m_{j,0}}_{m_{j,l}}. \end{aligned} \tag{13}$$

As same as is used for constructing the judging polynomial for bit comparison, the judging polynomial $f_{jud}(m_i, m_j)$ between integers m_i and m_j can be constructed by recursively exploiting the following judging polynomial for $m_{i,h}, m_{i,l}, m_{j,h}, m_{j,l}$ with the binary chopping method

$$\begin{aligned} & f_{jud}(m_i, m_j) \\ & = f_{jud,1}(m_{i,h}, m_{j,h})(1 - f_{jud,2}(m_{i,h}, m_{j,h})) \\ & (1 - f_{jud,3}(m_{i,h}, m_{j,h}))(f_{jud,2}(m_{i,l}, m_{j,l}) \\ & (1 - f_{jud,1}(m_{i,l}, m_{j,l}))(1 - f_{jud,3}(m_{i,l}, m_{j,l})) \\ & + f_{jud,3}(m_{i,l}, m_{j,l})(1 - f_{jud,2}(m_{i,l}, m_{j,l})) \\ & (1 - f_{jud,1}(m_{i,l}, m_{j,l}))) + f_{jud,1}(m_{i,l}, m_{j,l}) \\ & (1 - f_{jud,2}(m_{i,l}, m_{j,l}))(1 - f_{jud,3}(m_{i,l}, m_{j,l})) \\ & (f_{jud,1}(m_{i,h}, m_{j,h})(1 - f_{jud,2}(m_{i,h}, m_{j,h})) \\ & (1 - f_{jud,3}(m_{i,h}, m_{j,h})) + (1 - f_{jud,1}(m_{i,h}, m_{j,h})) \\ & f_{jud,2}(m_{i,h}, m_{j,h})(1 - f_{jud,3}(m_{i,h}, m_{j,h}))) \end{aligned} \tag{14}$$

such that

$$f_{jud}(m_i, m_j) = \begin{cases} 1, m_i > m_j, \\ 0, otherwise. \end{cases} \tag{15}$$

In our case, owing to the property of full homomorphism of our proposed MSOC and the fact that both the bit decomposition (i.e. the ciphertext of power of 2 can be also generated by senders Sen_i or Sen_j) and the recursive judging polynomial calculation only require multivariate polynomial evaluation, the cloud SER can calculate the judging polynomial $f_{jud}(m_i, m_j)$ in the encrypted domain namely $C_{f_{jud}(m_i,m_j)} = f_{jud}(C_{m_i}, C_{m_j})$, by exploiting the algorithm $MSOC.Evl(PPR, f_{jud}, sk_{f,ser}, sk_{f,csp}, C_{m_i}, C_{m_j})$.

LSCP.Dec is the same as **MSOC.Dec**. If the decryption result $f_{jud}(m_i, m_j) = 1$, the receiver decides $m_i > m_j$; otherwise, $m_i \leq m_j$.

TABLE 2: Notation Description for LPPA

Notation	Description
$m_{i,j}$	The j-th LBS message generated by vehicular user U_i
$Index_{i,j}$	The message identifier of $m_{i,j}$
$(x_{i,j}, y_{i,j})$	The coordinates denoting the location where LBS message $m_{i,j}$ is collected
$t_{i,j}$	The time when LBS message $m_{i,j}$ is collected
$R_{i,k,j}$	The rating of vehicular user U_i on user U_k's j-th LBS message $m_{k,j}$
$S(U_i, U_t)$	The similarity between vehicular users U_i and U_t
$RED_{k,j'}$	The redundancy factor denoting whether the j'-th LBS message $m_{k,j'}$ of vehicular user U_k is redundant
$PR_{i,k,j'}$	The predicted rating of vehicular user U_i on LBS message $m_{k,j'}$
T_a	The threshold for LBS message filtering

5.2 The Proposed LPPA

In this subsection, based on our proposed MSOC and LSCP, a lightweight privacy-preserving authentication protocol LPPA for location-based services in VANETs is proposed, by devising an efficient information filtering system in the encrypted domain. It is assumed that there exist n_u vehicular users $U_i(i = 1, 2, \cdots, n_u)$ in district $s(s = 1, 2, \cdots, n_d)$ managed by RSU_s. To efficiently achieve fine-grained encrypted LBS message access control and permit the authorized vehicular users to successfully decrypt LBS services, a ciphertext-policy attribute-based encryption (CP-ABE) is adopted, which is composed of the algorithms $ABE.Setup(1^\lambda), ABE.KeyGen(MSK, S), ABE.Enc(PPAR, m, \mathbb{A}), ABE.Dec(PPAR, C, SK)$. For LBS message authentication, an existentially unforgeable secure signature scheme Λ under adaptively chosen message attack is also adopted, which is composed of the algorithms $\Lambda.KeyGen(1^\lambda), \Lambda.Sign(sk, m), \Lambda.Verify(pk, m, \sigma)$. Table 2 shows the notations used in LPPA. The proposed LPPA comprises the following four algorithms: **Setup**, **LBS Message Generation**, **LBS Message Filtering** and **LBS Message Decryption and Verification**, which are presented as follows.

Setup: On input 1^λ where λ is the security parameter, it runs the algorithm $MSOC.Setup(1^\lambda)$ to generate pairs of public key and secret key $(pk_{f,RSU_s}, sk_{f,RSU_s})$, $(pk_{f,csp}, sk_{f,csp})$ and (pk_{f,U_i}, sk_{f,U_i}) respectively for the $RSU_s(s = 1, 2, \cdots, n_d)$, the CSP and each vehicular user $U_i(i = 1, 2, \cdots, n_u)$, where f, f^{-1} on $\{0,1\}^{2\lambda}$ is a pair of one-way trapdoor permutations. $H_0, H_1 : \{0,1\}^* \to \{0,1\}^{2\lambda}$ are cryptographic hash functions. It also runs $ABE.Setup(1^\lambda)$ to generate public parameter $PPAR_{ABE}$ and master secret key MSK_{ABE}. The public parameters are $PPR = (pk_{f,RSU_s}, pk_{f,csp}, pk_{f,U_i}, PPAR_{ABE}, H_0, H_1)$ and the secret keys are $sk_{f,RSU_s}, sk_{f,csp}, sk_{f,U_i}, MSK_{ABE}$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2923156, IEEE Transactions on Information Forensics and Security

respectively kept private by RSU_s, CSP, U_i and the system.

LBS Message Generation: Without loss of generality, it is assumed that each vehicular user U_i generates its j-th LBS message $m_{i,j}(i = 1,2,\cdots,n_u; j = 1,2,\cdots,n_i)$ associated to the tuple $Info_{i,j} = (U_i, U_i, Index_{i,j}, x_{i,j}, y_{i,j}, t_{i,j}, r_{i,i,j})$ where $Index_{i,j}, (x_{i,j}, y_{i,j}), t_{i,j}$ and $r_{i,i,j}$ denote the message identifier of $m_{i,j}$, the coordinates of the location and the time the message $m_{i,j}$ is collected, and the rating of user U_i on its generated LBS message $m_{i,j}$. It is assumed that each U_i rates its own messages with the highest score, namely $r_{i,i,j} = 5$ where the rating values range from 1 to 5 as integers. A rounding function is applied on the location coordinates and the time to guarantee $(x_{i,j}, y_{i,j}, t_{i,j})$ to be integers for cryptographic exploitation. Let $r_{i,k,j}(k = 1,2,\cdots,n_u)$ be the user U_i's rating on user U_k's j-th message $m_{k,j}(j = 1,2,\cdots,n_k)$. In the following two cases that U_i has not received $m_{k,j}$ from other users or that U_i decides $m_{k,j}$ as redundant LBS message, the rating $r_{i,k,j}$ is set to 0. Each vehicular user U_i and RSU_s respectively generates (pbk_i, pvk_i) and (pbk_s, pvk_s) by exploiting the algorithm $MSOC.KeyGen(PPR)$. It also derives $sk_{ABE,i}, (pk_{\Lambda,i}, sk_{\Lambda,i})$ for user U_i, by running $ABE.KeyGen(MSK_{ABE}, S_i)$ and $\Lambda.KeyGen(1^\lambda)$ where S_i is the attribute set of user U_i.

1) Each U_i generates the encrypted LBS message for $m_{i,j}$

$$\begin{aligned}
&C_{i,j} = ABE.Enc(PPAR_{ABE}, m_{i,j}, \mathbb{A}),\\
&\sigma_{i,j} = \Lambda.Sign(sk_{\Lambda,i}, C_{i,j}),\\
&C_{Loc_{i,j,x}} = MSOC.Enc(PPR, pbk_i, pvk_i, x_{i,j}),\\
&C_{Loc_{i,j,y}} = MSOC.Enc(PPR, pbk_i, pvk_i, y_{i,j}),\\
&C_{t_{i,j}} = MSOC.Enc(PPR, pbk_i, pvk_i, t_{i,j}),
\end{aligned} \quad (16)$$

where $\mathbb{A}$ is the access policy permitting that the authorized LBS users can successfully decrypt the underlying LBS message $m_{i,j}$, and broadcasts $Info_{i,j} = (U_i, U_i, Index_{i,j}, C_{i,j}, \sigma_{i,j}, C_{Loc_{i,j,x}}, C_{Loc_{i,j,y}}, C_{t_{i,j}})$ in its neighborhood.

2) Each vehicular user U_i initializes and updates a table T_i by storing the encrypted tuples $Info_{l,j} = (U_i, U_l, Index_{l,j}, C_{Loc_{l,j,x}}, C_{Loc_{l,j,y}}, C_{t_{l,j}}, C_{R_{i,l,j}})(l = 1,2,\cdots,n_u; j = 1,2,\cdots,n_l)$ associated to $m_{l,j}$ as the j-th LBS message it generated itself if $l = i$ or accepted as the j-th generated LBS message from other vehicles U_l if $l \neq i$, where

$$C_{R_{i,l,j}} = MSOC.Enc(PPR, pbk_i, pvk_i, R_{i,l,j}) \quad (17)$$

is the ciphertext of its rating $R_{i,l,j}$ on each LBS message $m_{l,j}$ located in table T_i. Then, for each tuple in table T_i, user U_i randomly selects $r^i_{l,j,x}, r^{i,'}_{l,j,x}, r^i_{l,j,y}, r^{i,'}_{l,j,y}, r^i_{l,j,t}, r^{i,'}_{l,j,t} \in_R \{0,1\}^{2\lambda}$ with the conditions that $r^i_{l,j,x}, r^i_{l,j,y}, r^i_{l,j,t} \in \mathbb{Z}^*_T$ and $r^{i,'}_{l,j,x}, r^{i,'}_{l,j,y}, r^{i,'}_{l,j,t} \in \mathbb{Z}_{T_l}$, where T, T_l are the temporary public keys of CSP and user U_l, re-encrypts it as

$$\begin{aligned}
&C^i_{l,j,x} = f_{pk_{f,RSU_s}}(r^i_{l,j,x}),\\
&C^i_{l,j,y} = f_{pk_{f,RSU_s}}(r^i_{l,j,y}), C^i_{l,j,t} = f_{pk_{f,RSU_s}}(r^i_{l,j,t}),\\
&C^{i,'}_{l,j,x} = f_{pk_{f,csp}}(r^{i,'}_{l,j,x}),\\
&C^{i,'}_{l,j,y} = f_{pk_{f,csp}}(r^{i,'}_{l,j,y}), C^{i,'}_{l,j,t} = f_{pk_{f,csp}}(r^{i,'}_{l,j,t}),\\
&C^{i,'}_{Loc_{l,j,x}} = r^i_{l,j,x} r^{i,'}_{l,j,x} C_{Loc_{l,j,x}},\\
&C^{i,'}_{Loc_{l,j,y}} = r^i_{l,j,y} r^{i,'}_{l,j,y} C_{Loc_{l,j,y}},\\
&C^{i,'}_{t_{l,j}} = r^i_{l,j,t} r^{i,'}_{l,j,t} C_{t_{l,j}}.
\end{aligned} \quad (18)$$

Finally, user U_i constructs table $T^{'}_i$ comprising the tuples $Info^{'}_{i,l,j} = (U_i, U_l, Index_{l,j}, C^{i,'}_{Loc_{l,j,x}}, C^{i,'}_{Loc_{l,j,y}}, C^{i,'}_{t_{l,j}}, C_{R_{i,l,j}})$ and sends $T^{'}_i$ to the RSU_s. Note that the blinding factors $r^i_{l,j,x}, r^i_{l,j,y}, r^i_{l,j,t}$ and $r^{i,'}_{l,j,x}, r^{i,'}_{l,j,y}, r^{i,'}_{l,j,t}$ are respectively adopted in the re-encryption to achieve the interest pattern privacy of vehicular users (i.e. please refer to Sec. 6.2 for the security proof).

LBS Message Filtering: The roadside unit $RSU_s(s = 1,2,\cdots,n_d)$ predicts the rating in the encrypted domain for each vehicular user $U_i(i = 1,2,\cdots,n_u)$ currently travelling in district s on LBS message $m_{k,j'}$ which U_i has not rated before.

1) For each LBS message $m_{k,j'}(k = 1,2,\cdots,n_u; j' = 1,2,\cdots,n_k)$ as the j'-th message generated by user U_k, for each tuple in $T^{'}_i$ sent by user U_i, RSU_s computes the function

$$\begin{aligned}
&F_{dist}(Info_{k,j'}, Info_{l,j})\\
&= (Loc_{k,j',x} - Loc_{l,j,x})^2 + (Loc_{k,j',y} - Loc_{l,j,y})^2\\
&+ (t_{k,j'} - t_{l,j})^2
\end{aligned} \quad (19)$$

in the encrypted domain by exploiting the algorithm

$$\begin{aligned}
C_{Dist_{k,j',l,j}} = \; & MSOC.Eval(PPR, F_{dist}, sk_{f,RSU_s},\\
& sk_{f,csp}, T^{'}_k, T^{'}_i),
\end{aligned} \quad (20)$$

where $T^{'}_k$ is the table maintained and updated by user U_k.

Note that to evaluate on the re-encryption ciphertexts (i.e. we take $C^{i,'}_{Loc_{l,j,x}}$ in the updated table $T^{'}_i$ for example), the CSP firstly deciphers $r^{i,'}_{l,j,x} = f^{-1}_{sk_{f,csp}}(C^{i,'}_{l,j,x})$, computes $C^{i,csp}_{Loc_{l,j,x}} = ((r^{i,'}_{l,j,x})^{-1} C^{i,'}_{Loc_{l,j,x}}) \ mod\ N_l$, and $C^{i,ser,bld}_{Loc_{l,j,x}} = p^{-1}p(C^{i,csp}_{Loc_{l,j,x}})^q_q + q^{-1}q(C^{i,csp}_{Loc_{l,j,x}})^q_q + r^{i,csp}_{l,j,x} N \ mod\ T$, where $r^{i,csp}_{l,j,x} \in_R \{0,1\}^\lambda$. Afterwards the RSU_s is also required to compute $r^i_{l,j,x} = f^{-1}_{sk_{f,RSU_s}}(C^i_{l,j,x})$ and execute an additional deblinding operation that $C^{i,ser}_{Loc_{l,j,x}} = r_l^{-1}(r^i_{l,j,x})^{-1} C^{i,ser,bld}_{Loc_{l,j,x}}$ where r_l is the blinding factor adopted to encrypt $Loc_{l,j,x}$ by the algorithm $MSOC.Enc$ in Eqn. (16) and can be decrypted by RSU_s using its secret key sk_{f,RSU_s}. The same operations are required to be performed on other re-encryption ciphertexts both in the updated table $T^{'}_i$ and $T^{'}_k$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2923156, IEEE Transactions on Information Forensics and Security

Then, RSU_s computes

$$C_{T_d} = MSOC.Enc(PPR, pbk_s, pvk_s, T_d), \quad (21)$$

where T_d is the threshold to decide whether an LBS message is redundant, and for each tuple in table $T_i^{'}$ associated to user U_l's j-th LBS message accepted by user U_i, compares $Dist_{k,j',l,j}(l = 1, 2, \cdots, n_u; j = 1, 2, \cdots, n_l; j^{'} = 1, 2, \cdots, n_k)$ and T_d, namely evaluating the judging polynomial $f_{jud}(Dist_{k,j',l,j}, T_d)$ in the encrypted domain by exploiting the algorithm

$$\begin{aligned} C_{f_{jud,k,j',l,j}} &= LSCP.Comp(PPR, f_{jud}, sk_{f,RSU_s}, \\ &\quad sk_{f,csp}, C_{Dist_{k,j',l,j}}, C_{T_d}). \quad (22) \end{aligned}$$

2) RSU_s computes the similarity $S(U_i, U_t)(i, t = 1, 2, \cdots, n_u)$ between users U_i and U_t

$$S(U_i, U_t) = \frac{(\sum_{j=1}^{\sum_{l=1}^{n_u} n_l} R_{i,l,j} R_{t,l,j})^2}{\sum_{j=1}^{\sum_{l=1}^{n_u} n_l} R_{i,l,j}^2 \sum_{j=1}^{\sum_{l=1}^{n_u} n_l} R_{t,l,j}^2} \quad (23)$$

in the encrypted domain, by performing the algorithm

$$\begin{aligned} C_{S(U_i,U_t)} &= MSOC.Evl(PPR, S(U_i, U_t), sk_{f,RSU_s}, \\ &\quad sk_{f,csp}, T_i^{'}, T_t^{'}), \quad (24) \end{aligned}$$

where $T_t^{'}$ is the table maintained and updated by user U_t. Then, it predicts user U_i's rating on LBS message $m_{k,j'}$, by computing

$$PR_{i,k,j'} = RED_{k,j'} \frac{\sum_{t=1,t\neq i}^{n_u} R_{t,k,j'} S(U_i, U_t)}{\sum_{t=1,t\neq i}^{n_u} S(U_i, U_t)} \quad (25)$$

in the encrypted domain through performing the algorithm

$$\begin{aligned} C_{PR_{i,k,j'}} &= MSOC.Evl(PPR, PR_{i,k,j'}, sk_{f,RSU_s}, \\ &\quad sk_{f,csp}, C_{RED_{k,j'}}, T_t^{'}, C_{S(U_i,U_t)}), \quad (26) \end{aligned}$$

where $C_{RED_{k,j'}} = \prod_{j=1}^{\sum_{l=1}^{n_u} n_l} C_{f_{jud,k,j',l,j}}$ is the encrypted redundancy factor. Finally, RSU_s transmits all the encrypted prediction ratings $C_{PR_{i,k,j'}} (k = 1, 2, \cdots, n_u; j^{'} = 1, 2, \cdots, n_k)$ to vehicular user U_i.

LBS Message Decryption and Verification: While receiving a newly-arriving LBS message $m_{k,j'}$, vehicular user U_i firstly recovers the prediction rating by performing

$$PR_{i,k,j'} = MSOC.Dec(PPR, sk_{f,U_i}, C_{PR_{i,k,j'}}), \quad (27)$$

and compares it to the predefined threshold T_a. If $PR_{i,k,j'} = 0$, U_i considers LBS message $m_{k,j'}$ to be duplicate to the messages she/he has accepted, discards and prevents it from being further broadcasted in the neighborhood; if $0 < PR_{i,k,j'} < T_a$, user U_i considers LBS message $m_{k,j'}$ as a useless but not redundant one without further authentication and transmit it to other vehicles in her/his neighborhood; otherwise, U_i decides LBS message $m_{k,j'}$ as a useful one and verifies the signature by performing the algorithm $\Lambda.Verify(pk_{\Lambda,k}, C_{k,j'}, \sigma_{k,j'})$. If it fails, U_i discards LBS message $m_{k,j'}$ and stop it from further transmission; otherwise, U_i accepts and recovers $m_{k,j'}$ by performing the ABE decryption algorithm $m_{k,j'} = ABE.Dec(PPAR_{ABE}, SK_{ABE,i}, C_{k,j'})$. Finally, U_i gives a rating $R_{i,k,j'}$ on LBS message $m_{k,j'}$ and adds the tuple $(U_i, U_k, j^{'}, C_{Loc_{k,j',x}}, C_{Loc_{k,j',y}}, C_{t_{k,j'}}, C_{R_{i,k,j'}})$ into its table T_i by performing Step 2) in the algorithm **LBS Message Generation**.

Remark:(Aggregated LBS bundles) It is noted that the techniques of aggregate signature and multi-signature [6] can be exploited by each vehicular user to compress all her/his accepted LBS messages originally generated and signed by different users, into a single bundle (i.e. the useless but not duplicate LBS messages can also be aggregated into a single bundle in the same way). Then, the specific vehicular user rates on both the accepted and the useless but not redundant LBS bundle, and each $RSU_s(s = 1, 2, \cdots, n_d)$ can predict the ratings on the bundles for other users based on their similarities in Eqn. (25). The communication cost on each vehicular user would be further reduced (i.e. please refer to Sec. 7.2 for performance evaluation).

6 Security Proof

In this section, we firstly give the formal security proof of our proposed multi-key secure outsourced computation scheme MSOC. Then, based on the primitive of MSOC, we elaborate that our proposed lightweight privacy-preserving authentication protocol LPPA for location-based services in VANETs achieves the security goals.

6.1 Security for the Proposed MSOC

Before giving the security proof, we present the correctness of our MSOC that serves the primitive of LPPA. In Eqn. (7), by exploiting Chinese Remainder Theorem and Euler's Theorem [29], we have

$$\begin{aligned} C_F^{CSP,1} &= C_F^{bld} \ mod \ N \\ &= r^{deg_F - deg_j} \sum_{j=1}^{K} a_j \prod_{l=1}^{n} (C_{l,SER}^{"})^{t_{l,j}} \ mod \ N \\ &= r^{deg_F} \sum_{j=1}^{K} a_j \prod_{l=1}^{n} (p^{-1} p m_{l,q}^{q} + q^{-1} m_{l,p}^{p} + r_{i,csp} N)^{t_{l,j}} \\ &\quad mod \ N \\ &= r^{deg_F} \sum_{j=1}^{K} a_j (p^{-1} p (\prod_{l=1}^{n} m_l^{t_{l,j}})_q^q + q^{-1} q (\prod_{l=1}^{n} m_l^{t_{l,j}})_p^p) \\ &\quad mod \ N \\ &= r^{deg_F} (p^{-1} p (\sum_{j=1}^{K} a_j \prod_{l=1}^{n} m_l^{t_{l,j}})_q^q + q^{-1} q (\sum_{j=1}^{K} a_j \prod_{l=1}^{n} m_l^{t_{l,j}})_p^p) \\ &\quad mod \ N \\ &= r^{deg_F} \sum_{j=1}^{K} a_j \prod_{l=1}^{n} m_l^{t_{l,j}} = r^{deg_F} F(m_1, m_2, \cdots, m_n). \end{aligned}$$

Therefore, the authorized receiver possessing the secret key $sk_{f,rec}$ can successfully recover $F(m_1, m_2, \cdots, m_n) = r^{-deg_F} C_F^{CSP,1}$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2923156, IEEE Transactions on Information Forensics and Security

We firstly give the security proof of the data privacy of the subset of uncorrupted senders $N_{Sen} \backslash T$ in our proposed MSOC against the collusion attack between the CSP, the subset of corrupted senders T and malicious receivers. We also take input privacy to detail the security proof and the proofs for the output privacy and the collusion with the cloud server can be similarly derived.

Theorem 1: (**Data Privacy for MSOC**) Let $\mathcal{A}$ be a malicious adversary defeating the CCA2 security for data privacy of our proposed MSOC with a nonnegligible advantage defined as $\epsilon^{',ploy(\lambda)}$, where $poly(\lambda)$ refers to the total number of queries made to the oracles and λ is the security parameter. There exists a simulator $\mathcal{B}$ who can use $\mathcal{A}$ to invert the one-way trapdoor permutation f with the nonnegligible probability ϵ that:

$$\epsilon \geq \epsilon^{',poly(\lambda)} - \frac{poly(\lambda)}{2^{\lambda-1}}. \quad (28)$$

Proof: We take input privacy to detail the security proof. Intuitively, although the adversary $\mathcal{A}$ considered as the collusion between the CSP, the malicious receiver and a subset of corrupted senders holds secret key $sk_{f,csp}$ that can be used to compute $N_i = f^{-1}_{sk_{f,csp}}(C_{i,csp})$ and $C^{'}_{i,i'} = C_{i,i'} \ mod \ N_i = r_i m_{i,i'} \ mod \ N_i$, the input $m_{i,i'}$ cannot be derived without the knowledge of r_i encrypted in $C_{i,ser} = f_{pk_{f,ser}}(r_i)$. Therefore, we can reduce the CCA2 security for input privacy to the inverse of one-way trapdoor permutation f without secret key $sk_{f,ser}$ and the proof is given by contradiction. In the initialization phase, the system performs $(f, f^{-1}) \leftarrow \mathcal{G}(1^\lambda), y_i = f_{pk_{f,ser}}(r_i)$ and the simulator $\mathcal{B}$ tries to solve $r_i = f^{-1}_{sk_{f,ser}}(y_i)$. The adversary $\mathcal{A}$ is given the public parameter PPR, the secret keys $sk_{f,csp}, sk_{f,rec}$ of the corrupted CSP and the corrupted receiver and all the temporary secret key pvk_{CSP}, pvk_i of the corrupted CSP and all corrupted senders $Sen_i \in T$. There are two oracles, namely $\mathcal{O}^{H_0}$ and $\mathcal{O}^{Dec}$. $\mathcal{B}$ can perform the simulations by answering the queries from the adversary as follows. For the collusion between the CSP, malicious receivers and a subset of corrupted senders, we mainly focus on the ciphertext components $C_{i,ser}, C_{i,i'}, C^{'}_{i,ser}$ in C_{Sen_i}.

$\mathcal{O}^{H_0}$ **Query**. If a query $r_i \parallel C_{i,1} \parallel \cdots \parallel C_{i,n_i}$ to $\mathcal{O}^{H_0}$ satisfies $f(r_i) = y_i$, then $\mathcal{B}$ outputs r_i and halts; otherwise, it returns a random element $Str_0 \in_R \{0,1\}^{2\lambda}$ as the response to the adversary and remains the triple $(r_i, C_{i,1} \parallel \cdots \parallel C_{i,n_i}, Str_0)$ in the H_0-list.

$\mathcal{O}^{Dec}$ **Query**. To answer the query $s^{'} \parallel d^{'} \parallel h^{'}$ to $\mathcal{O}^{Dec}$ where $d^{'} = \cup^{n_i}_{i'=1} C_{i,i'}$, the simulator $\mathcal{B}$ firstly checks if there exists a triple $(r_i, d^{'}, h^{'})$ in the H_0-list. If it does, the simulator $\mathcal{B}$ further checks whether $s^{'} = f_{pk_{f,ser}}(r_i)$ holds. If not, the simulator $\mathcal{B}$ returns invalid; otherwise it computes and returns $C^{'}_{i,i'} = r_i^{-1} d^{'}$ to the adversary $\mathcal{A}$ and $\mathcal{A}$ computes $m_i = C^{'}_i \ mod \ N_i$.

Then, the adversary $\mathcal{A}$ submits two challenge plaintexts $m_{i,i',0}, m_{i,i',1}$ associated to an uncorrupted sender $Sen_i \in N_{Sen} \setminus T$ of the same size $|m_{i,i',0}| = |m_{i,i',1}| = 2\lambda$, and the simulator $\mathcal{B}$ randomly selects $\beta \in \{0,1\}$ and returns the encryption of $m_{i,i',\beta}$ as the challenge ciphertext $c^*_{i,i'}$. After receiving $c^*_{i,i'}$, $\mathcal{A}$ can continue to make queries to the oracles $\mathcal{O}^{H_0}$ and $\mathcal{O}^{Dec}$ with the restriction that $c^*_{i,i'}$ cannot be queried to the decryption oracle $\mathcal{O}^{Dec}$ in the adaptive query phase.

To explain the interaction perfectly simulates the real environment of the adversary $\mathcal{A}$ running with its oracles, we study the following events. Let S be the event that for some ciphertext $s^{'} \parallel d^{'} \parallel h^{'}$, $\mathcal{A}$ made some query $r_i \parallel d^{'}$ to the oracle $\mathcal{O}^{H_0}$ satisfying $f(r_i) = s^{'}$. Then, we further let R be the event that $\mathcal{A}$ made some query $s^{'} \parallel d^{'} \parallel h^{'}$ to the decryption oracle $\mathcal{O}^{Dec}$ where $h^{'} = H_0(r_i \parallel d^{'})$ holds without making any query $(f^{-1}_{sk_{f,ser}}(s^{'}) \parallel d^{'})$ to the H_0-oracle $\mathcal{O}^{H_0}$. Let $poly(\lambda)$ be the total number of oracle queries made by the adversary $\mathcal{A}$. Then, we can conclude that

$$\begin{aligned}
&Pr[\mathcal{A}^{Suc}] \\
&= Pr[\mathcal{A}^{Suc}|R]Pr[R] + Pr[\mathcal{A}^{Suc}|\bar{R} \wedge S]Pr[\bar{R} \wedge S] \\
&+ Pr[\mathcal{A}^{Suc}|\bar{R} \wedge \bar{S}]Pr[\bar{R} \wedge \bar{S}] \\
&\leq poly(\lambda)2^{-\lambda} + Pr[S] + \frac{1}{2},
\end{aligned}$$

since $Pr[R] \leq \frac{poly(\lambda)}{2^\lambda}$ and $Pr[\mathcal{A}^{Suc}|\bar{R} \wedge \bar{S}] = \frac{1}{2}$ can be straightforwardly derived. Finally, it is observed that the probability of simulator $\mathcal{B}$ to fail in behaving like the adversary $\mathcal{A}$ in inverting the one-way trapdoor permutation f can be bounded by $Pr[R]$. Therefore,

$$\epsilon \geq \epsilon^{',poly(\lambda)} - \frac{poly(\lambda)}{2^{\lambda-1}},$$

which is also non-negligible. Therefore, theorem 1 holds. □

Theorem 2: (**Security for the Whole Protocol**) The proposed MSOC securely implements the functionality Fun, namely for every real world adversary $\mathcal{A}$, there exists an ideal world adversary $\mathcal{S}$ with access to $\mathcal{A}$ in a black-box manner such that for all input vectors $\vec{m}$, we have $Ideal_{Fun,\mathcal{S}}(\vec{m}) \approx_c Real_{MSOC,\mathcal{A}}(\vec{m})$.

Proof: Based on the data privacy (indistinguishability) proved in Theorem 1, we prove this theorem when the server is corrupted via a series of hybrid games, by using an ideal/real paradigm. The proofs for other cases of a corrupted CSP, corrupted sender, corrupted receiver and their collusion can be analogously derived.

Game 0. This is the real world execution of our proposed MSOC.

Game 1. Instead of executing **MSOC.Dec** where the honest receiver uses its secret key, we run the simulator $\mathcal{S}_{MSOC.Dec}$ interacting with the adversary $\mathcal{A}$. Owing to the data privacy (i.e. we mean output privacy here) of our proposed MSOC, if the ideal decryption functionality is correctly emulated, the joint output is computationally indistinguishable in a real world execution of our proposed MSOC with the adversary $\mathcal{A}$, and in a ideal world execution with the adversary $\mathcal{S}_{MSOC.Dec}$.

Game 2. In this game, by replacing computing $\hat{y} = MSOC.Dec(PPR, \hat{sk}_{f,rec}, C_F)$, the joint output is