IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 29, 2020, accepted May 13, 2020, date of publication May 18, 2020, date of current version June 2, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2995240

Analysis of Error Impact for Batch Handover Authentication Protocols in Mobile Wireless Networks

YOUNSOO PARK AND HO-HYUN PARK
School of Electrical and Electronics Engineering, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Ho-Hyun Park (hohyun@cau.ac.kr)

This work was supported in part by the Ministry of Education, through the Face Recognition and Searching Robust in Pose, Illumination, and Expression Utilizing Video Big Data Project, under Grant NRF-2016R1D1A1B03933895, and in part by the Ministry of Trade, Industry, and Energy, through the HRD Program for Industrial Convergence of Wearable Smart Devices Project, under Grant P0002397.

ABSTRACT The handover authentication protocol controls the access of multiple access points in a mobile wireless network (MWN). PairHand is a handover authentication protocol that uses identity-based public key cryptography based on bilinear pairing and provides batch verification to efficiently control multiple accesses. However, there may be errors in the messages transmitted via a wireless channel; such errors can affect the batch verification process. Previous studies involving PairHand have calculated the verification delay without considering errors. We have classified the message verification methods into two types: individual verification, in which the messages are individually verified and batch verification, in which the messages are verified all at once. We formulate the verification delay as a function of the bit error probability and visualize it. In our graph, the batch verification delay rapidly increases as the bit error probability increases and then forms an intersection with the individual verification delay. This intersection can be a guideline for whether to use batch verification or individual verification, depending on the bit error rate. We also classify the previously proposed handover authentication protocols into pairing-based protocols and pairing-free protocols and analyze the computation and communication costs of each. Based on the analysis results, we propose a mini-batch method to reduce the batch verification delay in an error-prone MWN.

INDEX TERMS Batch handover authentication, batch verification delay, bilinear pairing, elliptic curve cryptography, handover authentication protocol, mini-batch verification, mobile wireless network.

I. INTRODUCTION

Advancement in mobile communication technologies such as 5G has entailed a dramatic improvement in the speed of wireless communication networks [1]. A mobile wireless network (MWN) [2], as shown in Fig. 1, is composed of an authentication server (AS) that is a major entity, a number of access points (APs) that are connected to the AS, and a number of mobile nodes (MNs) that are connected to an AP. An MN first registers its own real identity to the AS, and then a pseudo-identity is issued from the AS. Thereafter, the MN connects to the AS using the pseudo-identity. In order for an MN to be connected to the AS, it should be verified as a legitimate user by a nearby AP. The AP verifies the validity of the MN and allows only authorized MNs to connect to the AS. Handover authentication protocol provides seamless roaming securely when an MN is moving from the currently connected AP to another AP.

This protocol has to preserve privacy and guarantee user anonymity and untraceability. In addition, it should provide not only subscription validation between an AP and a MN, but also mutual authentication and session key agreement [3]. Because the wireless communication environment is vulnerable to security breaches, it should have resistance to all known attacks [4]. Therefore, the handover authentication protocol should be efficient while guaranteeing security by using cryptographic technology. To accomplish this, many protocols have been proposed over the past few decades [5]–[9]. However, most of them are insecure or inefficient. PairHand, proposed by He *et al.* [10], is a handover authentication

The associate editor coordinating the review of this manuscript and approving it for publication was Gautam Srivastava.

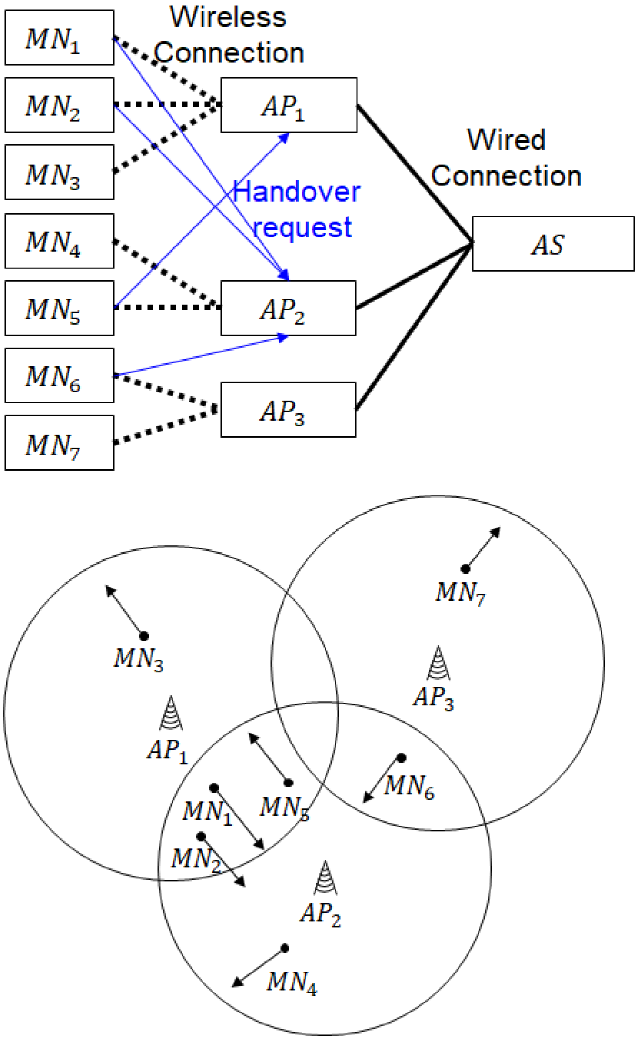


FIGURE 1. The environment of a mobile wireless network (MWN), showing the authentication server (AS), access points (APs), and mobile nodes (MNs).

protocol that uses identity-based public key cryptography (ID-based PKC) based on bilinear pairing. It provides privacy preservation and user anonymity and minimizes communication and computation overhead.

PairHand guarantees appropriate security levels with minimum signature length by utilizing elliptic curve cryptography (ECC). Moreover, it minimizes handshakes between server and client, thereby improving the efficiency dramatically compared to other handover authentication protocols [5]–[7]. It also provides not only the individual authentication that validates messages individually but also the batch authentication [11]–[14] that validates multiple messages all together in order to decrease the burden on an AP that handles lots of handover requests.

He *et al.* [10], however, pointed out that their protocol suffers from the key compromised problem [15]. Also, they proposed a simple improved protocol. But, Yeo *et al.* [16] proved that the protocol proposed by He *et al.* [15] still suffers from the key compromised problem and is therefore insecure. Subsequently, He *et al.* [17], Wang and Hu [18], and Tsai *et al.* [19] proposed improved protocols in which security features are supplemented by inserting random numbers into request messages. The protocols proposed by them are pairing-based handover authentication protocols that use bilinear pairing on both the AP and MN side. However, they have a problem in that a burden can be imposed on an MN with limited resources because the protocol uses heavy bilinear pairing operations and map-to-point operations [4]. Since then, Islam and Khan [4], Wang *et al.* [20], and Chaudhry *et al.* [21] have proposed pairing-free handover authentication protocols that do not use bilinear pairing during on the authentication process. And He *et al.* [22] have analyzed the communication cost and security of various protocols proposed thus far to seek further enhancement.

However, previous studies [23]–[31] have not considered errors that may occur in the wireless communication channels between APs and MNs. Since multiple signals share the same propagation medium over the wireless communication channel, severe interference occurs between the various signals. As a result, errors are more likely to occur in a wireless communication channel than in a wired communication channel [32]. If errors occur frequently due to the geographic environment at a specific position in the MWN or an attacker takes part in the community of MNs and inserts errors into the verification message continuously [21], verification failures of APs will occur frequently. This means that the burden of message verification on the APs becomes heavier. For example, AP2 in Fig. 1 receives handover request messages from several MNs. Individual verification of these request messages does not affect the authentication of the remaining non-error messages even if some messages have errors. On the other hand, in a batch verification, an error in one message affects the verification of messages that have no errors because the error causes a verification failure for the entire (batched) message. This means that a batch verification requires additional operations if there is a high probability of errors in the handover request messages. When the probability of errors is over a certain level, the delay in batch verification may exceed the sum of delays for individual verifications because of frequently occurring errors.

Until now, the proposed handover authentication protocols have focused only on security and efficiency, so there has been no prior investigation of the impact of errors on computation cost. In this study, we aim to analyze the performance of batch verification through an analysis of computation costs in the presence of errors that are likely to occur in wireless channels. To this end, we measure the processing times of the He *et al.* [10]'s protocol and derive the individual verification delay as a function of the processing time and graph it. And we derive the batch verification delay as a function of the bit error probability and graph it. As the bit error probability increases, the individual verification delay remains unchanged, but the batch verification delay increases. Next, we find the point which the intersection of batch verification graph and individual verification graph meet. Also, we evaluate the communication and computation cost of the handover authentication protocols previously proposed.

From the comparison, we find that the point at which the intersection of batch verification graph and individual verification graph appears varies by protocol. In the case where the intersection is formed when the error probability is relatively low, the delay increase is large even if the error probability is slightly increased. On the other hand, protocols for which intersections are formed when the error probability is high have a small increase in delay. This comparison shows that the lower the communication cost and the faster the batch verification is compared to the individual verification, the less influenced the protocol is by error rate. In other words, both communication cost and computation cost should be minimized in order to minimize the influence of errors in batch verification. We propose a mini-batch verification method to reduce the impact of errors based on these analysis results. We also demonstrate that the application of this method to previous studies [4], [10], [17], [18], [20] can mitigate the impact of errors.

Our main contributions are as follows:

- We analyze the performance (communication and computations costs) of various handover authentication protocols [4], [10], [17], [18], [20] operating in wireless communication environments containing errors.
- We investigate batch verification through formulas and simulations and find that the batch verification delay can exceed the individual verification delay owing to the impact of errors.
- We propose a mini-batch method to mitigate the impact of errors in batch verification and demonstrate its applicability to previously proposed protocols.

The rest of this paper is organized as follows. Section II reviews the protocol proposed by He *et al.* [10] that provides preliminary knowledge for a better understanding of this study. A method for estimating the computation cost of verification methods is analyzed in Section III. Section IV evaluates and compares the computation cost and communication cost of the protocols previously proposed. In Section V, a mini-batch verification method is proposed to mitigate error impact based on the analysis results presented in Sections III and IV. The conclusions are summarized in Section VI.

II. PRELIMINARIES

A. ELLIPTIC CURVE AND BILINEAR MAPS

When p is a large prime number, $E\left(F_p\right)$ is a point on an elliptic curve that is defined as $y^2 \bmod q = x^3+ax+b \bmod q$ $(a, \in F_p)$ in the finite field F_p. All points on $E\left(F_p\right)$ including the point at infinite O form a cyclic additive group G. Assuming that $P\left(x_1, y_1\right)$ and $Q\left(x_2, y_2\right)$ are points on $E\left(F_p\right)$, and $R\left(x_3, y_3\right)$ is a coordinate of the point at which the line (if $P = Q$, then the tangent line) that crosses P and Q meets a elliptic curved line, the coordinates of $P+Q$ become $\left(x_3, -y_3\right)$. Here, operation $kP = P+\mathrm{P}+P+\ldots+P$ (k times) is referred to as the scalar point multiplication in the group G.

When G is a cyclic additive group of order q and G_T is a cyclic multiplicative group of the same order q (where q is a large prime), the bilinear map $\hat{e} : G \times G \rightarrow G_T$ satisfies the following properties [33]:

(1) Bilinearity: $\hat{e}\left(aP, bQ\right) = \hat{e}\left(P, Q\right)^{ab}$, where P, $Q \in G$ and a, $b \in Z_q^*$.

(2) Non-degeneracy: $\hat{e}\left(P, Q\right) \neq 1_{G_T}$.

(3) Computability: There exists an efficient algorithm to compute $\hat{e}\left(P, Q\right)$ for some P, $Q \in G$.

The advantage of using ECC is that it is computationally very difficult to infer abP when P, aP and bP (computational Diffie–Hellman) are given. Furthermore, the advantage of using bilinear pairing is that $\hat{e}\left(P, P\right)^{ab}$ is exceedingly difficult to infer when P, aP, bP (bilinear Diffie–Hellman) are given [33]. Notably, PairHand [10] uses the aforementioned properties to validate the handover request messages. It also reduces the number of handshakes between AP and MN over previous studies requiring at least three rounds of handshake [5]–[7]. These advantages have led to PairHand attracting the attention of researchers [13]–[24].

B. ERROR PROBABILITY FOR A TRANSMITTED MESSAGE

In data transmission, a bit error ratio (BER) p_e represents the probability of receiving an erroneous bit. A packet error ratio (PER) p_p represents the probability of receiving an erroneous packet. When the length of one packet is m bits, p_p is determined by p_e as shown in (1).

$$p_p = 1 - \left(1 - p_e\right)^m \tag{1}$$

C. REVIEW OF PairHand

Previously proposed handover protocols were implemented using chameleon hashing [5], symmetric encryption [8], or simple hash function [9]. Additionally, even if ECC is used, at least a three-way handshake is required [6], [7]. However, PairHand, which was proposed by He *et al.* [10], requires only a two-way handshake between MN and AP for mutual authentication and key establishment [23]. In this section, we will review the three phases of PairHand. Thus, we will be able to understand how PairHand could reduce the number of handshakes.

In the initialization phase, the AS chooses $s \in Z_q^*$ as a master key and calculates P of G and $P_{pub} = sP$. Afterwards, it chooses a map-to-point hash H_1 and a cryptographic hash H_2, where H_1: $\{0, 1\}^* \rightarrow G$ and H_2: $\{0, 1\}^* \rightarrow Z_q^*$. Then, the AS broadcasts the public system parameters $params = \left\{G, G_T, q, P, P_{pub}, H_1, H_2\right\}$ to the MWN and stores s securely.

In the registration phase, an AP that received the system parameters from the AS sends its own identity ID_{AP} to AS via a secure channel. Then after checking its validity, the AS calculates $\left(H_1\left(ID_{AP}\right), sH_1\left(ID_{AP}\right)\right)$ and sends it to the AP to complete the registration procedure. Likewise, an MN sends its own real identity ID_i to the AS via a secure channel. After verifying the validity of the MN, the AS creates the pseudo-ID and secret key pair, $PID = \left\{pid_j, sH_1\left(pid_j\right)\right\}$

TABLE 1. Summary of various protocols.

PROTOCOL	BATCH	ECC	PROTOCOL	BATCH	ECC
He *et al.* [2]'s protocol	O	O	He *et al.* [17]'s	O	O
Yang *et al.* [3]'s protocol	X	O	Wang *et al.* [18]'s	O	O
Islam *et al.*[4]'s protocol	O	O	Tasai *et al.* [19]'s protocol	O	O
Choi *et al.* [5]'s protocol	X	X	Wang *et al.* [20]'s protocol	O	O
Yang *et al.* [6]'s protocol	X	O	Chaudhry *et al.* [21]'s protocol	X	O
He *et al.* [7]'s protocol	X	O	He *et al.* [23]'s protocol	O	O
Hsiang *et al.* [8]'s protocol	X	X	Li *et al.* [24]'s protocol	X	O
Liao *et al.* [9]'s protocol	X	X	Gao *et al.* [26]'s protocol	X	O
He *et al.* [10]'s protocol	O	O	Gao *et al.* [27]'s protocol	O	O
Yoon *et al.* [11]'s protocol	O	O	Gao *et al.* [28]'s protocol	X	O
Zhang *et al.* [12]'s protocol	O	O	Lei *et al.* [29]'s protocol	X	O
Bayat *et al.* [13]'s protocol	O	O	Ma *et al.* [30]'s protocol	X	O
Limbasiya *et al.* [14]'s protocol	O	X	Xue *et al.* [31]'s protocol	X	X
Yeo *et al.* [15]'s protocol	O	O			

($j = 1, 2, 3, \ldots$) and sends it to the MN to complete the registration procedure. After the registration procedure is completed, the AS does not participate in handover authentication, and therefore, the burden of AS is dramatically reduced.

In the handover authentication phase, as shown in Fig. 2, MN i selects the $\left(pid_i, sH_1\left(pid_i\right)\right)$ pair that was not used and calculates $\sigma_i = H_2\left(\mathcal{M}_i\right) \cdot sH_1\left(pid_i\right)$, where $\mathcal{M}_i = pid_i||ID_{AP}||ts$, and ts is a timestamp. Subsequently, it sends a handover request message $\langle\mathcal{M}_i, \sigma_i\rangle$ to an AP and calculates its own symmetric key $K_{i-A} = \hat{e}\left(sH_1\left(pid_i\right), H_1\left(ID_{AP}\right)\right)$. After receiving a request message from MN i, the AP checks a freshness of ts. Then, the AP calculates $H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right)$ to verify that $\hat{e}\left(\sigma_i, P\right)$ is identical to $\hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P_{pub}\right)$, as shown below. The following equation shows that PairHand does not need MN's secret key for authentication.

$$\begin{aligned}\hat{e}\left(\sigma\mathcal{M}a_i, P\right) &= \hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot sH_1\left(pid_i\right), P\right)\\ &= \hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P\right)^s\\ \hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P_{pub}\right)&\\ &= \hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), sP\right)\\ &= \hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P\right)^s\end{aligned}$$

If two pairing results match, a symmetric key of AP $K_{A-i} = \hat{e}\left(H_1\left(pid_i\right), sH_1\left(ID_{AP}\right)\right)$ is calculated; otherwise, authentication is rejected. Next, after calculating an authentication message $Aut = H_2\left(K_{A-i}||pid_i||sH_1\left(ID_{AP}\right)\right)$, the pid_i, ID_{AP}, and Aut are sent to MN i. Since the message contains symmetric key information, additional handshake for symmetric key exchange [5], [9] is not required for AP and MN.

After receiving the authentication message from the AP, MN i calculates a verification message $Ver = H_2\left(K_{i-A}||pid_i||ID_{AP}\right)$ using the symmetric key K_{i-A} that it owns and checks a validity of the authentication message by verifying whether Aut and Ver are identical. Consequently, PairHand does not need to send or verify certificates as it does in traditional public key cryptography systems.

D. BATCH VERIFICATION OF PairHand

In the MWN environment shown in Fig. 1, a number of handover request messages are transmitted to one AP. To efficiently process these, the protocol by He *et al.* [10] provides a batch authentication. It is possible to verify the handover request messages that were received from n MNs $\langle\mathcal{M}_1, \sigma_1\rangle, \langle\mathcal{M}_2, \sigma_2\rangle, \ldots, \langle\mathcal{M}_n, \sigma_n\rangle$ at once using $\hat{e}\left(\sum_{i=1}^{n}\sigma_i, P\right) \stackrel{?}{=} \hat{e}\left(\sum_{i=1}^{n}H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P_{pub}\right)$. Table 1 summarizes whether the protocol employs the ECC and supports batch verification. Batch verification can dramatically reduce the computational cost, so it can be widely used for practical handover authentication protocols.

As described in section II.C, PairHand [10] efficiently minimizes the number of handshakes, reduces the burden of AS and AP, and does not require the transmission of certificates. This protocol also offers an adequate level of security, in addition to providing efficient batch verification. These are the advantages of PairHand that we focus on, and we will analyze its performance in section III. In section IV, we will evaluate the additional batch verification protocols developed by He *et al.* [17], Wang and Hu [18], Islam and Khan [4] and Wang *et al.* [20], which are known to be excellent among the protocols listed in Table 1.

III. ANALYSIS OF VERIFICATION DELAY

In this section, we analyze the performance of the protocol proposed by He *et al.* [10]. In Section IV, we classify various handover authentication protocols into pairing-based protocols and pairing-free protocols and evaluate and compare communication and computation costs. Table 2 lists the average processing time and notation of super singular elliptic curve and non-super singular elliptic curve protocols described in Sections III and IV. The pairing-based protocols [10], [17], [18] all use a super singular elliptic curve, whereas the pairing-free protocols [4], [20] all use a non-super singular elliptic curve. The operations that pairing-based protocols typically use include scalar point multiplication T_{sm}^{G}, point addition T_{pa}^{G}, map-to-point hash function in G T_{M2P}^{G}, bilinear pairing T_{bp}^{G}, and cryptographic hash T_h. On the other hand, pairing-free protocols do not use bilinear

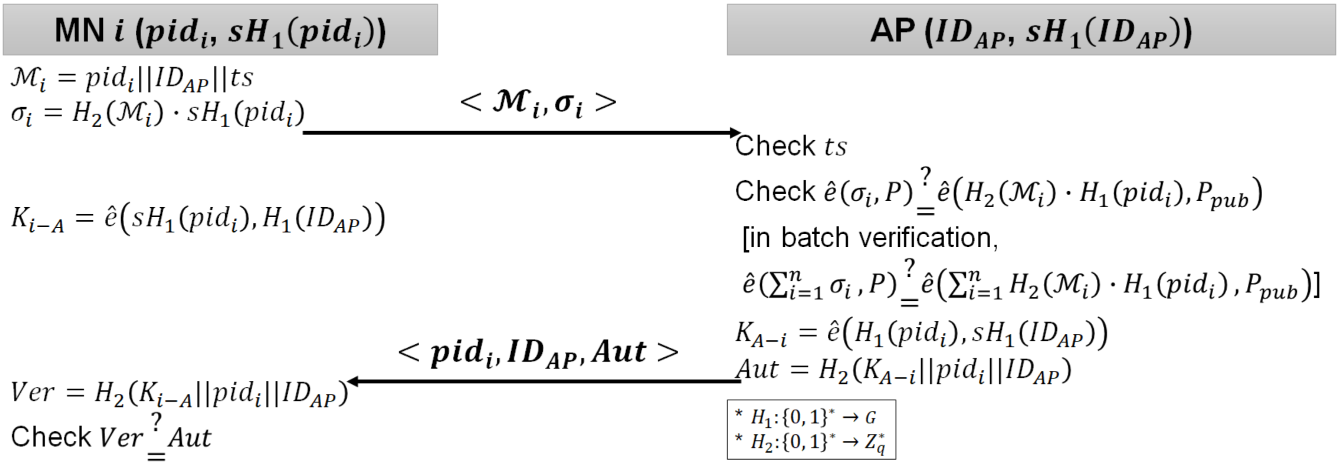

MN i $(pid_i, sH_1(pid_i))$
AP $(ID_{AP}, sH_1(ID_{AP}))$
$\mathcal{M}_i = pid_i||ID_{AP}||ts$
$\sigma_i = H_2(\mathcal{M}_i) \cdot sH_1(pid_i)$
$< \mathcal{M}_i, \sigma_i >$
Check ts
Check $\hat{e}(\sigma_i, P) \overset{?}{=} \hat{e}(H_2(\mathcal{M}_i) \cdot H_1(pid_i), P_{pub})$
[in batch verification,
$\hat{e}(\sum_{i=1}^{n} \sigma_i, P) \overset{?}{=} \hat{e}(\sum_{i=1}^{n} H_2(\mathcal{M}_i) \cdot H_1(pid_i), P_{pub})]$
$K_{i-A} = \hat{e}(sH_1(pid_i), H_1(ID_{AP}))$
$K_{A-i} = \hat{e}(H_1(pid_i), sH_1(ID_{AP}))$
$Aut = H_2(K_{A-i}||pid_i||ID_{AP})$
$< pid_i, ID_{AP}, Aut >$
$Ver = H_2(K_{i-A}||pid_i||ID_{AP})$
Check $Ver \overset{?}{=} Aut$
* $H_1: \{0,1\}^* \to G$
* $H_2: \{0,1\}^* \to Z_q^*$

FIGURE 2. Handover authentication phase of He *et al.* [10]'s protocol.

TABLE 2. Notation and average processing time.

NOTATION	DETAILS	AP'S PROCESSING TIME (IN MS)
T_{sm}^{G}	Average processing time of a scalar point multiplication operation in G	1.5449
T_{pa}^{G}	Average processing time of a point addition operation in G	0.0090
T_{M2P}^{G}	Average processing time of a map-to-point hash function in G	4.0854
$T_{sm}^{\bar{G}}$	Average processing time of a scalar point multiplication operation in $\bar{G}$	0.3686
$T_{pa}^{\bar{G}}$	Average processing time of a point addition operation in $\bar{G}$	0.0030
T_{bp}^{G}	Average processing time of a bilinear pairing operation	3.4410
T_h	Average processing time of a cryptographic hash operation	0.0006

pairing and map-to-point hash function in $\bar{G}$, therefore; only scalar point multiplication $T_{sm}^{\bar{G}}$ and point addition $T_{pa}^{\bar{G}}$ operations are described. The super singular elliptic curve $E(F_p)$ is defined on the finite field F_p and is implemented by using the Tate pairing [34] defined on G. The group G with order q is a point on $E(F_p)$, and both p and q are prime numbers with 512 and 160 bits, respectively. The non-super singular elliptic curve $\bar{E}(F_{\bar{p}})$ is defined on the finite field $F_{\bar{p}}$ and the group $\bar{G}$ with order $\bar{q}$ is also a point on $\bar{E}(F_{\bar{p}})$. Both $\bar{p}$ and $\bar{q}$ are prime numbers of 160 bits. Each processing time listed in Table 2 represents the average time required for performing each operation 10,000 times. Simple operations except the hash and elliptic curve operations are skipped. In addition, an operation time of the MN is not described because it is irrelevant to the verification time of the AP. MIRACL [35] lib with i5-4690, Ubuntu-16.04 32-bit is used for measurement.

A. ANALYSIS OF VERIFICATION DELAYS ASSUMING NO ERRORS

An individual verification indicates a method to individually verify the n request messages in an AP. A batch verification refers to a method to process the n request messages at once. In the case of individually verifying n messages with the protocol of He *et al.* [10], the individual verification delay d_{ind} is expressed as shown in (2).

$$\begin{aligned} d_{ind} &= n\left(T_{sm}^{G} + T_{M2P}^{G} + 3T_{bp}^{G} + 2T_h\right) \\ &\approx n\,(15.9545)\ ms \end{aligned} \tag{2}$$

In batch verification, the $2n$ bilinear pairing operations in individual verification are replaced by $2n$ point addition operations and 2 bilinear pairing operations, resulting in a batch verification delay d_{bat} expressed as shown in (3).

$$\begin{aligned} d_{bat} &= n\left(T_{sm}^{G} + 2T_{pa}^{G} + T_{M2P}^{G} + T_{bp}^{G} + 2T_h\right) + 2T_{bp}^{G} \\ &\approx n\,(9.0905) + 6.8820\ ms \end{aligned} \tag{3}$$

If there is an error in a message sent by multiple MNs in Fig. 1, the signature check will fail and consequently the whole message verification will fail. Failure in verification, in turn, causes a rejection of authentications without performing a calculation of the symmetric key K_{A-i} and authentication message Aut. Thus, the computation times for symmetric key and authentication message are not affected by errors. Accordingly, (2) and (3) can be expressed as (4) by grouping into a signature check term that is affected by errors and a key computation term that is not affected by errors. In (4), $d_{ind-chk}$ and $d_{bat-chk}$ denote the delays required for signature check and d_{key} denotes a delay for calculating K_{A-i} and Aut.

$$\begin{aligned} d_{ind} &= d_{ind-chk} + d_{key} \\ &= n\left(T_{sm}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} + T_h\right) + n\left(T_{bp}^{G} + T_h\right) \\ &\approx n\,(12.5129) + n\,(3.4416)\ ms \end{aligned}$$

$$\begin{aligned} d_{bat} &= d_{bat-chk} + d_{key} \\ &= n\left(T_{sm}^{G} + 2T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 2T_{bp}^{G} \\ &\quad + n\left(T_{bp}^{G} + T_h\right) \\ &\approx n\,(5.6489) + 6.8820 + n\,(3.4416)\, ms \end{aligned} \tag{4}$$

Based on (2)–(4), individual verifications and a batch verification require 1.5954 s and 0.9159 s, respectively, to process 100 messages.

B. ANALYSIS OF VERIFICATION DELAY CONSIDERING ERRORS

If a request message contains errors, individual verification can reject only the messages with errors because individual verification inspects each message individually, whereas even one bit error in a batch verification causes a verification failure of the entire batch of messages. Therefore, the messages have to be requested again to all MNs or individual verifications must be performed again to seek the message containing errors. A message re-transmission request to all MNs incurs a computation cost as well as a communication cost, thus it is not appropriate because of a longer delay and a burden to the MN. In this regard, when a failure occurs in a batch verification due to errors, it is relevant to check the messages for errors using individual verification and to reject only the message(s) containing errors. When a batch verification fails due to errors, a delay $d_{bat-fail}$ is represented as (5).

$$\begin{aligned} d_{bat-fail} &= d_{ind} + d_{bat} - d_{key} \\ &= d_{ind-chk} + d_{bat-chk} + d_{key} \\ &= n\left(2T_{sm}^{G} + 2T_{pa}^{G} + 2T_{M2P}^{G} + 2T_{bp}^{G} + 2T_h\right) \\ &\quad + 2T_{bp}^{G} + n\left(T_{bp} + T_h\right) \\ &\approx n\,(18.1618) + 6.8820 \\ &\quad + n\,(3.4416)\, ms \end{aligned} \tag{5}$$

To calculate the computation cost caused by an error during the verification process, a probability of error occurrence in a request message must be calculated. If the length of a request message is m bits and the probability of error occurrence in each bit is p_e (p_e is independent), the probability P_E that an error occurs in one or more bits within the n request messages can be expressed as (6). In (6), an increase in p_e leads to an increase in P_E.

$$\begin{aligned} P_E &= 1 - \left((1-p_e)^m\right)^n \\ &= 1 - (1-p_e)^{mn} \end{aligned} \tag{6}$$

$E[d]$, the expectation of verification delay according to P_E, is expressed as (7).

$$\begin{aligned} E[d] &= d_{bat-chk} \times (1-P_E) \\ &\quad + (d_{bat-chk} + d_{ind-chk}) \times P_E + d_{key} \\ &= d_{bat-chk} + d_{ind-chk} \times P_E + d_{key} \end{aligned} \tag{7}$$

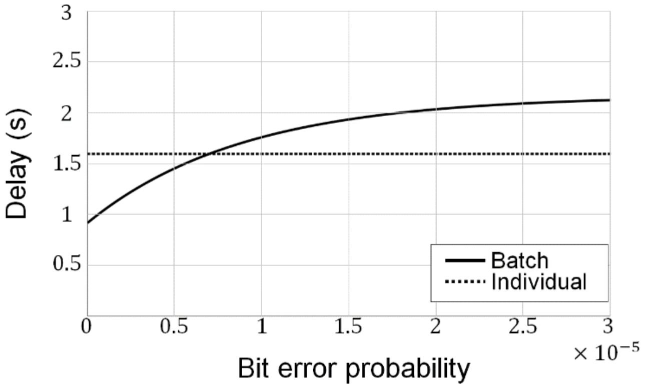


FIGURE 3. Comparison of the batch and individual verification delays in processing 100 messages for the PairHand handover authentication protocol described by He *et al.* [10].

In (7), if P_E is 0, then $E[d] = d_{bat}$ by (4). And if P_E is 1, then $E[d] = d_{bat-fail}$ by (5). Consequentially, $E[d]$ increases as p_e increases; thus, if p_e is big enough, $E[d]$ exceeds d_{ind}. Fig. 3 depicts the verification delay according to bit error probability when the protocol proposed by He *et al.* [10] is used to process 100 handover request messages in one AP. A dashed line and a solid line indicate an individual verification delay and a batch verification delay, respectively. Let us now find P_E where the solid line meets the dashed line. Equation (8) is a representation of P_E after substituting d_{ind} for $E[d]$ in (7) when $E[d] = d_{ind}$.

$$P_E = 1 - \frac{d_{bat-chk}}{d_{ind-chk}} \tag{8}$$

Equation (9) is a representation of p_e after substituting (6) for P_E in (8).

$$p_e = 1 - \left(\frac{d_{bat-chk}}{d_{ind-chk}}\right)^{\frac{1}{mn}} \tag{9}$$

Here, if we set p_e in (9) to the bit error probability p_x at intersection of individual and batch verifications, we see that when p_e is less than p_x, the batch verification is efficient because the batch verification delay is less than the individual verification delay. On the other hand, when p_e is greater than p_x, a batch verification is less efficient than individual verification. Lower p_x in a batch verification implies the protocol is more sensitive to errors. Therefore, in (9), as $d_{bat-chk}/d_{ind-chk}$ increases the protocol becomes more sensitive to errors. Likewise, as the length of the request message m and the number of MNs n participating in verification increase, the protocol becomes more sensitive to errors.

Here, since m and n are directly associated with communication cost, this can be stated in another way: as the communication cost increases, the protocol becomes more sensitive to errors. This will be proved in Section IV through analysis of each protocol.

IV. EVALUATION AND COMPARISON OF PROTOCOLS

In this section, we calculate the computation costs of various protocols previously proposed. Moreover, the bit error

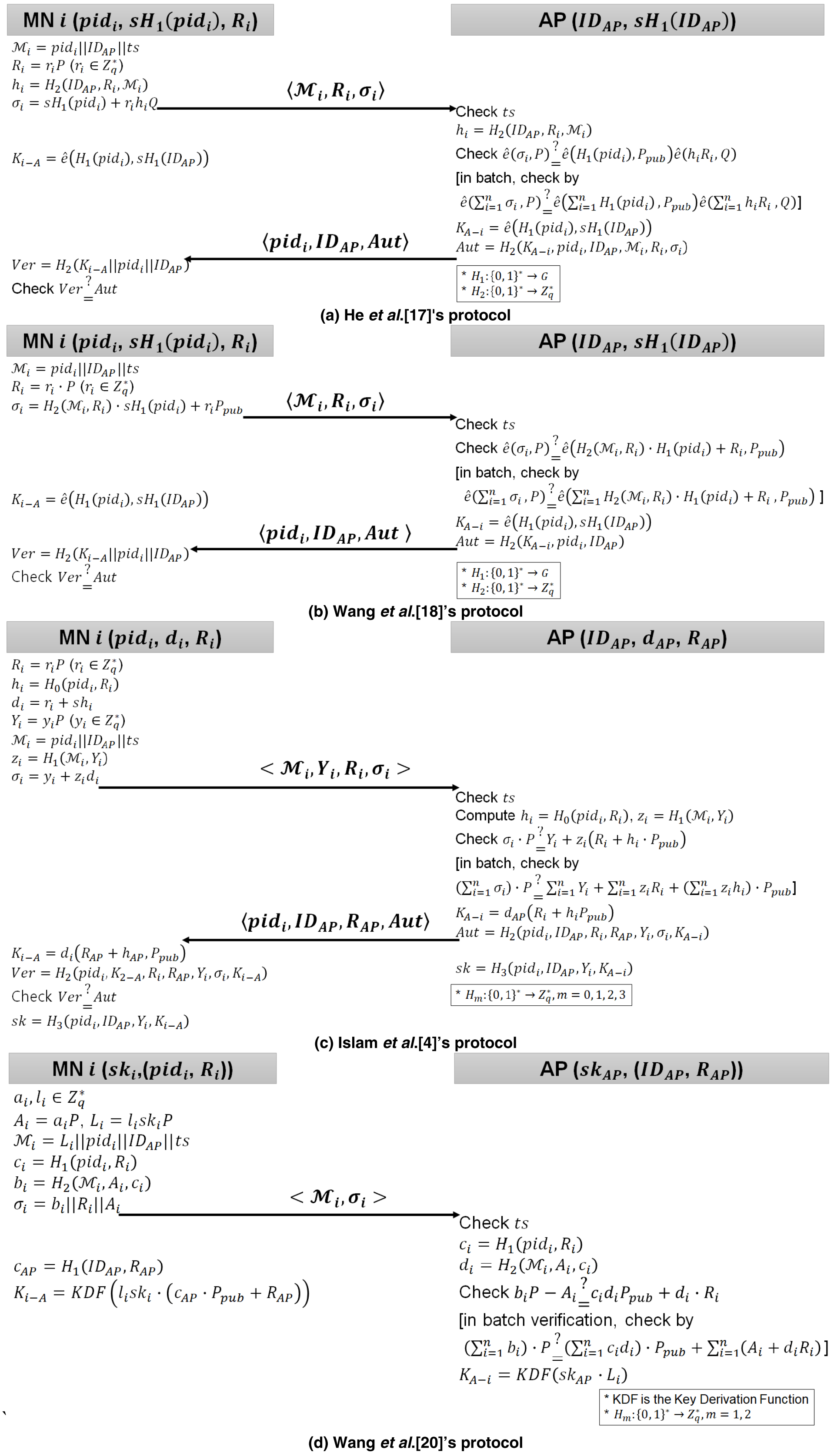

MN i $(pid_i, sH_1(pid_i), R_i)$
AP $(ID_{AP}, sH_1(ID_{AP}))$
$\mathcal{M}_i = pid_i||ID_{AP}||ts$
$R_i = r_iP$ $(r_i \in Z_q^*)$
$h_i = H_2(ID_{AP}, R_i, \mathcal{M}_i)$
$\sigma_i = sH_1(pid_i) + r_ih_iQ$
$\langle \mathcal{M}_i, R_i, \sigma_i \rangle$
Check ts
$h_i = H_2(ID_{AP}, R_i, \mathcal{M}_i)$
Check $\hat{e}(\sigma_i, P) \overset{?}{=} \hat{e}(H_1(pid_i), P_{pub})\hat{e}(h_iR_i, Q)$
[in batch, check by
$\hat{e}(\sum_{i=1}^n \sigma_i, P) \overset{?}{=} \hat{e}(\sum_{i=1}^n H_1(pid_i), P_{pub})\hat{e}(\sum_{i=1}^n h_iR_i, Q)]$
$K_{i-A} = \hat{e}(H_1(pid_i), sH_1(ID_{AP}))$
$K_{A-i} = \hat{e}(H_1(pid_i), sH_1(ID_{AP}))$
$Aut = H_2(K_{A-i}, pid_i, ID_{AP}, \mathcal{M}_i, R_i, \sigma_i)$
$\langle pid_i, ID_{AP}, Aut \rangle$
$Ver = H_2(K_{i-A}||pid_i||ID_{AP})$
Check $Ver \overset{?}{=} Aut$
* $H_1: \{0,1\}^* \to G$
* $H_2: \{0,1\}^* \to Z_q^*$
(a) He et al.[17]'s protocol
MN i $(pid_i, sH_1(pid_i), R_i)$
AP $(ID_{AP}, sH_1(ID_{AP}))$
$\mathcal{M}_i = pid_i||ID_{AP}||ts$
$R_i = r_i \cdot P$ $(r_i \in Z_q^*)$
$\sigma_i = H_2(\mathcal{M}_i, R_i) \cdot sH_1(pid_i) + r_iP_{pub}$
$\langle \mathcal{M}_i, R_i, \sigma_i \rangle$
Check ts
Check $\hat{e}(\sigma_i, P) \overset{?}{=} \hat{e}(H_2(\mathcal{M}_i, R_i) \cdot H_1(pid_i) + R_i, P_{pub})$
[in batch, check by
$\hat{e}(\sum_{i=1}^n \sigma_i, P) \overset{?}{=} \hat{e}(\sum_{i=1}^n H_2(\mathcal{M}_i, R_i) \cdot H_1(pid_i) + R_i, P_{pub})$]
$K_{i-A} = \hat{e}(H_1(pid_i), sH_1(ID_{AP}))$
$K_{A-i} = \hat{e}(H_1(pid_i), sH_1(ID_{AP}))$
$Aut = H_2(K_{A-i}, pid_i, ID_{AP})$
$\langle pid_i, ID_{AP}, Aut \rangle$
$Ver = H_2(K_{i-A}||pid_i||ID_{AP})$
Check $Ver \overset{?}{=} Aut$
* $H_1: \{0,1\}^* \to G$
* $H_2: \{0,1\}^* \to Z_q^*$
(b) Wang et al.[18]'s protocol
MN i (pid_i, d_i, R_i)
AP $(ID_{AP}, d_{AP}, R_{AP})$
$R_i = r_iP$ $(r_i \in Z_q^*)$
$h_i = H_0(pid_i, R_i)$
$d_i = r_i + sh_i$
$Y_i = y_iP$ $(y_i \in Z_q^*)$
$\mathcal{M}_i = pid_i||ID_{AP}||ts$
$z_i = H_1(\mathcal{M}_i, Y_i)$
$\sigma_i = y_i + z_id_i$
$< \mathcal{M}_i, Y_i, R_i, \sigma_i >$
Check ts
Compute $h_i = H_0(pid_i, R_i)$, $z_i = H_1(\mathcal{M}_i, Y_i)$
Check $\sigma_i \cdot P \overset{?}{=} Y_i + z_i(R_i + h_i \cdot P_{pub})$
[in batch, check by
$(\sum_{i=1}^n \sigma_i) \cdot P \overset{?}{=} \sum_{i=1}^n Y_i + \sum_{i=1}^n z_iR_i + (\sum_{i=1}^n z_ih_i) \cdot P_{pub}]$
$K_{A-i} = d_{AP}(R_i + h_iP_{pub})$
$Aut = H_2(pid_i, ID_{AP}, R_i, R_{AP}, Y_i, \sigma_i, K_{A-i})$
$\langle pid_i, ID_{AP}, R_{AP}, Aut \rangle$
$K_{i-A} = d_i(R_{AP} + h_{AP}, P_{pub})$
$Ver = H_2(pid_i, K_{2-A}, R_i, R_{AP}, Y_i, \sigma_i, K_{i-A})$
Check $Ver \overset{?}{=} Aut$
$sk = H_3(pid_i, ID_{AP}, Y_i, K_{i-A})$
$sk = H_3(pid_i, ID_{AP}, Y_i, K_{A-i})$
* $H_m: \{0,1\}^* \to Z_q^*, m = 0,1,2,3$
(c) Islam et al.[4]'s protocol
MN i $(sk_i, (pid_i, R_i))$
AP $(sk_{AP}, (ID_{AP}, R_{AP}))$
$a_i, l_i \in Z_q^*$
$A_i = a_iP$, $L_i = l_isk_iP$
$\mathcal{M}_i = L_i||pid_i||ID_{AP}||ts$
$c_i = H_1(pid_i, R_i)$
$b_i = H_2(\mathcal{M}_i, A_i, c_i)$
$\sigma_i = b_i||R_i||A_i$
$< \mathcal{M}_i, \sigma_i >$
Check ts
$c_i = H_1(pid_i, R_i)$
$d_i = H_2(\mathcal{M}_i, A_i, c_i)$
Check $b_iP - A_i \overset{?}{=} c_id_iP_{pub} + d_i \cdot R_i$
[in batch verification, check by
$(\sum_{i=1}^n b_i) \cdot P \overset{?}{=} (\sum_{i=1}^n c_id_i) \cdot P_{pub} + \sum_{i=1}^n (A_i + d_iR_i)]$
$c_{AP} = H_1(ID_{AP}, R_{AP})$
$K_{i-A} = KDF(l_isk_i \cdot (c_{AP} \cdot P_{pub} + R_{AP}))$
$K_{A-i} = KDF(sk_{AP} \cdot L_i)$
* KDF is the Key Derivation Function
* $H_m: \{0,1\}^* \to Z_q^*, m = 1,2$
(d) Wang et al.[20]'s protocol

FIGURE 4. Handover authentication phase of various protocols: (a) He *et al.* [17]'s protocol, (b) Wang *et al.* [18]'s protocol, c) Islam *et al.* [4]'s protocol, and (d) Wang *et al.* [20]'s protocol.

TABLE 3. Evaluation of communication cost.

PROTOCOL	COMMUNICATION COST (IN BITS)		ELLIPTIC CURVE
	AP RECEIVE	AP SEND	
He *et al.* [10]	32+32+32+1,024=1,120	32+32+160=224	Super singular
He *et al.* [17]	32+32+32+1,024+1,024=2,144	32+32+160=224	Super singular
Wang *et al.* [18]	32+32+32+1,024+1,024 = 2,144	32+32+160=224	Super singular
Islam *et al.* [4]	32+32+32+320+320+160=896	32+32+320+160=544	Non-super singular
Wang *et al.* [20]	320+32+32+32+160+320+320=1,216	0	Non-super singular

probability p_x at intersection is derived by comparing the computation costs and the verification delays according to the bit error probability using a graph. By doing this, it is possible to verify which protocol is more sensitive to errors.

A. COMPARISON OF AUTHENTICATION PHASES OF EACH PROTOCOL

Fig. 4 depicts the handover authentication phases of each protocol. From this, we can infer the lengths of request messages transmitted via wireless channels and the operations required for verification in an AP. The protocols of He *et al.* [17] and Wang and Hu [18] have a higher communication cost than the protocol of He *et al.* [10]. Because the protocols of Islam and Khan [4] and Wang *et al.* [20] use non-super singular elliptic curves, the communication cost is low and the verification is fast. The protocol of Wang *et al.* [20] exchanges the parameters required for key computation in the initialization phase, and after the verification succeeds, the symmetric key is calculated immediately without sending an authentication message from the AP to the MN.

B. EVALUATION OF COMMUNICATION AND COMPUTATION COST

According to analysis results in Section III, the batch verification delay increases if an error occurs in the message that the AP receives from the MN. On the other hand, if an error occurs in the message that the AP sent to the MN, the batch verification delay of the AP does not increase because the MN must perform the verification process all over again from the beginning. Therefore, it is necessary to classify the communication cost into two parts: the cost that the AP receives from the MN and the cost that the AP sends to the MN. To analyze the communication cost, the length of each parameter must be identified. The lengths of p, $\bar{p}$, q, and $\bar{q}$, parameters that are used in each protocol, are 512, 160, 160, and 160 bits, respectively. Accordingly, the lengths of G, $\bar{G}$, and Z_q^* are 1024, 320, and 160 bits, respectively. Pseudo-ID pid_i of the MN, ID ID_{AP} of the AP, and timestamp ts are each 4 bytes. In addition, the results of the cryptographic hash function and the map-to-point in G function are 160 bits and 1024 bits, respectively. Based on the description thus far, Table 3 summarizes the communication cost calculation when an AP sends or receives messages.

In the protocol of He *et al.* [10] that uses a super singular elliptic curve, a request message consists of pid_i, ID_{AP}, ts, and σ_i. Since the length of pid_i, ID_{AP}, and ts is 32 bits each, and σ_i is 1024 bits, the length of a message that the AP receives, as presented in Table 3, becomes 1,120 bits. Since the request messages of He *et al.* [17] and Wang and Hu [18] are longer than those of He *et al.* [10] by a random point R_i, the message length that the AP receives is 2,144 bits. In the protocol of Islam and Khan [4] that uses a non-super singular elliptic curve, a request message consists of pid_i, ID_{AP}, ts, Y_i, R_i, and σ_i. Since the lengths of pid_i, ID_{AP}, ts, Y_i, R_i, and σ_i are 32, 32, 32, 320, 320, 160 bits, respectively, the length of a message that the AP receives becomes 896 bits. Finally, the protocol of Wang *et al.* [20] with a request message consists of L_i, pid_i, ID_{AP}, ts, b_i, R_i, and A_i. Since the lengths of L_i, pid_i, ID_{AP}, ts, b_i, R_i, and A_i are 320, 32, 32, 32, 160, 320, 320 bits, respectively, the length of a message that the AP receives becomes 1,216 bits.

The protocols of He *et al.* [10], Wang and Lu [18], and He *et al.* [17] all transmit pid_i, ID_{AP}, and authentication message Aut to the MN after a successful verification, so the length of the message that the AP sends becomes 224 bits. The protocol of Islam and Khan [4] transmits pid_i, ID_{AP}, R_{AP}, and Aut to the MN after a successful verification, so the length of the message that the AP sends becomes 544 bits. The protocol of Chaudhry *et al.* [21] is capable of performing key computation of the AP and the MN without a reply from the AP and the length of the message that the AP transmits is 0, accordingly.

Table 4 presents the delays under the assumption that each protocol performs individual verifications for messages sent from 100 MNs using the processing times listed in Table 2. In the case of He *et al.* [10]'s protocol depicted in Fig. 2, T_{bp}^G is required in the calculation of $\hat{e}\left(\sigma_i, P\right)$ and T_h, T_{M2P}^G, T_{sm}^G, T_{bp}^G are required for $\hat{e}\left(H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P_{pub}\right)$. Specifically, T_h is required in the calculation of $H_1\left(pid_i\right)$, T_{M2P}^G in the calculation of $H_2\left(\mathcal{M}_i\right)$, and T_{sm}^G in the calculation of $H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right)$. Since $H_1\left(pid_i\right)$ is already calculated, only T_{bp}^G is required in the calculation of $\hat{e}\left(H_1\left(pid_i\right), sH_1\left(ID_{AP}\right)\right)$. Furthermore, T_{bp}^G is required in the calculation of K_{A-i} and T_h in the calculation of Aut. Consequently, He *et al.* [10]'s protocol requires $T_{sm}^G + T_{M2P}^G + 2T_{bp}^G + T_h$ to check one message and

TABLE 4. Evaluation of individual verification delay.

PROTOCOL	INDIVIDUAL VERIFICATION DELAY ($d_{ind-chk} + d_{key}$, $n = 100$)		
	$d_{ind-chk}$	d_{key}	TOTAL
He *et al.* [10]	$n\left(T_{sm}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} + T_h\right) \approx$ 1,251.29 ms	$n\left(T_{bp}^{G} + T_h\right) \approx$ 344.16 ms	1,559.45 ms
He *et al.* [17]	$n\left(T_{sm}^{G} + T_{M2P}^{G} + 3T_{bp}^{G} + T_h\right) \approx$ 1,595.39 ms	$n\left(T_{bp}^{G} + T_h\right) \approx$ 344.16 ms	1,939.55 ms
Wang *et al.* [18]	$n\left(T_{sm}^{G} + T_{pa}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} + T_h\right) \approx$ 1,252.19 ms	$n\left(T_{bp}^{G} + T_h\right) \approx$ 344.16 ms	1,596.35 ms
Islam *et al.* [4]	$n\left(3T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} + 2T_h\right) \approx$ 111.30 ms	$n\left(2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}} + 2T_h\right) \approx$ 74.14 ms	185.44 ms
Wang *et al.* [20]	$n\left(3T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} + 2T_h\right) \approx$ 111.30 ms	$n\left(T_{sm}^{\bar{G}} + T_h\right) \approx$ 36.92 ms	148.22 ms

TABLE 5. Evaluation of batch verification delay.

PROTOCOL	BATCH VERIFICATION DELAY ($d_{bat-chk} + d_{key}$, $n = 100$)			
	$d_{bat-chk}$	Total	$d_{bat-chk}/d_{ind-chk}$	p_x
He *et al.* [10]	$n\left(T_{sm}^{G} + 2T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 2T_{bp}^{G} \approx$ 571.77 ms	915.93 ms	0.4569	6.99×10^{-6}
He *et al.* [17]	$n\left(T_{sm}^{G} + 3T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 3T_{bp}^{G} \approx$ 576.11 ms	920.27 ms	0.3611	4.75×10^{-6}
Wang *et al.* [18]	$n\left(T_{sm}^{G} + 3T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 2T_{bp}^{G} \approx$ 572.67 ms	916.83 ms	0.4573	3.64×10^{-6}
Islam *et al.* [4]	$n\left(T_{sm}^{\bar{G}} + 4T_{pa}^{\bar{G}} + 2T_h\right) + 2T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} \approx 38.92\ ms$	113.06 ms	0.3496	11.72×10^{-6}
Wang *et al.* [20]	$n\left(T_{sm}^{\bar{G}} + 4T_{pa}^{\bar{G}} + 2T_h\right) + 2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}} \approx 38.92\ ms$	75.84 ms	0.3496	8.64×10^{-6}

$T_{bp}^{G} + T_h$ to calculate one key. If the protocol individually verifies n messages, the verification delay can be expressed as $n(T_{sm}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} + T_h) + n\left(T_{bp}^{G} + T_h\right)$.

Since He *et al.* [17]'s protocol depicted in Fig. 4(a) requires one more bilinear pairing operation than He *et al.* [10]'s protocol, it needs $\left(T_{sm}^{G} + T_{M2P}^{G} + 3T_{bp}^{G} + T_h\right)$ to check one message and $T_{bp}^{G} + T_h$ to calculate one key. If the protocol individually verifies n messages, the verification delay can be expressed as $n\left(T_{sm}^{G} + T_{M2P}^{G} + 3T_{bp}^{G} + T_h\right) + n\left(T_{bp}^{G} + T_h\right)$. Since Wang and Hu [18]'s protocol depicted in Fig. 4(b) requires one more point addition operation than He *et al.* [10]'s protocol, it needs $T_{sm}^{G} + T_{pa}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} + T_h$ to check one message and $T_{bp}^{G} + T_h$ to calculate one key. If the protocol individually verifies n messages, the verification delay can be expressed as $n\left(T_{sm}^{G} + T_{pa}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} + T_h\right) + n\left(T_{bp}^{G} + T_h\right)$.

Regarding Islam and Khan [4]'s protocol depicted in Fig. 4(c), $2T_h$ is required in the calculation of z_i and h_i, $T_{sm}^{\bar{G}}$ is required in the calculation of $\sigma_i \cdot P$, and $2T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}}$ is required in the calculation of $Y_i + z_i\left(R_i + h_i \cdot P_{pub}\right)$. Furthermore, $2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}$ is required in the calculation of K_{A-i} and $2T_h$ in the calculation of *Aut* and *sk*. Consequently, Islam and Khan [4]'s protocol needs $3T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} + 2T_h$ to check one message and $2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}} + 2T_h$ to calculate one key. If the protocol individually verifies n messages, the verification delay can be expressed as $n\left(3T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} + 2T_h\right) + n\left(2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}} + 2T_h\right)$. Regarding Wang *et al.* [20]'s protocol depicted in Fig. 4(d), $2T_h$ is required in the calculation of c_i and d_i, $T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}$ in the calculation of $b_iP - A_i$, $2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}$ in the calculation of $c_i d_i P_{pub} + d_i \cdot R_i$, and $T_{sm}^{\bar{G}} + T_h$ in the calculation of K_{A-i}. Consequently, Wang *et al.* [20]'s protocol needs $3T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} + 2T_h$ to check one message and $T_{sm}^{\bar{G}} + T_h$ to calculate one key. If the protocol individually verifies n messages, the verification delay can be expressed as $n\left(3T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}} + 2T_h\right) + n\left(T_{sm}^{\bar{G}} + T_h\right)$.

Table 5 presents the parameters related to batch verification delay. We consider only the signature checking time because the key computation time is similar to that in the case of individual verification. In He *et al.* [10]'s protocol depicted in Fig. 2, $n \times T_{pa}^{G} + T_{bp}^{G}$ is required in the calculation of $\hat{e}\left(\sum_{i=1}^{n} \sigma_i, P\right)$ and $n \times \left(T_{sm}^{G} + T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + T_{bp}^{G}$ in the calculation of $\hat{e}\left(\sum_{i=1}^{n} H_2\left(\mathcal{M}_i\right) \cdot H_1\left(pid_i\right), P_{pub}\right)$. Consequently, He *et al.* [10]'s protocol needs $n \times \left(T_{sm}^{G} + 2T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 2T_{bp}^{G}$ to batch verify n messages. Since He *et al.* [17]'s protocol in Fig. 4(a) requires n more point addition operations $n \times T_{pa}^{G}$ and one more bilinear pairing operation T_{bp}^{G} than He *et al.* [10]'s protocol, it needs $n \times \left(T_{sm}^{G} + 3T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 3T_{bp}^{G}$ to batch verify n messages. Since Wang and Hu [18]'s protocol depicted in Fig. 4(b) requires n more point addition operations for $\hat{e}\left(\sum_{i=1}^{n} \ldots + R_i, \ldots\right)$ than He *et al.* [10]'s protocol, $n \times \left(T_{sm}^{G} + 3T_{pa}^{G} + T_{M2P}^{G} + T_h\right) + 2T_{bp}^{G}$ is required to check n messages.

For the protocol of Islam and Khan [4] depicted in Fig. 4(c), $n \times T_{pa}^{\bar{G}} + T_{sm}^{\bar{G}}$ is required in the calculation of $\left(\sum_{i=1}^{n} \sigma_i\right) \cdot P$, and

$n \times T_{pa}^{\bar{G}} + n \times \left(T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}\right) + n \times T_{pa}^{\bar{G}} + T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}}$ is required in the calculation of $\sum_{i=1}^{n} Y_i + \sum_{i=1}^{n} z_i R_i + \left(\sum_{i=1}^{n} z_i h_i\right) \cdot P_{pub}$. The calculation of z_i and h_i is the same as that in the case of individual verification. Consequently, Islam and Khan [4]'s protocol needs $n \times \left(T_{sm}^{\bar{G}} + 4T_{pa}^{\bar{G}} + 2T_h\right) + 2T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}}$ to batch verify n messages. Regarding Wang *et al.* [20]'s protocol shown in Fig. 4(d), $n \times T_{pa}^{\bar{G}} + T_{sm}^{\bar{G}}$ is required in the calculation of $\left(\sum_{i=1}^{n} b_i\right) \cdot P$, and $n \times T_{pa}^{\bar{G}} + T_{sm}^{\bar{G}} + n \times \left(T_{sm}^{\bar{G}} + 2T_{pa}^{\bar{G}}\right) + T_{pa}^{\bar{G}}$ in the calculation of $\left(\sum_{i=1}^{n} c_i d_i\right) \cdot P_{pub} + \sum_{i=1}^{n} \left(A_i + d_i R_i\right)$. The calculation of c_i and d_i is the same as that in the case of individual verification. Consequently, Wang *et al.* [20]'s protocol needs $n \times \left(T_{sm}^{\bar{G}} + 4T_{pa}^{\bar{G}} + 2T_h\right) + 2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}$ to batch verify n messages. The batch verification delay between the protocols of Islam and Khan [4] and Wang *et al.* [20] has only a difference of $T_{pa}^{\bar{G}}$, which is insignificant compared to operations such as $T_{sm}^{\bar{G}}$ and T_{M2P}^{G}.

In Table 5, p_x was derived using (9). When the protocols of He *et al.* [17] and Wang *et al.* [20], which have similar $d_{bat-chk}/d_{ind-chk}$, are compared, the protocol of Wang *et al.* [20], which has a lower communication cost, shows a larger p_x. Likewise, when the protocols of He *et al.* [10] and Wang and Hu [18], which have similar $d_{bat-chk}/d_{ind-chk}$, are compared, the protocol of He *et al.* [10], which has a lower communication cost, shows a larger p_x. When the protocols of He *et al.* [17] and Wang and Hu [18], which both have the same communication costs, are compared, the protocol of He *et al.* [17], which has a lower $d_{bat-chk}/d_{ind-chk}$, shows a larger p_x. Consequently, as described in Section III, the greater the $d_{bat-chk}/d_{ind-chk}$ and the greater the communication cost, the more sensitive the protocol is to errors.

Fig. 5 is a graph that represents the expectation $E\left[d\right]$ of verification delay according to a bit error probability p_e by applying the results of Table 4 and Table 5 as well as (7). As previously described in Section III.A, because an individual verification is not affected by errors, no change in delay is observed even if p_e increases. On the other hand, batch verification delay increases as p_e increases. In Fig. 5, p_x indicates the intersection of individual verification delay and batch verification delay of each protocol. As shown in Table 5. The protocol of Wang and Hu [18] with a small p_x shows a sharply increased tendency in batch verification delay with increasing p_e. This implies that the protocol is more sensitive to errors as described in Section III.B. On the contrary, the protocols of Islam and Khan [4] and Wang *et al.* [20], which have higher p_x, show a relatively gradual increase in batch verification delay as p_e increases. This implies that these protocols are less sensitive to errors.

V. MINI-BATCH VERIFICATION TO MITIGATE THE IMPACT OF ERRORS

In this section, we propose a method to reduce the delay of batch verification. In Section III, we showed that batch verification fails when an error of one bit or more occurs among several handover request messages. Therefore, by reducing the number of handover request messages that are simultaneously processed in the batch verification phase, the impact of errors can be reduced. We therefore propose a method to verify handover request messages by dividing them into several mini-batches rather than verifying them all at once. If 100 handover request messages are assumed to be processed by one AP, these messages can divided into two mini-batches, as shown in Fig. 6. The number of messages in a mini-batch should be 2 or more.

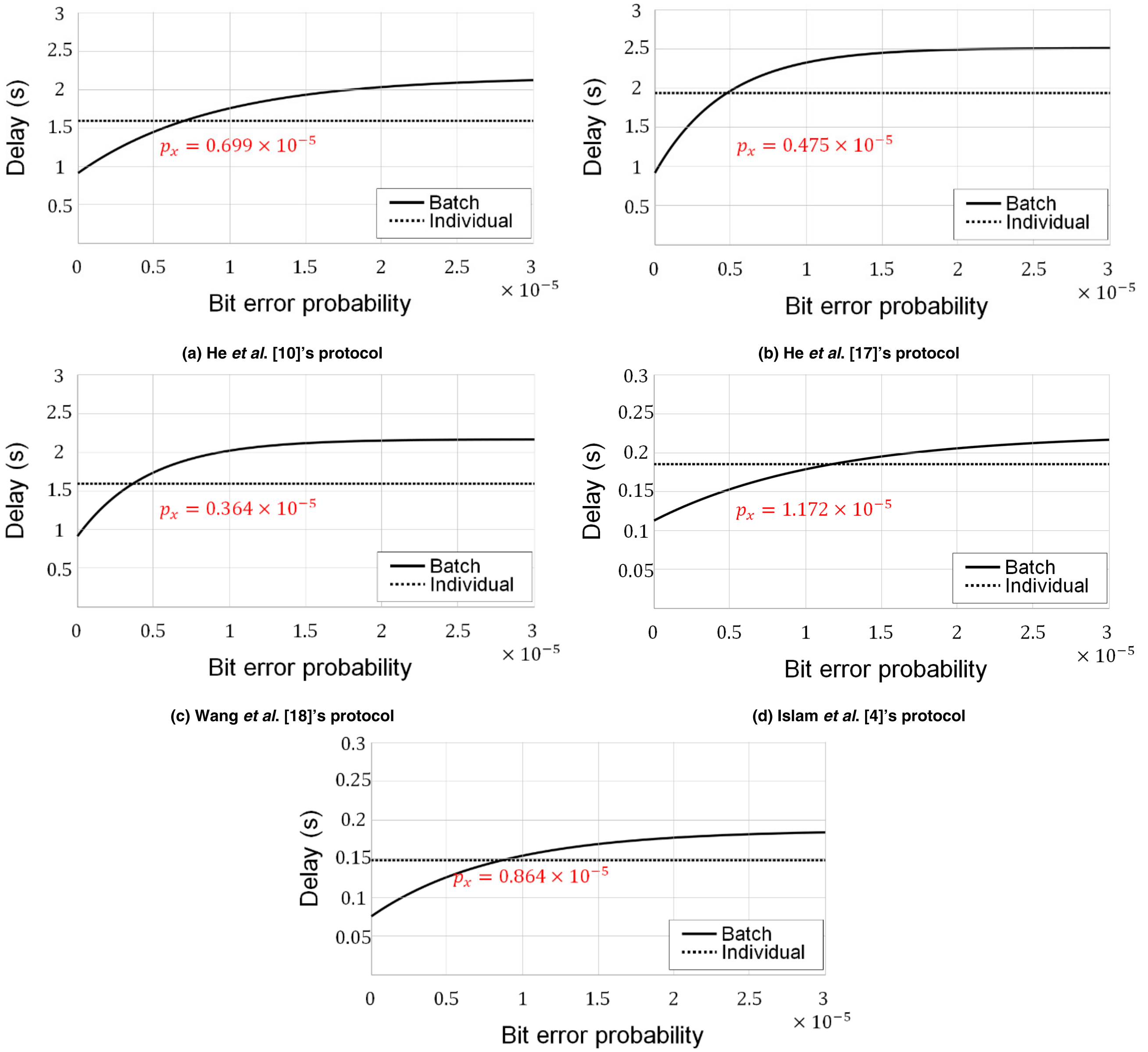
If we denote each verification delay as d_{k-mb} when n handover messages are divided into k mini-batches, d_{k-mb} can be expressed as (10). In (10), $d_{\mathrm{bat}-chk}(n/k)$ refers to the delay required for signature check while verifying n/k handover request messages, and $d_{key}(n)$ refers to the delay required for calculating n keys.

$$\begin{aligned} d_{k-mb} &= k \times d_{bat-chk}\left(n/k\right) + d_{key}\left(n\right) \\ &= k \times (n/k \times (T_{sm}^{G} + 2T_{pa}^{G} + T_{M2P} + T_h) \\ &\quad + 2T_{bp}^{G}) + d_{key}\left(n\right) \\ &= d_{bat-chk}\left(n\right) + 2\left(k-1\right) \times T_{bp}^{G} + d_{key}\left(n\right) \qquad (10) \end{aligned}$$

The term $2\left(k-1\right) \times T_{bp}^{G}$ in (10) represents the additional overhead due to mini-batch verification compared to (4). $E\left[d_{k-mb}\right]$, which is the expectation of a mini-batch verification delay, is expressed as (11). In (11), $P_E(n/k)$ indicates the probability of an error occurring in one or more bits within n/k request messages.

$$\begin{aligned} E\left[d_{k-mb}\right] &= k \times (d_{bat-chk}(n/k) + d_{ind-chk}(n/k) \\ &\quad \times P_E(n/k)) + d_{key}(n) \\ &= k \times (n/k \times (T_{sm}^{G} + 2T_{pa}^{G} + T_{M2P} + T_h) \\ &\quad + 2T_{bp}^{G}) + n/k \times (T_{sm}^{G} + T_{M2P}^{G} + 2T_{bp}^{G} \\ &\quad + T_h) \times P_E(n/k)) + d_{key}(n) \\ &= d_{bat-chk}\left(n\right) + d_{ind-chk}\left(n\right) \times P_E\left[n/k\right] \\ &\quad + 2\left(k-1\right) \times T_{bp}^{G} + d_{key}(n) \qquad (11) \end{aligned}$$

The verification delays when the protocols analyzed in section IV were applied to mini-batch verification, are summarized in Table 6. Fig. 7 depicts the expectation of mini-batch verification delay when $k = 1, 2, 5, 10, 25, 50$, which are based on the expectation values presented in Table 6. Two characteristics are observed to be common among the five graphs depicted in Fig. 7. First, if the bit error probability is exceedingly small (close to zero), or very large (greater than 10^{-3}), the performance of mini-batch verification is worse as compared to that of batch verification. This is due to the additional overhead $2\left(k-1\right) \times T_{bp}^{G}$ of mini-batch verification in (11). However, when the bit error probability is a mid-range value, the delays of some mini-batch verifications are found to be smaller than those of batch verifications. It is interpreted that $P_E\left[n/k\right]$ in (11) is always smaller than P_E in (7), and the difference is maximized when the bit error probability is a mid-range value, and consequently, the mini-batch overhead is offset.



(a) He *et al.* [10]'s protocol

(b) He *et al.* [17]'s protocol

(c) Wang *et al.* [18]'s protocol

(d) Islam *et al.* [4]'s protocol

(e) Wang *et al.* [20]'s protocol

FIGURE 5. Expectation graph for verification delays for various protocols: (a) He *et al.* [10]'s protocol, (b) He *et al.* [17]'s protocol, (c) Wang *et al.* [18]'s protocol, (d) Islam *et al.* [4]'s protocol, and (e) Wang *et al.* [20]'s protocol.

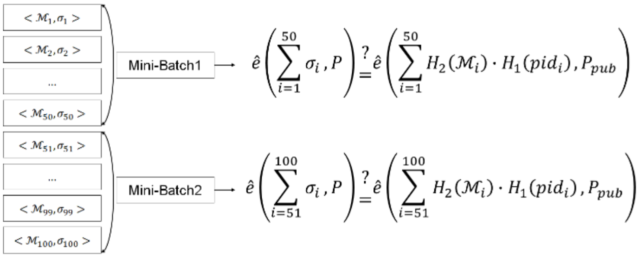

$< \mathcal{M}_1, \sigma_1 >$
$< \mathcal{M}_2, \sigma_2 >$
...
$< \mathcal{M}_{50}, \sigma_{50} >$
$< \mathcal{M}_{51}, \sigma_{51} >$
...
$< \mathcal{M}_{99}, \sigma_{99} >$
$< \mathcal{M}_{100}, \sigma_{100} >$
Mini-Batch1
Mini-Batch2
$\hat{e}\left(\sum_{i=1}^{50} \sigma_i, P\right) \overset{?}{=} \hat{e}\left(\sum_{i=1}^{50} H_2(\mathcal{M}_i) \cdot H_1(pid_i), P_{pub}\right)$
$\hat{e}\left(\sum_{i=51}^{100} \sigma_i, P\right) \overset{?}{=} \hat{e}\left(\sum_{i=51}^{100} H_2(\mathcal{M}_i) \cdot H_1(pid_i), P_{pub}\right)$

FIGURE 6. An example showing how to divide and verify *n* messages into 2 mini-batches.

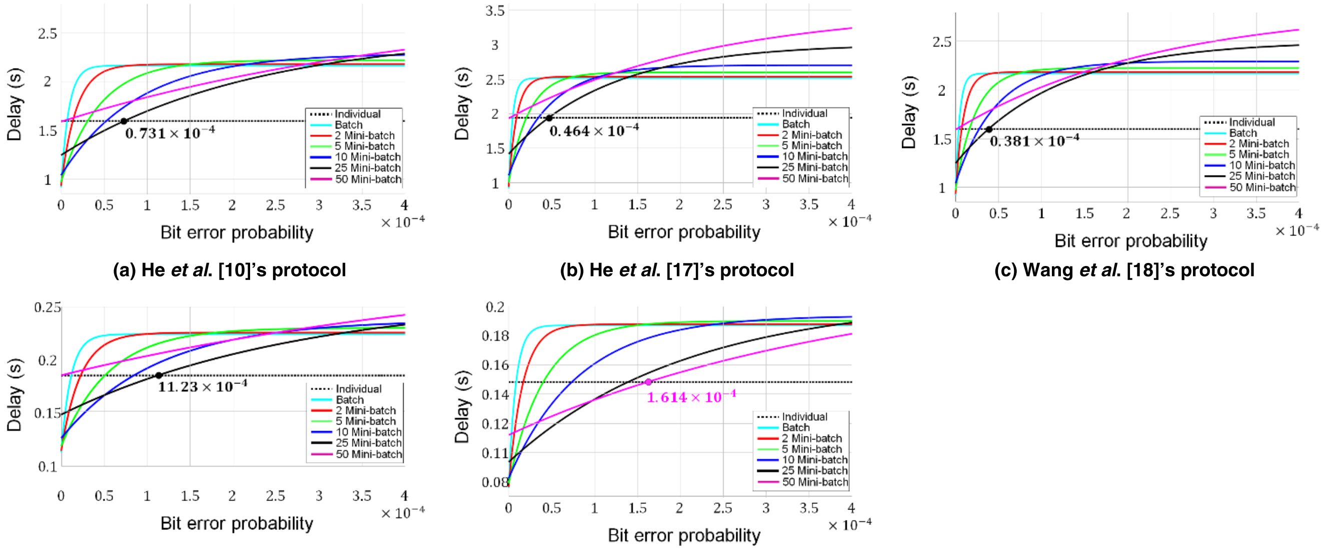
Second, the number of mini-batches is not always proportional to the performance of the verification method. Depending on the bit error probability, the k value of the mini-batch that minimizes the expected verification delay varies. This is due to the two terms, $d_{ind-chk}(n) \times P_E[n/k]$ and $2(k-1) \times T_{bp}^G$, given in (11). Notably, $d_{ind-chk}(n) \times P_E[n/k]$ decreases as k increases. On the contrary, $2(k-1) \times T_{bp}^G$ increases as k increases. If k is too large, the delay becomes worse because the decrease due to $d_{ind-chk}(n) \times P_E[n/k]$ is offset by the increase due to $2(k-1) \times T_{bp}^G$. That is, it is necessary to find the optimal value of k that can minimize (11). Consider the following equation:

$$k^* = argmin\left(E\left[d_{k-mb}\right]\right), \quad k = 1, 2, \ldots, n/2 \qquad (12)$$

When mini-batch verification is used, messages can be verified with a delay smaller than that of individual verification over a wider range of bit error probability as

TABLE 6. Verification delay of mini-batch.

PROTOCOL	MINI-BATCH VERIFICATION DELAY	EXPECTATION VALUE OF VERIFICATION DELAY
He *et al.* [10]	$d_{bat-chk}(n) + 2(k-1) \times T_{bp}^{G} + d_{key}(n)$	$d_{bat-chk}(n) + d_{ind-chk}(n) \times P_E(n/k) + 2(k-1)T_{bp}^{G} + d_{key}(n)$
He *et al.* [17]	$d_{bat-chk}(n) + 3(k-1) \times T_{bp}^{G} + d_{key}(n)$	$d_{bat-chk}(n) + d_{ind-chk}(n) \times P_E(n/k) + 3(k-1)T_{bp}^{G} + d_{key}(n)$
Wang *et al.* [18]	$d_{bat-chk}(n) + 2(k-1) \times T_{bp}^{G} + d_{key}(n)$	$d_{bat-chk}(n) + d_{ind-chk}(n) \times P_E(n/k) + 2(k-1)T_{bp}^{G} + d_{key}(n)$
Islam *et al.* [4]	$d_{bat-chk}(n) + (k-1)\left(2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}\right) + d_{key}(n)$	$d_{bat-chk}(n) + d_{ind-chk}(n) \times P_E(n/k) + (k-1)\left(2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}\right) + d_{key}(n)$
Wang *et al.* [20]	$d_{bat-chk}(n) + (k-1)\left(2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}\right) + d_{key}(n)$	$d_{bat-chk}(n) + d_{ind-chk}(n) \times P_E(n/k) + (k-1)\left(2T_{sm}^{\bar{G}} + T_{pa}^{\bar{G}}\right) + d_{key}(n)$



(a) He *et al.* [10]'s protocol

(b) He *et al.* [17]'s protocol

(c) Wang *et al.* [18]'s protocol

(d) Islam *et al.* [4]'s protocol

(e) Wang *et al.* [20]'s protocol

FIGURE 7. Expectation graph for mini-batch verification delays for various protocols: (a) He *et al.* [10]'s protocol, (b) He *et al.* [17]'s protocol, (c) Wang *et al.* [18]'s protocol, (d) Islam *et al.* [4]'s protocol, and (e) Wang *et al.* [20]'s protocol.

compared to using only batch verification. For example, in the case of the protocol of He *et al.* [10], the batch verification delay is smaller than the individual verification delay when the bit error probability is approximately 0.699×10^{-5} or less, as shown in Fig. 5(a). However, in the case of mini-batch, the interval of the bit error probability is enlarged. As shown in Fig. 7(a), the interval where the mini-batch verification delay is smaller than the individual verification delay ranges from zero to 0.731×10^{-4}. In other words, the effective range of mini-batch verification compared to individual verification is roughly 10 times larger than that of a single batch. In general wireless communication, the bit error probability is measurable, and therefore, if only p_e and the number of messages are given, the optimal value of k can be found using (12).

VI. CONCLUSION

This study analyzed the communication cost and computation cost of the handover authentication protocols that have been previously proposed. Previous studies analyzed only communication cost and computation cost of individual verification and did not account for errors in message transmission, whereas this study also analyzed the computation cost of a batch verification under the assumption of bit error probability p_e in the handover request message. According to our analysis results, although the individual verification delay is not affected by errors, the batch verification delay increased with an increase in p_e.

In particular, when the error probability is high, batch verification may show lower performance compared to individual verification. Moreover, this study derived the bit error probability p_x at which the batch and individual verification delay graphs intersect. Lower p_x means that the protocol is more sensitive to errors. In addition, when we implemented and compared the previously proposed protocols, we determined that the factors influencing p_x include communication cost and the ratio $d_{bat-chk}/d_{ind-chk}$ of the batch verification delay to the individual verification delay.

Finally, we proposed a method to mitigate the impact of errors. The method verifies n messages by dividing them into several mini-batches rather than a single batch. By applying this mini-batch verification to the protocols of He *et al.* [10], He *et al.* [17], Wang and Hu [18], Islam and Khan [4], and Wang *et al.* [20], we showed that our method can reduce the impact of errors in batch verification. However, mini-batch verification has additional overhead than batch verification, and the impact of additional overhead is significant when the bit error probability is extremely small, for example, less than approximately 10^{-5} or exceedingly large, for example, greater than approximately 10^{-4}. Therefore, mini-batch verification can be applied in a mobile wireless network environment in which the bit error rate is as low as 10^{-4} or less. Furthermore, to perform mini-batch verification, it is necessary to find the optimal value of k according to the bit error rate.

In a real-world communication model such as VANET [36], messages sent over wireless communication generate more errors than theoretically calculated errors. A major contribution of this study is that it analyzed the error impact of the handover authentication protocol in a practical communication model and proposed a novel method to mitigate the impact of errors.

ACKNOWLEDGMENT

The authors would like to thank Editage (www.editage.co.kr) for English language editing.

REFERENCES

[1] J. Gozalvez, “Samsung electronics sets 5G speed record at 7.5 Gb/s [Mobile Radio],” *IEEE Veh. Technol. Mag.*, vol. 10, no. 1, pp. 12–16, Mar. 2015.

[2] D. He, D. Wang, Q. Xie, and K. Chen, “Anonymous handover authentication protocol for mobile wireless networks with conditional privacy preservation,” *Sci. China Inf. Sci.*, vol. 60, no. 5, May 2017, Art. no. 052104.

[3] X. Yang, X. Huang, and J. K. Liu, “Efficient handover authentication with user anonymity and untraceability for mobile cloud computing,” *Future Gener. Comput. Syst.*, vol. 62, pp. 190–195, Sep. 2016.

[4] S. H. Islam and M. K. Khan, “Provably secure and pairing-free identity-based handover authentication protocol for wireless mobile networks,” *Int. J. Commun. Syst.*, vol. 29, no. 17, pp. 2442–2456, Nov. 2016.

[5] J. Choi and S. Jung, “A handover authentication using credentials based on chameleon hashing,” *IEEE Commun. Lett.*, vol. 14, no. 1, pp. 54–56, Jan. 2010.

[6] G. Yang, Q. Huang, D. Wong, and X. Deng, “Universal authentication protocols for anonymous wireless communications,” *IEEE Trans. Wireless Commun.*, vol. 9, no. 1, pp. 168–174, Jan. 2010.

[7] D. He, J. Bu, S. Chan, C. Chen, and M. Yin, “Privacy-preserving universal authentication protocol for wireless communications,” *IEEE Trans. Wireless Commun.*, vol. 10, no. 2, pp. 431–436, Feb. 2011.

[8] H.-C. Hsiang and W.-K. Shih, “Improvement of the secure dynamic ID based remote user authentication scheme for multi-server environment,” *Comput. Standards Interfaces*, vol. 31, no. 6, pp. 1118–1123, Nov. 2009.

[9] Y.-P. Liao and S.-S. Wang, “A secure dynamic ID based remote user authentication scheme for multi-server environment,” *Comput. Standards Interfaces*, vol. 31, no. 1, pp. 24–29, Jan. 2009.

[10] D. He, C. Chen, S. Chan, and J. Bu, “Secure and efficient handover authentication based on bilinear pairing functions,” *IEEE Trans. Wireless Commun.*, vol. 11, no. 1, pp. 48–53, Jan. 2012.

[11] H. J. Yoon, J. H. Cheon, and Y. Kim, “Batch verifications with ID-based signatures,” in *Proc. Int. Conf. Inf. Secur. Cryptol.*, Seoul, South Korea, 2004, pp. 233–248.

[12] C. Zhang, R. Lu, X. Lin, P.-H. Ho, and X. Shen, “An efficient identity-based batch verification scheme for vehicular sensor networks,” in *Proc. IEEE 27th Conf. Comput. Commun. (INFOCOM)*, Phoenix, AZ, USA, Apr. 2008, pp. 246–250.

[13] M. Bayat, M. Barmshoory, M. Rahimi, and M. R. Aref, “A secure authentication scheme for VANETs with batch verification,” *Wireless Netw.*, vol. 21, no. 5, pp. 1733–1743, Jul. 2015.

[14] T. Limbasiya and D. Das, “Secure message confirmation scheme based on batch verification in vehicular cloud computing,” *Phys. Commun.*, vol. 34, pp. 310–320, Jun. 2019.

[15] D. He, C. Chen, S. Chan, and J. Bu, “Analysis and improvement of a secure and efficient handover authentication for wireless networks,” *IEEE Commun. Lett.*, vol. 16, no. 8, pp. 1270–1273, Aug. 2012.

[16] S. L. Yeo, W.-S. Yap, J. K. Liu, and M. Henricksen, “Comments on’ analysis and improvement of a secure and efficient handover authentication based on bilinear pairing functions,”’ *IEEE Commun. Lett.*, vol. 17, no. 8, pp. 1521–1523, May 2013.

[17] D. He, M. K. Khan, and N. Kumar, “A new handover authentication protocol based on bilinear pairing functions for wireless networks,” *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 18, nos. 1–2, pp. 67–74, 2015.

[18] W. Wang and L. Hu, “A secure and efficient handover authentication protocol for wireless networks,” *Sensors*, vol. 14, no. 7, pp. 11379–11394, Jun. 2014.

[19] J.-L. Tsai, N.-W. Lo, and T.-C. Wu, “Secure handover authentication protocol based on bilinear pairings,” *Wireless Pers. Commun.*, vol. 73, no. 3, pp. 1037–1047, Dec. 2013.

[20] C. Wang, Y. Yuan, and J. Wu, “A new privacy-preserving handover authentication scheme for wireless networks,” *Sensors*, vol. 17, no. 6, p. 1446, Jun. 2017.

[21] S. A. Chaudhry, M. S. Farash, H. Naqvi, S. H. Islam, and T. Shon, “A robust and efficient privacy aware handover authentication scheme for wireless networks,” *Wireless Pers. Commun.*, vol. 93, no. 2, pp. 311–335, Mar. 2017.

[22] D. He, S. Zeadally, L. Wu, and H. Wang, “Analysis of handover authentication protocols for mobile wireless networks using identity-based public key cryptography,” *Comput. Netw.*, vol. 128, pp. 154–163, Dec. 2017.

[23] D. He, S. Chan, and M. Guizani, “Handover authentication for mobile networks: Security and efficiency aspects,” *IEEE Netw.*, vol. 29, no. 3, pp. 96–103, May 2015.

[24] G. Li, Q. Jiang, F. Wei, and C. Ma, “A new privacy-aware handover authentication scheme for wireless networks,” *Wireless Pers. Commun.*, vol. 80, no. 2, pp. 581–589, Jan. 2015.

[25] K.-A. Shim, “Security analysis of various authentication schemes based on three types of digital signature schemes,” *IEEE Access*, vol. 6, pp. 68804–68812, 2018.

[26] T. Gao, Q. Wang, X. Wang, and X. Gong, “An anonymous access authentication scheme based on proxy ring signature for CPS-WMNs,” *Mobile Inf. Syst.*, vol. 2017, pp. 1–11, Jun. 2017.

[27] T. Gao, X. Deng, Y. Wang, and X. Kong, “PAAS: PMIPv6 access authentication scheme based on identity-based signature in VANETs,” *IEEE Access*, vol. 6, pp. 37480–37492, 2018.

[28] T. Gao, X. Deng, N. Guo, and X. Wang, “An anonymous authentication scheme based on PMIPv6 for VANETs,” *IEEE Access*, vol. 6, pp. 14686–14698, 2018.

[29] L. Lei, W. Zhang, Y. Wang, and X. Wang, “A pairing-free identity-based handover AKE protocol with anonymity in the heterogeneous wireless networks,” *Int. J. Commun. Syst.*, vol. 32, no. 12, p. e4000, Aug. 2019.

[30] R. Ma, J. Cao, D. Feng, H. Li, Y. Zhang, and X. Lv, “PPSHA: Privacy preserving secure handover authentication scheme for all application scenarios in LTE—A networks,” *Ad Hoc Netw.*, vol. 87, pp. 49–60, May 2019.

[31] K. Xue, H. Zhou, W. Meng, D. S. L. Wei, and M. Guizani, “A lightweight and secure group key based handover authentication protocol for the software-defined space information network,” *IEEE Trans. Wireless Commun.*, early access, Feb. 28, 2020, doi: 10.1109/TWC.2020.2975781.

[32] D. Eckhardt and P. Steenkiste, “Measurement and analysis of the error characteristics of an in-building wireless network,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 4, pp. 243–254, Oct. 1996.

[33] D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing,” in *Proc. Annu. Int. Cryptol. Conf.*, Santa Barbara, CA, USA, 2001, pp. 213–229.

[34] S. D. Galbraith, K. Harrison, and D. Soldera, “Implementing the Tate pairing,” in *Proc. Algorithmic Number Symp.*, Sydney, NSW, Australia, 2002, pp. 324–337.

[35] M. Scott. *Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL)*. [Online]. Available: http://www.shamus.ie/

[36] H. Hartenstein and K. Laberteaux. *VANET: Vehicular Applications and Inter-Networking Technologies*, vol. 1. Hoboken, NJ, USA: Wiley, 2009. [Online]. Available: https://ieeexplore.ieee.org/book/8039897

YOUNSOO PARK received the B.E. and M.E. degrees from the School of Electrical and Electronics Engineering, Chung-Ang University, Seoul, South Korea, in 2014 and 2016, respectively, where he is currently pursuing the Ph.D. degree in engineering. His research interests include computer networks, big data thinking education, computational thinking education, and elliptic curve cryptography (ECC).

HO-HYUN PARK received the B.S. degree from Seoul National University, in 1987, and the M.S. and Ph.D. degrees in computer science and engineering from KAIST, in 1995 and 2001, respectively. From 1987 to 2003, he was a Principal Engineer at Samsung Electronics. He is currently a Professor of electrical and electronics engineering with Chung-Ang University. His research interests include big data, deep learning, machine vision, information security, and real-time and embedded systems.