Journal of Southeast University (English Edition) Vol. 39, No. 1, pp. 81 −88 Mar. 2023 ISSN 1003—7985

Gradient parameter data integrity protection scheme in the Ring Allreduce architecture

Fang Yunan Jiang Rui

(School of Cyber Science and Engineering, Southeast University, Nanjing 210096, China)

Abstract: As there is no research on protecting gradient parameter data integrity in the Ring Allreduce architecture, a Ring Allreduce architecture oriented gradient parameter data integrity protection scheme (RAA-DIP) is proposed. The identity-based group key agreement algorithm and the Boneh-Lynn-Shacham signature are used to protect the integrity of gradient parameter data in Ring Allreduce (RAA). Combined with identity authentication and the key negotiation algorithm, secure and efficient dynamic management of working nodes is realized. On the basis of the decisional bilinear Diffie-Hellman problem, secure group key negotiation is implemented so that the key generation center or network attackers cannot calculate the shared secret of worker nodes, which solves the key escrow problem and ensures the integrity of transmission gradient data. Finally, the RAA-DIP scheme is formally proved, and its simulation performance is compared with those of related schemes. The results show that the RAA-DIP scheme can guarantee the integrity of the gradient parameter data transmission process in Ring Allreduce, realize the dynamic management of working nodes, and solve the problem of key escrow. Compared with related schemes, it can meet security and performance requirements.

Key words: distributed machine learning; data integrity; group key agreement; Ring Allreduce architecture

DOI: 10.3969/j.issn.1003 −7985.2023.01.010

In the era of big data, artificial intelligence (AI), as a representative of emerging technology, has been the focus of social attention. AI technology has been widely used in production and the tasks of daily life, such as navigation, automatic driving, and intelligent recommendations. Machine learning (ML) is a fundamental way to realize AI. With the rising demand of society for AI capability, distributed ML has been proposed to run ML tasks on multiple nodes in parallel. The Ring Allreduce[1] architecture is widely used for data-parallel distributed communication. In this architecture, all working nodes construct a ring communication topology, where each working node receives parameter data from the upper node and transmits parameter data to the lower node on the ring. Because the Ring Allreduce architecture shows satisfactory performance for dense models[2], recent research has adopted the Ring Allreduce structure as its distributed training architecture[3-6]. However, in the Ring Allreduce architecture, if some working nodes fail, the performance of the system will substantially deteriorate. To solve this problem, CodedReduce[7] redundantly distributes the dataset among working nodes so that if some working nodes fail, the total gradient parameter data can still be recovered. To fully use the bandwidth for reducing the network delay and communication cost for large-scale working nodes, a hierarchical Allreduce algorithm[8] has been proposed, in which working nodes are divided into multiple groups, and the hierarchical Allreduce algorithm is executed among multiple groups. 2D-Torus Allreduce[9] arranges working nodes in a 2D grid to reduce communication costs. However, to date, no research focused on investigating the integrity of parameter data during the communication transmission process in the Ring Allreduce architecture.

Typically, to protect data integrity, cryptography algorithms should be applied in network communication systems, particularly in a group communication system such as the Ring Allreduce system. Hence, many schemes for data integrity protection in group communication have been proposed. To ensure the security and integrity of messages in group communication, some studies[10-11] have been proposed based on public key infrastructure (PKI) cryptosystems. SeDS[10] is a secure data sharing and transmission scheme with a PKI-based digital signature and a mutual authentication mechanism. However, this scheme has a certificate management problem. A scheme for a group leader to manage keys[11] was pro-

Received 2022-10-07, **Revised** 2023-01-15.

Biographies: Fang Yunan (1998—), female, graduate; Jiang Rui (corresponding author), male, doctor, professor, R.Jiang@seu.edu.cn.

Foundation items: The National Natural Science Foundation of China (No. 61372103), the Natural Science Foundation of Jiangsu Province (No. BK20201265), Foundation of the National Engineering Research Center of Classified Protection and Safeguard Technology for Cybersecurity (No. C21640-2).

Citation: Fang Yunan, Jiang Rui. Gradient parameter data integrity protection scheme in the Ring Allreduce architecture[J]. Journal of Southeast University (English Edition), 2023, 39(1): 81 −88. DOI: 10.3969/j.issn.1003 −7985.2023.01.010.

posed, but in this scheme, the group leader could control the security keys, possibly causing the key escrow problem. ID-based AGKA[12] was the first ID-based GKA protocol based on an extension of the scheme[13]. It alleviates the expensive overhead in PKI-based protocols. However, the above schemes[10-13] cannot be applied in the dynamic group. Tree key structures have been further proposed in schemes[14-15] for dynamic management, but the maximum cost would reach $O(\log n)$ rounds of interaction for members joining or leaving, where n is the number of group members. PPAKA-HMAC and PPAKA-IBS[16] were proposed for secure dynamic group communication, but these two schemes must rely on secure channels in the process of dynamic group membership management. Because the above schemes[10-16] have these shortcomings, they cannot be directly applied to the Ring Allreduce architecture. Hence, a scheme for guaranteeing the integrity of transmitted gradient parameter data in the Ring Allreduce architecture must be designed.

1 Preliminaries

1.1 Bilinear mapping

Definition 1 Let G_1, G_2 be multiplicative cyclic groups of a large prime order q, and a map function e: $G_1 \times G_1 \rightarrow G_2$ is a bilinear pairing. It satisfies the following properties: 1) Bilinear: For $\forall u, v \in G_1$, $\forall a, b \in Z_q$, $e(u^a, v^b) = e(u, v)^{ab}$. 2) Non-Degeneracy: For $\exists u, v \in G_1$, there is $e(u, v) \neq 1$. 3) Computability: For $\forall u, v \in G_1$, there exists an efficient algorithm to compute $e(u, v)$.

1.2 Computational Diffie-Hellman (CDH) assumption[12]

Definition 2 Given g^a, $g^b \in G_1$, where $a, b \in Z_q^*$, for any polynomial-time adversary, it is feasible to computationally solve $g^{ab} \in G_1$. More formally, the advantage of a probabilistic polynomial time (PPT) algorithm ξ in solving the CDH problem is defined as follows.

$$A_{\mathrm{CDH}} = \left| P_{\mathrm{r}}[\xi(g, g^a, g^b) = g^{ab}: a, b \in Z_q^*] \right|$$

The CDH assumption is satisfied when A_{CDH} is negligible for any PPT algorithm ξ.

1.3 Decisional bilinear Diffie-Hellman (DBDH) assumption[12]

Definition 3 Given $g^a, g^b, g^c \in G_1$, for any polynomial-time adversary, where $a, b, c, d \in Z_q^*$, it is feasible to computationally distinguish $(g, g^a, g^b, g^c, e(g, g)^{abc})$ and $(g, g^a, g^b, g^c, e(g, g)^d)$. More formally, the advantage of a PPT algorithm ξ in solving the problem is defined as follows.

$$A_{\mathrm{DBDH}} = \left| P_{\mathrm{r}}[\xi(g, g^a, g^b, g^c, e(g, g)^{abc}) = 1: a, b, c \in Z_q^*] - P_{\mathrm{r}}[\xi(g, g^a, g^b, g^c, e(g, g)^d) = 1: a, b, c, d \in Z_q^*] \right|$$

The DBDH assumption is satisfied when A_{DBDH} is negligible for any ξ.

1.4 Co-Computational bilinear Diffie-Hellman (Co-CDH) assumption[17]

Definition 4 Given $g, g^a \in G_1$, $u \in G_2$, where $a \in Z_q^*$, for any polynomial-time adversary, it is feasible to computationally solve $u^a \in G_2$. More formally, the advantage of a PPT algorithm ξ in solving the problem is defined as follows:

$$A_{\text{Co-CDH}} = P_{\mathrm{r}}[\xi(g, g^a, u) = u^a: a \in Z_q^*]$$

The Co-CDH assumption is satisfied when $A_{\text{Co-CDH}}$ is negligible for any ξ.

2 Proposed RAA-DIP Scheme

2.1 System model

The system model is presented in Fig. 1. The system contains two entities: the key generation center (KGC) and the working nodes. The KGC is responsible for registering working nodes, issuing corresponding private keys for working nodes according to their identity, and assisting working nodes in negotiating group private keys. Working nodes honestly perform model training, generate the correct signatures for gradient parameter data, and transmit them to the next working node.

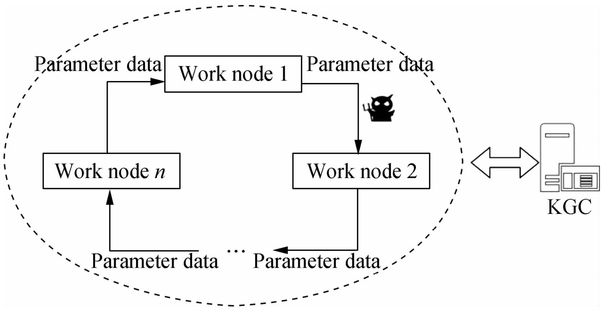

Fig. 1 System model for our RAA-DIP scheme

2.2 Details of our RAA-DIP scheme

In this section, we describe the detailed construction of our RAA-DIP scheme. It includes five phases: system setup, work node registration, group key agreement, secure parameter data transmission, and dynamic management.

2.2.1 System setup

In the system setup phases, KGC picks a random number $\alpha \in Z_q^*$ as a system master key and computes $P_{\mathrm{pub}} = g^{\alpha}$

as a system public key. KGC keeps α secret and publishes the system parameter data $\{G_1, G_2, e, u, g, H, H_1, H_2, h, P_{\text{pub}}\}$, where G_1, G_2 are two multiplicative cyclic groups with the same large prime order q, e: $G_1 \times G_1 \rightarrow G_2$ is an efficiently computable bilinear map, u is an element randomly selected from G_1, g is a generator of G_1, and H, H_1, H_2, h are four cryptographic hash functions that obey $H(\cdot), h(\cdot): \{0,1\}^* \rightarrow G_1$, $H_1(\cdot): \{0,1\}^* \rightarrow Z_q^*$, and $H_2(\cdot): G_2 \rightarrow Z_q^*$.

2.2.2 Work node registration

First, the i-th working node ω_i sends its identity I_i to the KGC for registration. Then, the KGC computes the key pair (p_i, s_i) for ω_i, where $p_i = H(I_i) \in G_1$ is a public key and $s_i = p_i^{\alpha} \in G_1$ is a private key. KGC stores the work node information in the registration list and sends s_i to ω_i through a secure channel. Finally, ω_i verifies the correctness of the private key from the KGC with p_i as follows:

$$e(s_i, g) = e(p_i, P_{\text{pub}}) \tag{1}$$

If this equation holds, then the private key received from the KGC is correct, and ω_i accepts it and saves the private key. Otherwise, ω_i discards it, terminates the protocol, delivers an alarm, and makes the registration again.

2.2.3 Group key agreement

In this phase, working nodes negotiate with each other to share a session key. The entire phase includes session requests and session establishment.

1) Session request. Assume that a set of n working nodes is involved in training. Each ω_i sends session request information $L_i^{\text{req}} = \{I_i\}$ to the KGC. The KGC first checks whether the identity is legal and rejects the session request if it is not. When the KGC receives all requests, it initiates a group session and chooses a random number $\eta_{\text{sid}} \in Z_q^*$ as the session identifier. Then, the KGC creates a ring group structure $R_0 = \{I_1, I_2, \cdots, I_n\}$ based on the identity of ω_i. Finally, the KGC broadcasts $\{\eta_{\text{sid}}, R_0\}$. In the system, ω_{i-1} and ω_{i+1} are the left and right work nodes of ω_i, respectively. Specifically, there is $\omega_{-1} = \omega_{n-1}$ and $\omega_{n-1} = \omega_1$.

2) Session establishment. After receiving the message $\{\eta_{\text{sid}}, R_0\}$ from the KGC, working nodes initiate the process of two rounds of mutual authentication and key negotiation to obtain a shared session key.

Round 1 First, each ω_i has a secret x_i, where x_i is a random value. Each ω_i generates its first key agreement message $M_i = I_i \parallel X_i \parallel \eta_{\text{sid}} \parallel 1$, where $X_i = g^{x_i}$ is the first key hint, and parameter "1" indicates the sequence of messages from ω_i. Second, each ω_i generates a signature $\sigma_i = P_{\text{pub}}^{x_i} s_i^{h_i}$ for M_i, where $h_i = H_1(M_i)$. Finally, each ω_i sends $\{M_i, \sigma_i\}$ to ω_{i-1} and ω_{i+1}.

Round 2 After ω_i receives message $\{M_{i-1}, \sigma_{i-1}\}$, $\{M_{i+1}, \sigma_{i+1}\}$ from ω_{i-1} and ω_{i+1}, respectively, each ω_i first checks the message to verify the correctness of I, η_{sid}, and the sequence parameter. Then, each ω_i performs signature verification. The verification equation is

$$e\left(\prod_{k\in\{-1,1\}} \sigma_{i+k}, g\right) = e\left(\prod_{k\in\{-1,1\}} X_{i+k} + p_{i+k}^{h_{i+k}}, P_{\text{pub}}\right) \tag{2}$$

If the verification equation (2) holds, ω_i generates $\{M_i', \sigma_i'\}$ and broadcasts, where $M_i' = I_i \parallel Y_i \parallel V_i' \parallel \eta_{\text{sid}} \parallel 2$ is the second key agreement message, $Y_i = \dfrac{e(X_{i+1}^{x_i}, P_{\text{pub}})}{e(X_{i-1}^{x_i}, P_{\text{pub}})}$ is the second key hint, $V_i' = g^{y_i}$ with a random number $y_i \in Z_q^*$ is used to help the verification, and $\sigma_i' = P_{\text{pub}}^{y_i} s_i^{h_i'}$ is the signature for M_i', with $h_i' = H_i(M_i')$.

Key generation After each ω_i receives $\{M_j', \sigma_j'\}$ from ω_j, where $j = 1, 2, \cdots, n, j \neq i$, each ω_i first checks the message to verify the correctness of I, η_{sid}, and the sequence parameter. Then, each ω_i performs signature verification. The verification equation is

$$e\left(\prod_{j=1}^{n, j\neq i} \sigma_j', g\right) = e\left(\prod_{j=1}^{n, j\neq i} V_j' p_j^{h_j'}, P_{\text{pub}}\right) \tag{3}$$

If the verification equation (3) holds, each ω_i then computes a negotiated secret message

$$\begin{aligned}\kappa_i = e(X_{i-1}^{x_i}, P_{\text{pub}})^n Y_i^{n-1} Y_{i+1}^{n-2} \cdots Y_{i-2} = \\ e(g,g)^{\alpha(x_1x_2 + x_2x_3 + \cdots + x_nx_1)}\end{aligned} \tag{4}$$

Finally, each ω_i can compute the same session key pair $(o_{\text{sk}}, o_{\text{pk}})$, where $o_{\text{sk}} = H_2(\kappa_i)$ is the session secret key, and $o_{\text{pk}} = g^{o_{\text{sk}}}$ is the corresponding session public key.

2.2.4 Secure parameter data transmission

In this phase, the system starts training the model and passing gradient parameter data. The entire phase includes signature and verification.

1) Signature. Assume the gradient parameter data file $F = (m_1, m_2, \cdots, m_i, \cdots, m_k)$, where m_i is the i-th data block of the file, and k is the number of data blocks. For each block m_i, ω_i generates the corresponding signature $T_i = (h(\eta_{\text{name}} \parallel i) u^{m_i + h(m_i \parallel \tau)})^{o_{\text{sk}}}$, where η_{name} is the unique identifier of the file F, and τ is a timestamp. ω_i sends $\{F, T, \tau\}$ to ω_{i+1}, where $T = \prod_{i=1}^{k} T_i$.

(2) Verification. After ω_{i+1} receives message $\{F, T, \tau\}$, ω_{i+1} first verifies the timestamp, computes $M = \sum_{i=1}^{k} m_i + \sum_{i=1}^{k} h(m_i \parallel \tau)$, and performs signature verification. The verification equation is

$$e(T, g) = e\left(\prod_{i=1}^{k} h(\eta_{\text{name}} \parallel i) u^M, o_{\text{pk}}\right) \tag{5}$$

If Eq. (5) holds, then the gradient parameter data are intact from the working node ω_i. Otherwise, ω_{i+1} discards the information.

2.2.5 Dynamic management

If the system executes a complete group key agreement protocol every time the working node dynamically changes, the communication and computational costs will be very high. Hence, secure and efficient dynamic work node management in the Ring Allreduce architecture must be realized.

1) Working node joining. Let $G = \{\omega_1, \omega_2, \cdots, \omega_n\}$ be the current group. If there is a new working node ω_{n+1} joining the ring, it first reports to the KGC about its join. After ω_{n+1} completes its registration with the KGC, the KGC broadcasts $\{\eta_{\text{sid}}, R'_0\}$, where $R'_0 = \{I_1, I_2, \cdots, I_n, I_{n+1}\}$. We denote h_2 as a cryptographic hash function $h_2(\cdot): G_1 \to Z_q^*$, which can be readily found in numerous standard documents. Next, ω_1 executes key agreement protocol ω_{n+1}. ω_1 sends $\{U_1, V_1, N\}$ to ω_{n+1}, where $U_1 = g^{o_{\text{sk}}}$, $V_1 = s_1^{o_{\text{sk}}^{-1} h_2(U_1)}$, and $N \in Z_q^*$ is a random number. After ω_{n+1} receives the message, ω_{n+1} checks $e(U_1, V_1) = e(P_{\text{pub}}, p_1^{h_2(U_1)})$. If the verification passes, ω_{n+1} picks the random value $x_{n+1} \in Z_q^*$ and computes the shared secret key $o_{\text{sk}}^+ = H_2(e(U_1, P_{\text{pub}}^{x_{n+1}}))$. Then, ω_{n+1} sends $\{U_{n+1}, V_{n+1}, E_{o_{\text{sk}}^+}(N)\}$ to ω_1, where $U_{n+1} = g^{x_{n+1}}$, $V_{n+1} = s_{n+1}^{h_2(U_{n+1}) x_{n+1}^{-1}}$, and $E_{o_{\text{sk}}^+}(N)$ is cipher text with the encryption function $E_{o_{\text{sk}}^+}$. After ω_1 receives the message, ω_1 checks $e(U_{n+1}, V_{n+1}) = e(P_{\text{pub}}, p_{n+1}^{h_2(U_{n+1})})$. If the verification passes, ω_1 computes the shared secret key by $o_{\text{sk}}^+ = H_2(e(U_{n+1}, P_{\text{pub}}^{o_{\text{sk}}}))$, decrypts $E_{o_{\text{sk}}^+}(N)$ to verify value N with decryption function $D_{o_{\text{sk}}^+}(N)$, encrypts $N+1$, and sends message $\{E_{o_{\text{sk}}^+}(N+1)\}$ to ω_{n+1}. ω_{n+1} decrypts this message to verify value $N+1$. Thus, ω_1 and ω_{n+1} confirm the new session key o_{sk}^+. Finally, ω_1 uses original session key o_{sk} to encrypt and broadcast a new session key o_{sk}^+ to ω_i, where $i = 2, 3, \cdots, n$.

2) Working node revocation. Let $G = \{\omega_1, \omega_2, \cdots, \omega_n\}$ be the current group. If the working node ω_i leaves the ring, we let $G' = G \setminus G^- = \{\omega_1, \omega_2, \cdots, \omega_{i-1}\} \cup \{\omega_{i+1}, \omega_{i+2}, \cdots, \omega_n\}$ be a set of all remaining members. First, the KGC removes the ω_i information from the registration list and the ring. The KGC creates a new ring group structure based on G' and broadcasts $\{\eta_{\text{sid}}, R'_0\}$. Next, $\omega_{i-1}, \omega_{i+1}$ calculate the key hint $X_{i-1} = g^{r_{i-1}}$, $X_{i+1} = g^{r_{i+1}}$, where $r_{i-1}, r_{i+1}, \in Z_q^*$ are random values. According to the first round of session establishment, ω_{i-1} generates message $\{M_{i-1}, \sigma_{i-1}\}$ and sends it to ω_{i-2} and ω_{i+1}. Similarly, ω_{i+1} generates message $\{M_{i+1}, \sigma_{i+1}\}$ and sends it to ω_{i-1} and ω_{i+2}. According to the second round of session establishment, upon receiving and verifying the integrity of the messages, $\omega_{i-2}, \omega_{i-1}, \omega_{i+1}, \omega_{i+2}$ compute their second agreement message and corresponding signature. The remaining working nodes use the original message and the corresponding signature. Finally, each ω_l broadcasts $\{M'_l, \sigma'_l\}$, where $l = 1, 2, \cdots, i-1, i+1, \cdots, n$. Each working node computes a new shared secret message $\kappa'_i = e(g, g)^{\alpha(x_1x_2 + x_2x_3 + \cdots + x_{i-2}r_{i-1} + r_{i-1}r_{i+1} + r_{i+1}x_{i+2} + \cdots + x_nx_1)}$ and obtains a new session key $o_{\text{sk}}^- = H_2(\kappa'_i)$.

2.3 Correctness

Theorem 1 Our Ring Allreduce architecture-oriented gradient parameter data integrity protection scheme (RAA-DIP) is correct.

Proof ω_i can verify the correctness of the received private key generated by the KGC according to Eq. (1) as follows:

$$e(s_i, g) = e(p_i^{\alpha}, g) = e(p_i, g^{\alpha}) = e(p_i, P_{\text{pub}})$$

Our scheme can correctly run to verify the integrity of the first round of agreement messages according to Eq. (2) as follows:

$$e\Big(\prod_{k \in \{-1,1\}} \sigma_{i+k}, g\Big) = e\Big(\prod_{k \in \{-1,1\}} P_{\text{pub}}^{x_{i+k}} s_{i+k}^{h_{i+k}}, g\Big) =$$
$$e\Big(\prod_{k \in \{-1,1\}} g^{x_{i+k}} p_{i+k}^{h_{i+k}}, g^{\alpha}\Big) = e\Big(\prod_{k \in \{-1,1\}} X_{i+k} p^{h_{i+k}}, P_{\text{pub}}\Big)$$

Eqs. (2) and (3) have the same proof process. After verifying the integrity of the agreement message, each working node can correctly generate the session key $o_{\text{sk}} = H_2(\kappa_i)$ according to Eq. (4) as follows:

$$\kappa_i = e(X_{i-1}^{x_i}, P_{\text{pub}})^n Y_i^{n-1} Y_{i+1}^{n-2} \cdots Y_{i-2} =$$
$$e(X_{i-1}^{x_i}, P_{\text{pub}})^n \frac{e(X_{i+1}^{x_i}, P_{\text{pub}})^{n-1}}{e(X_{i-1}^{x_i}, P_{\text{pub}})^{n-1}} \frac{e(X_{i+2}^{x_{i+1}}, P_{\text{pub}})^{n-2}}{e(X_i^{x_{i+1}}, P_{\text{pub}})^{n-2}} \cdots$$
$$\frac{e(X_{i+(n-1)}^{x_{i+(n-2)}}, P_{\text{pub}})}{e(X_{i+(n-3)}^{x_{i+(n-2)}}, P_{\text{pub}})} =$$
$$e(X_{i-1}^{x_i}, P_{\text{pub}}) e(X_{i+1}^{x_i}, P_{\text{pub}}) e(X_{i+2}^{x_{i+1}}, P_{\text{pub}}) \cdots$$
$$e(X_{i+(n-1)}^{x_{i+(n-2)}}, P_{\text{pub}}) = e(g, g)^{\alpha(x_{i-1}x_i + x_ix_{i+1} + \cdots + x_{i-2}x_{i-1})} =$$
$$e(g, g)^{\alpha(x_1x_2 + x_2x_3 + \cdots + x_nx_1)}$$

Then, we can prove that our RAA-DIP scheme can correctly run to verify the integrity of the parameter data according to Eq. (5) as follows:

$$e(T, g) = e\Big(\prod_{i=1}^{k} (h(\eta_{\text{name}} \| i) u^{m_i + h(m_i \| \tau)})^{o_{\text{sk}}}, g) =$$
$$e\Big(\prod_{i=1}^{k} h(\eta_{\text{name}} \| i) u^{\sum_{i=1}^{k} (m_i + h(m_i \| \tau))}, g^{o_{\text{sk}}}\Big) =$$
$$e\Big(\prod_{i=1}^{k} h(\eta_{\text{name}} \| i) u^{M}, o_{\text{pk}})$$

Theorem 1 proves that our RAA-DIP scheme is correct.

3 Security Analysis

3.1 Formal proof

In this section, we formally prove that our scheme can securely negotiate session keys and generate signatures to ensure the integrity of gradient parameter data, realize secure dynamic group management, and solve the key escrow problem.

Lemma 1 The advantage of forging a signature to pass the working nodes' verification in the group key agreement phase is $\varepsilon' \geqslant \varepsilon(1-1/q_E)/q$, where the adversary wins the game by advantage ε, and q_E is the queries to the Extract.

Lemma 2 The advantage of computing the session key from the agreement message is $\varepsilon_1 \leqslant 2n(q_E + q_S) A_{DBDH}(t)$, where $A_{DBDH}(t)$ represents the maximum advantage of an adversary for solving the DBDH problem, which is negligible.

Theorem 2 In our RAA-DIP scheme, the group key agreement phase is secure.

Proof In the group key agreement phase, the adversary may attack the group key agreement in two ways, forging an authentication signature message and computing the session key from the transmission message. Let $A_{DBDH}(t) = \varepsilon_0$. Combining Lemma 1 and Lemma 2, after the adversary makes q_E, q_S queries to the Execute and Send, respectively, the maximum advantage of the adversary in attacking our scheme key negotiation process is

$$A_{adv}(t,q_E,q_S) \leqslant A_{\Gamma}^{Forge}(t) + 2n(q_E+q_S)A_{DBDH}(t) = \varepsilon' + 2n(q_E+q_S)\varepsilon_0$$

Therefore, the adversary cannot break our scheme. Theorem 2 proves that our scheme can realize a secure key agreement. That is, a network adversary cannot determine the session key and break the scheme.

Theorem 3 In our RAA-DIP scheme, it is computationally infeasible for a network adversary to break the integrity of the gradient parameter data transmission phase. Suppose an algorithm can tamper or forge a signature. Then, we can find an adversary that can solve the Co-CDH problem.

Proof In the secure parameter data transmission phase, a network adversary may tamper or forge a signature to break the integrity of gradient parameter data. For the correct (F,T) signature, it can pass the equation as $e(T,g) = e\left(\prod_{i=1}^{k} h(\eta_{name} \| i) u^{M}, o_{pk}\right)$. Assuming that the adversary tampers with or forges the signature (F', T'), which can be verified by the integrity of the working nodes, it can still pass the equation as $e(T',g) = e\left(\prod_{i=1}^{k} h(\eta_{name} \| i) u^{M'}, o_{pk}\right)$. Then, $e(T'/T,g) = e(u^{M'-M}, g^{o_{sk}})$ can be obtained according to the two equations above. Letting $\Delta M = M' - M$, we can obtain $u^{o_{sk}} = (T'/T)^{1/\Delta M}$. According to Definition 4, given $a \in Z_q^*$, $g, g^a \in G_1$, and $u \in G_2$ as input, if output $u^a \in G_2$, then the Co-CDH problem is solved. In our scheme, u, g are public, and $g^{o_{sk}}$ is the public key. For a given $g, g^{o_{sk}}, u$, we can calculate $u^{o_{sk}}$ through the above steps. Thus, there is an algorithm that can solve the Co-CDH problem, which contradicts Definition 4. Therefore, it is computationally infeasible for a network adversary to break the integrity of the parameter data transmission phase. Theorem 3 proves that our RAA-DIP scheme can realize a secure gradient parameter data transmission.

Theorem 4 Our RAA-DIP scheme can achieve secure dynamic group member management.

Theorem 5 Our RAA-DIP scheme can solve the key escrow problem. Suppose the KGC makes at most q_E, q_S queries to the Execute and Send, respectively. The advantage of the KGC calculating the correct session key κ is $\varepsilon_1' \leqslant 2n(q_E+q_S)A_{DBDH}(t)$, which is negligible.

3.2 Security comparison

Tab. 1 compares the security performance of RAA-DIP with those of SeDS[10], Lopes's scheme[11], ID-based AGKA[12], and PPAKA-IBS[16]. According to Tab. 1, our RAA-DIP scheme can resolve forgery attacks, achieve secure key agreement, realize dynamic group management, and resolve the key escrow problem, unlike other schemes. Therein, "√" and "×" mean that the scheme can and cannot meet the corresponding security goal, respectively.

Tab. 1 Security comparison

Scheme	SeDS[10]	Lopes's scheme[11]	ID-based AGKA[12]	PPAKA-IBS[16]	RAA-DIP
Resolve forgery attack	√	√	×	√	√
Secure key agreement	√	√	×	√	√
Dynamic group management	×	√	×	×	√
Resolve key escrow	√	×	√	√	√

4 Performance Evaluation

In this section, we analyze the performance of our protocol in terms of computational, communication, and storage costs and compare it with other schemes[10-12,16]. We run the scheme on our 64-bit Intel@ Core 2.30 GHz desktop PC.

We apply the MNT curve of pairing-based cryptography library version 0.5.4 for the basic pairing algorithm.

4.1 Computational cost

In this section, we evaluate the computational cost during the execution of the key agreement and compare itwith those of other schemes[10–12,16]. As SeDS is a two-party protocol, to compare it with other schemes, we run SeDS $n(n-1)/2$ times. Tab. 2 gives the concrete computational cost and computational complexity of all schemes, where O_R, O_E, O_H, O_M, O_P denote the randomselection, exponent operation, hash function operation, multiplication operation, and pair operation, respectively. $O(\cdot)$ is a function of computational complexity. n is the number of working nodes in the system.

Tab. 2 Computational cost comparison

Scheme	Computational cost	Computational complexity
RAA-DIP	$3O_R + (2n+4)O_H + (2n+9)O_E + 7O_P + (4n+1)O_M$	$O(n)$
SeDS[10]	$1/2n(n-1)(3O_R + 5O_E + 2O_H + 4O_P)$	$O(n^2)$
Lopes's scheme[11]	$3O_R + (3n+6)O_H + (8n-2)O_E + 2O_P + (3n-1)O_M$	$O(n)$
ID-based AGKA[12]	$2O_R + 4O_H + (n+8)O_E + 4O_P + (n+8)O_M$	$O(n)$
PPAKA-IBS[16]	$3O_R + (2n+6)O_H + (n+10)O_E + (n+2)O_P + 5nO_M$	$O(n)$

We simulate the computational costs in Fig. 2. The costs of the main computations of pairing and exponential operations can be estimated as 5.86 and 0.78 ms, respectively[18]. As n increases, the computational costs of different schemes increase. The computational cost of SeDS is higher than that of our RAA-DIP scheme. This result is obtained because the computational cost of SeDS increases with complexity $O(n^2)$, while the computational cost of RAA-DIP increases with complexity $O(n)$. Our RAA-DIP scheme has a much lower computational cost compared to Lopes et al. [11] and PPAKA-IBS. The computational cost of ID-based AGKA is lower than that of our RAA-DIP scheme. However, ID-based AGKA may suffer from man-in-the-middle attacks.

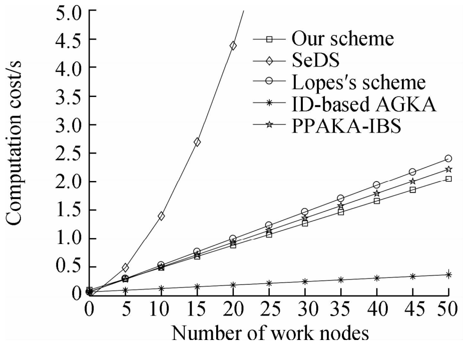

Fig. 2 Computational costs of different schemes

4.2 Communication cost

To analyze the communication costs of all related schemes, we set the specific size of each parameter transferred in the protocols. The size of the identity is 16 bytes, and η_{sid} is 8 bytes. The other parameters are set as 20 bytes. Fig. 3 compares the communication costs for the key agreement among the five schemes.

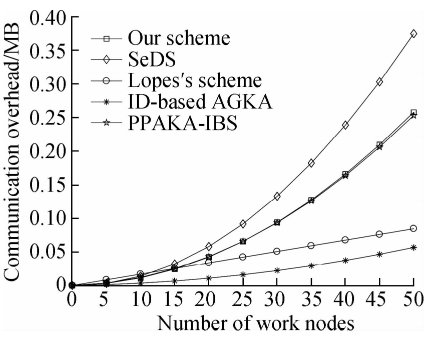

Fig. 3 Communication costs of different schemes

The communication cost of our RAA-DIP scheme ($101n^2 + 105n$) is lower than that of SeDS, which is $153n^2 - 153n$, but higher than that of Lopes et al., which is $153n^2 - 153n$. However, Lopes et al. have a key escrow problem, which may cause key leakage and data integrity compromise. The communication cost of our RAA-DIP scheme is higher than that of ID-based AGKA., which is $20n^2 + 136n$. However, ID-based AGKA may suffer from a man-in-the-middle attack, in which the adversary forges a message to attack the scheme. The communication cost of our RAA-DIP scheme is higher than that of PPAKA-IBS, which is $97n^2 + 109n$. However, PPAKA-IBS needs secure channels and does not realize secure dynamic group management.

4.3 Storage cost

In this section, we evaluate the storage cost and compare it with those of related schemes[10–12,16]. Generally, we focus on the storage cost in the registration and key agreement phases. Because SeDS is the two-party protocol, in the following simulation, we take the average of the storage capacity of two entities as the result. The results are shown in Tab. 3.

Tab. 3 Storage cost comparison

Scheme	Entity	Entity leader	KGC	Storage cost
RAA-DIP	$85n+201$		$16n$	$101n+201$
SeDS[10]	244		$60n$	$60n+244$
Lopes's scheme[11]	$56n+84$	$76n-20$	$16n$	$148n+64$
ID-based AGKA[12]	$40n+140$		$16n$	$56n+140$
PPAKA-IBS[16]	$85n+217$		$32n$	$117n+217$

The storage cost of SeDS is lower than that of our scheme because $60n+244<101n+201$. However, SeDS is a two-party protocol. The storage cost of Lopes et al. is higher than that of our scheme because $148n+64>101n+201$. The storage cost of ID-based AGKA is lower than that of our scheme because $56n+140<101n+201$. However, ID-based AGKA may suffer from man-in-the-middle attacks, in which the adversary forges a message to attack the scheme. The storage cost of PPAKA-IBS is higher than that of our scheme because $117n+217>101n+201$.

5 Conclusions

1) In the RAA-DIP scheme, the integrity of transmission parameter data in the Ring Allreduce architecture is protected, thus ensuring the correctness of ML training results.

2) The RAA-DIP scheme can realize secure and efficient dynamic group management.

3) The RAA-DIP scheme can solve the key escrow problem.

References

[1] Gibiansky A. Bringing HPC techniques to deep learning [EB/OL]. (2017-02-21) [2022-12-05]. https://andrew. gibiansky. com/blog/machine-learning/baidu-allreduce.

[2] Kim S, Yu G I, Park H, et al. Parallax: Sparsity-aware data parallel training of deep neural networks [C]//*Proceedings of the Fourteenth EuroSys Conference*. Dresden, MO, USA, 2019: 1 − 15. DOI: 10. 1145/3302424. 3303957.

[3] Kurth T, Treichler S, Romero J, et al. Exascale deep learning for climate analytics [C]//*International Conference for High Performance Computing, Networking, Storage and Analysis*. Dallas, TX, USA, 2018: 649 − 660. DOI: 10. 1109/SC. 2018. 00054.

[4] Lü G F, Li M F, An H, et al. Distributed deep learning system for cancerous region detection on Sunway TaihuLight [J]. *CCF Transactions on High Performance Computing*, 2020, **2**(4): 348 − 361. DOI: 10. 1007/s42514-020-00046-5.

[5] Cui H, Radosavljevic V, Chou F C, et al. Multimodal trajectory predictions for autonomous driving using deep convolutional networks [C]//2019 *International Conference on Robotics and Automation*. Montreal, Canada, 2019: 2090-2096. DOI: 10. 1109/ICRA. 2019. 8793868.

[6] Liang J, Makoviychuk V, Handa A, et al. Gpu-accelerated robotic simulation for distributed reinforcement learning [C]//*Conference on Robot Learning*. Zurich, Switzerland, 2018: 270 − 282. DOI: abs/1810. 05762.

[7] Reisizadeh A, Prakash S, Pedarsani R, et al. Codedreduce: A fast and robust framework for gradient aggregation in distributed learning [J]. *IEEE/ACM Transactions on Networking*, 2022, **30** (1): 148 − 161. DOI: 10. 1109/TNET. 2021. 3109097.

[8] Jia X, Song S, He W, et al. Highly scalable deep learning training system with mixed-precision: Training ImageNet in four minutes [EB/OL]. (2018-07-30) [2022-09-05]. https://arxiv. org/abs/1807. 11205.

[9] Mikami H, Suganuma H, Tanaka Y, et al. Massively distributed SGD: ImageNet/ResNet-50 training in a flash [EB/OL]. (2019-03-05) [2022-09-05]. https://arxiv. org/abs/1811. 05233.

[10] Zhang A, Chen J, Hu R Q, et al. SeDS: Secure data sharing strategy for D2D communication in LTE-Advanced networks [J]. *IEEE Transactions on Vehicular Technology*, 2015, **65** (4): 2659 − 2672. DOI: 10. 1109/TVT. 2015. 2416002.

[11] Lopes A P G, Gondim P R L. Group authentication protocol based on aggregated signatures for D2D communication [J]. *Computer Networks*, 2020, **178**: 107192. DOI: 10. 1016/j. comnet. 2020. 107192.

[12] Choi K Y, Hwang J Y, Lee D H. Efficient ID-based group key agreement with bilinear maps [C]//*Proceedings of the 7th International Workshop on Public Key Cryptography*. Singapore, 2004: 130 − 144. DOI: 10. 1007/978-3-540-24632-9_10.

[13] Burmester M, Desmedt Y. A secure and efficient conference key distribution system [C]//*EUROCRYPT'*94. Perugia, Italy, 1994: 275 − 286. DOI: 10. 1007/BFb0053443.

[14] Kim Y, Perrig A, Tsudik G. Tree-based group key agreement [J]. *ACM Transactions on Information and System Security*, 2004, **7** (1): 60 − 96. DOI: 10. 1145/984334. 984337.

[15] Mao Y, Sun Y, Wu M, et al. JET: Dynamic join-exit-tree amortization and scheduling for contributory key management [J]. *IEEE/ACM Transactions on Networking*, 2006, **14** (5): 1128 − 1140. DOI: 10. 1109/TNET. 2006. 882851.

[16] Wang M, Yan Z. Privacy-preserving authentication and key agreement protocols for D2D group communications [J]. *IEEE Transactions on Industrial Informatics*, 2017, **14** (8): 3637 − 3647. DOI: 10. 1109/TII. 2017. 2778090.

[17] Scott M. Efficient implementation of cryptographic pairings [EB/OL]. (2016-02-08) [2022-12-05]. http://www. pairing-conference. org/2007/invited/Scott_slide. pdf.

Ring Allreduce 中的梯度参数完整性保护方案

方雨楠　蒋　睿

（东南大学网络空间安全学院，南京 210096）

摘要：为了解决 Ring Allreduce 架构中没有保护梯度参数完整性的问题，提出了一种面向 Ring Allreduce 架构的梯度参数完整性保护方案（RAA-DIP）. 使用基于身份的组密钥协商算法和 Boneh-Lynn-Shacham（BLS）签名方案保障 Ring Allreduce 中梯度参数的完整性. 结合身份认证和密钥协商算法，实现安全高效的动态工作节点管理. 基于 DBDH 问题实现安全的组密钥协商，使密钥生成中心（KGC）或网络攻击者无法计算工作节点的共享密钥，解决密钥托管问题，保障传输参数的完整性. 对 RAA-DIP 方案进行形式化证明，并将仿真结果与相关方案进行比较. 结果表明：RAA-DIP 方案可以保障 Ring Allreduce 中梯度参数传输过程的完整性，实现工作节点动态管理，解决密钥托管问题；与其他方案相比，RAA-DIP 方案可以同时满足安全和性能需求.

关键词：分布式机器学习；数据完整性；组密钥协商；Ring Allreduce 架构

中图分类号：TP309.2