Received: 21 September 2018 | Revised: 12 March 2019 | Accepted: 6 April 2019

DOI: 10.1002/ett.3633

RESEARCH ARTICLE

WILEY

A twofold group key agreement protocol for NoC-based MPSoCs

Gaurav Sharma[1] | Veronika Kuchta[2] | Rajeev Anand Sahu[1] | Soultana Ellinidou[1] | Suman Bala[1] | Olivier Markowitch[1] | Jean-Michel Dricot[1]

[1]Cybersecurity Research Center, Université Libre de Bruxelles, Brussels, Belgium

[2]Cyber Security & Systems, Monash University, Melbourne, Australia

Correspondence
Gaurav Sharma, Cybersecurity Research Center, Université Libre de Bruxelles, 1050 Brussels, Belgium.
Email: sharmagaurav@ieee.org

Funding information
Project ARC (Concerted Research Action) of Federation Wallonie-Bruxelles

Abstract

A symmetric group key agreement protocol enables the group members to derive a shared session key for secure communication among them, whereas an asymmetric one facilitates security to any communication from outside, without adding outsiders into the group. In order to combine both the functionalities, a hybrid key agreement protocol is needed, which can output a shared symmetric key for inside communication and an asymmetric key pair for any unrestricted sender. The application mentioned in this paper pushes the need of secure on-chip communication for intersecure and intrasecure zones simultaneously. In particular, we look forward for a solution to ensure communication security among multiple processing clusters actively running on an integrated circuit. The proposed protocol offers a lightweight symmetric encryption for intra-zone communication and a public key encryption for interzone communication taking most advanced security issues into account.

1 | INTRODUCTION

The emerging practical needs are constantly looking for a scalable solution to accommodate multiple processing units in an electronic device, processing simultaneously to achieve a common goal. The common bus standards for on-chip communication are advanced high-performance bus and advanced eXtensible interface. However, to make the design more scalable, network-on-chip is more appropriate option. To execute a complex task, more than one processing units (cluster of processors, processing clusters [PCs]) might be needed. These PCs will create a virtual zone to run the application securely. In literature, these virtual zones are referred as secure zones. Figure 1 depicts three secure zones. Indeed, the running programs need to exchange data and instruction sets quite frequently. Usually, this communication is in plaintext and any malicious program can intentionally extract and modify the data. To make this communication secure, there is a need of an encryption algorithm (symmetric or asymmetric). There may be some applications that do not need security at all. However, sensitive applications need to share their data and operation codes, only in an encrypted manner. Whenever PCs share data among themselves (within the same secure zone), symmetric encryption key can be a viable and cost effective solution. Furthermore, the usage of same symmetric key on all the zones will undermine the overall security standard. The compromise of this common symmetric key will expose the security of the whole system. Moreover, this solution is only restricted to inside communication of a secure zone.

In several situations, in order to timely execute a particular application, an outside IP may be borrowed from another cluster. The borrowed IP might be available at a distance from secure zone, and therefore, the computations performed

An extended abstract of this contribution has already been published[1] at Privacy, Security & Trust (PST) 2018.

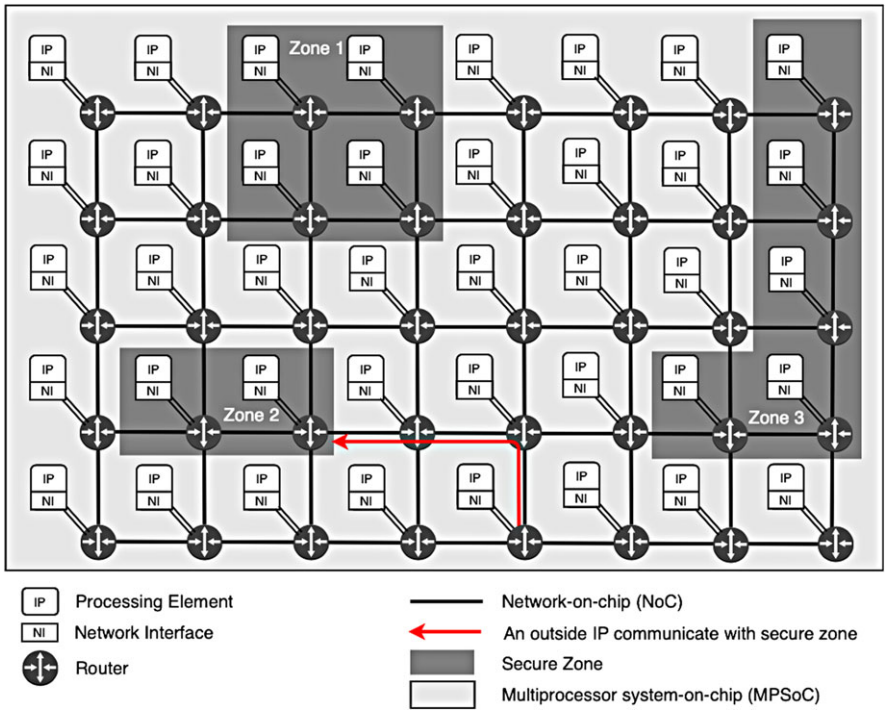

FIGURE 1 Three secure zones in an MPSoC platform

by this IP must be encrypted before forwarding to zone members. Following the above discussion, a suitable solution can be achieved by deriving different session keys for all the zones for secure intrazone communication and an asymmetric encryption key for interzone communication. In our paper, we propose a twofold group key agreement (GKA) protocol, which outputs a shared symmetric key as well as a group encryption/decryption key pair. In our case, the asymmetric GKA (ASGKA) will derive different decryption key for all the participants.

1.1 | The need of group key agreement

A group key agreement (GKA) protocol ensures establishment of a common session key among the group members, which remains unknown to outsiders. Practically, a GKA enables multiple remote users to communicate securely in an open environment. This group session key is further used for encryption and decryption of shared data. There are various GKA protocols which can provide a session key. However, this session key will be limited to group members only and no outsider unit can communicate to these group members. For any outside communication, which is very relevant in the real-world scenario, ASGKA protocol was first proposed by Wu et al.[2] In the subsequent improvements, the privilege of adding and removing group members was added. These protocols are known as dynamic ASGKA protocols.

The GKA protocol is applicable in various real-world communication networks such as ad hoc networks, wireless sensor networks, and body area networks, where devices are involved in sharing common secret data over an open channel. There are numerous real-life examples of GKA including distributed computations, video conferencing, multiuser games, etc. The key establishment protocols can be categorized into two sets: key transport protocols and key agreement protocols. In the former, the session key is derived by one of the powerful nodes, and then the key is transferred securely to all the members of the group. In the latter, a common session key is derived by all the members by interactive participation in an agreement protocol. Moreover, GKA protocols can be further categorized into balanced and imbalanced protocols. All the participants in balanced GKA share same computing burden, whereas in imbalanced protocols, a powerful node verifies all the received signatures. As established by Bellare and Rogaway (Crypto'93),[3] *authentication* is an essential security requirement for key exchange protocols, otherwise the man in the middle attack yields the protocol vulnerable to impersonation attacks.

The issuing and storage of digital certificates in tiny devices are complex tasks. In order to support multiple applications simultaneously, more PCs are required to be employed. However, as the multiprocessor system-on-chip (MPSoC) scales up, the public key storage of all the PCs on board is a challenging task itself. Motivated by Shamir's idea of identity-based

(ID-based) cryptosystem,[4] we deploy our scheme on the ID-based setting to avoid overhead of certificate management due to classical PKI setup. In ID-based system, the trusted private key generator (PKG) generates private keys for all the participating members and public keys are directly derivable from the user's identity.

1.2 | Related work

We extensively review the related literature and present state of the art in three major sets, namely, *symmetric GKA (SGKA)*, *ASGKA*, and secure communication in system-on-chip (SoC).

1.2.1 | Symmetric GKA (SGKA)

After the seminal work of Diffie and Hellman,[5] there have been extensive efforts to convert their two-party key exchange protocol to multiparty key exchange protocol.[6-8] Among the most notable works, Joux's one-round three-party key agreement protocol[9] is considered as a significant contribution for practical GKA protocol due to the functionality of pairing. Based on Joux's work,[9] Barua et al[10] have presented protocols of multiparty key agreement in two flavors: *unauthenticated* (based on ternary trees) and *authenticated* (from bilinear maps). Unfortunately, their protocols are secure against passive adversaries only. The first provable security model for authenticated key exchange (AKE) security was introduced by Bresson et al,[11-13] but their protocol accounts $O(n)$ rounds, which is very expensive. Furthermore, the model was improved in the work of Katz and Yung.[14] They proposed a scalable compiler, which transforms any unauthenticated GKA into the authenticated one. Later, Katz and Shin[15] modeled the *insider security* in GKA protocols. In 2009, Gorantla et al[16] proposed a security model, we call it the GBG model, which addresses the *forward secrecy* and *key compromise impersonation resilience* (KCIR) for GKA protocols to take into account AKE security and mutual authentication (MA) security. Their model was revisited and enhanced by Zhao et al[17] in 2011. They improved the GBG model to a stronger extended GBG model, we call it the EGBG model, where they addressed the ephemeral secret key leakage (ESL) attack. The EGBG model is the strongest model, as it takes into account both the leakage of secret key and the leakage of ephemeral key independently. Later, Tseng et al[18] argued that unforgeable adaptive chosen message attack (UF-ACMA) security is not sufficient and proposed a UF-ACM-ESL secure signature based on the work of Schnorr.[19]

In the *ID-based* setting, the first authenticated GKA (AGKA) protocol was formalized by Choi et al[20] in 2004, but their scheme was found vulnerable to insider colluding attack.[21] In 2007, Shim[22] claimed that scheme in the work of Choi et al[20] is vulnerable to another insider colluding attack and improved the protocol. Unfortunately, none of these AGKA protocols could achieve the perfect forward secrecy. Perfect forward secrecy allows the compromise of long-term secret keys of all participants maintaining all earlier shared secrets unrevealed. In 2011, Wu et al[23] presented a provably secure ID-authenticated group key exchange (AGKE) protocol from pairings, providing forward secrecy and security against the insider attacks. Later, Wu et al[24] presented their first revocable ID-based AGKE protocol, which is provably secure and can resist malicious participants as well. The main attraction of this protocol was efficient revocation of group members. The protocol takes three rounds and cannot identify malicious participants.

In a subsequent improvement, Wu and Tseng[25] proposed an ID-based AGKE protocol, which can passively detect malicious participants and also proved its security against insider attacks; although the protocol was later found insecure against an insider colluding attack by Wei et al.[26] Afterwards, a two-round revocable ID-AGKE protocol was presented by Wu et al,[27] which can identify malicious participants. The major limitation of existing literature is not to consider the ephemeral key leakage.[17] In 2015, Teng et al[28] presented first ID-based AGKA protocol secure in EGBG model. Their protocol satisfies MA security with KCIR, achieving full forward secrecy. This protocol includes extensive number of pairing operation ($2n^2 - 2n$), which is inefficient for practical implementations specially for low power devices.

1.2.2 | Asymmetric GKA (ASGKA)

The trivial solution for outsider communication in a group may include assigning of individual (public, private) key pairs and the sender needs to have all the public keys. In this case, the ciphertext size grows linearly. The notion of asymmetric GKA (ASGKA) was introduced by Wu et al[2] and they presented a generic protocol with its instantiation. The objective was to derive a group encryption key and a separate decryption keys for all the group members. This publicly accessible group encryption key enables any outsider sender to broadcast messages securely to the group members, and an individual decryption key is used to decrypt any ciphertext, encrypted under the group encryption key. In contrast to conventional GKA protocols, ASGKA can provide key confirmation without extra communication. Any participant can locally encrypt

a message under the group public key and the decryption can be performed by using individual decryption keys. This first construction[2] was secure only against passive attackers. Moreover, the protocol is vulnerable to collusion attack.[29] Any subset of group members may collude to derive a new decryption key, which is different from those of colluders. Some other improvements based on PKI were also presented in the works of Wu et al.[30,31]

In *ID-based* setting, first authenticated ASGKA protocol was presented by Zhang et al.[32] The preliminary version of this paper proposed ID-based authenticated ASGKA protocol, secure against indistinguishable chosen plaintext attack (IND-CPA). Extended version of this paper[33] provided detailed proof and achieved stronger security against indistinguishable chosen ciphertext attack (IND-CCA). These authenticated ASGKA protocols do not achieve perfect forward secrecy. Perfect forward secrecy allows the compromise of long-term secret keys of all participating members and, still, no earlier shared secrets are reveled. Moreover, it is noted that ID-based cryptosystem suffers from key escrow problem. The Key Generation Center (KGC) has all the private keys and hence can always read the secrets. Zhao et al[34] presented a dynamic ASGKA protocol for ad hoc networks. They offered a common encryption and decryption key to all the group members. Furthermore, Zhang et al[35] proposed an authenticated escrow-free ASGKA protocol. The additional key escrow feature disables the KGC to corrupt the ASGKA protocol. The perfect forward secrecy is directly implied by escrow freeness. Most of the ASGKA protocols except the work of Zhao et al[34] support static group. Later, Li et al[36] presented a dynamic one-round authenticated ASGKA protocol, which is secure against active adversary. The security attributes claimed by Li et al[36] are known key security, unknown key share, key compromise impersonation (KCI), perfect forward secrecy, key control security, and backward secrecy. This protocol provides an additional feature, which supports joining and leaving together.

1.2.3 | Secure communication in SoC

The best strategy to optimize the performance of MPSoC architecture is to split the application in appropriate number of tasks and spread it over multiple IP cores. Throughout this paper, we refer IP core and PC interchangeably. In order to execute a sensitive application, one of the common approach is to embed firewalls on the boundary IPs of the zone[37,38] and this zone is referred as secure zone. The major limitation of this approach is that, it can protect physically close IPs (related work follows the term "continuous security zones"). However, it is a weak assumption that the task scheduler always allocates a continuous security zone to a secure application.

Another grouping of IP cores could be through scattering of application on distant IPs and this zone is called as disrupted security zone. A secure application forces them to exchange sensitive data, and therefore a temporary session key is required to encrypt the communication. The initial solutions achieve pairwise key among IP cores.[39,40] Sepúlveda et al[37] addressed this issue and presented elastic security zones for 3D-MPSoC with groupwise shared secret. A hybrid group key solution was also presented in the work of Sepúlveda et al,[38] which employs asymmetric (to derive shared key) and symmetric (to encrypt exchanged communication) together. The major limitation of these approaches is lack of scalability and efficiency. Some other GKA solutions based on Diffie-Hellman–based key exchange[38,41-43] were introduced later. Another improved variant of the same approach presented three hierarchical groupwise key agreement protocols.[44] This work suggests to store a permanent secret during fabrication and a global manager becomes the single point compromise as it stores all the credentials to securely communicate with IPs.

1.3 | Motivation and our contribution

In order to support a complex application or running multiple applications in parallel, more processing units are required, and therefore multiple clusters need to be employed. To restrict the communication among the PCs in a cluster, a GKA protocol is implemented. The creation and destruction of these clusters are expected to be quite frequent. All the existing GKA protocols are either completely symmetric or asymmetric but not both. The derivation of only symmetric or asymmetric key does not assist in the discussed application. If only asymmetric key is established, the cost of encryption/decryption will be extremely high. In addition, only the symmetric key will not allow any outside communication without adding them to the group. Even if the symmetric key is transported by one of the participating members after the establishment of asymmetric key agreement, the key will not be contributive at all.

From the application perspective, the missing link in MPSoC security is zone to zone communication. All the existing GKA protocols enable secure communication within the zone only. To protect the interzone communication, an asymmetric protocol needs to be taken into account. This will greatly enhance the complexity. Our proposal presents a hybrid approach to fulfill the need of two separate protocols into one. This hybrid approach can especially be time saver for setting up the parameters for two different protocols. Moreover, the key generation and secure delivery of private key are also

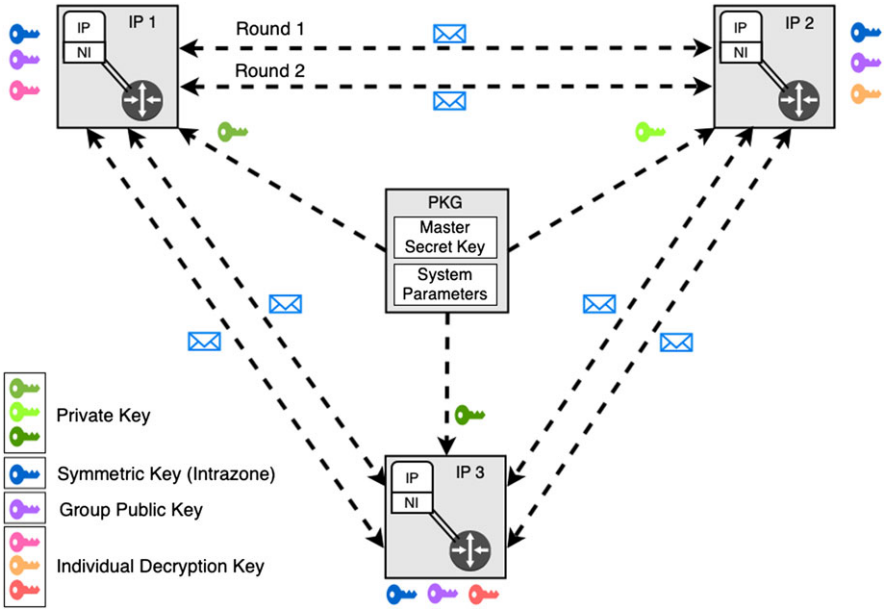

FIGURE 2 Twofold group key agreement protocol

performed once in our protocol. It is always better to provide security in a composable manner. Figure 2 below depicts the twofold GKA (TGKA) protocol.

In this paper, we present the first GKA protocol, which contributes a dual keying solution for secure communication among zone participants as well outside contributors. Furthermore, in the existing literature, there is no security model that enables the leakage of ephemeral key for ASGKA protocols. The major contributions in this paper are listed as follows:

- The enhanced security standard for existing ASGKA protocols considering the leakage of ephemeral key.
- A hybrid solution providing a shared *symmetric key* to all the participating members and an individual group decryption key against a common group *public key*, enabling secure communication from an unrestricted sender.

1.4 | Road map

The rest of this paper is organized as follows. In Section 2, we introduce related definitions and hardness assumption for the security. In Section 3, we define the TGKA protocol and its security model. The proposed GKA protocol is described in Section 4. The performance and security analyses of the proposed protocol are presented in Sections 5 and 6, followed by the conclusion in Section 7.

2 | PRELIMINARIES

In this section, we introduce computational problems and hardness assumptions. If X is a set, then $y \xleftarrow{\$} X$ denotes the operation of choosing an element y of X according to the uniform random distribution on X.

2.1 | Definitions and assumptions

Definition 1 (Bilinear map)**.**
Let there be two cyclic groups G, an additive cyclic group, and G_T, a multiplicative cyclic group, of prime order q. Let P be a generator of G. We define a map $e{:}G \times G \rightarrow G_T$ to be a *cryptographic bilinear map* if it fulfills the following properties.

Bilinearity: $e(xP,yP) = e(P,P)^{xy}$, for all $x, y \in \mathbb{Z}_q^*$. In other words, $e(R+S,T) = e(R,T)e(S,T)$ and $e(R,S+T) = e(R,S)e(R,T)$, for all $R,S,T \in G$.

Nondegeneracy: There exists $R,S \in G$ such that $e(R,S) \neq 1$. In other words, $e(P,P) \neq 1$, ie, there is a generator $e(P,P)$ of G_T, which is not unity.
Computability: There exists algorithm to efficiently compute $e(R,S) \in G_T$, for all $R,S \in G$.

Definition 2 (Computational Diffie-Hellman problem [CDHP])**.**
Let G be an additive cyclic group (we consider an elliptic curve group) of order q with generator P. Then, for given $P,aP,bP \in G$, the CDHP is used to efficiently compute $abP \in G$ without the knowledge of $a, b \in \mathbb{Z}_q^*$. (Note that gaining the value of $a \in \mathbb{Z}_q^*$, given $P,aP \in G$ is solving the elliptic curve discrete logarithm problem).

Definition 3 (Computational Diffie-Hellman assumption)**.**
This assumption says that, for a security parameter λ, the probability, of solving the above defined CDHP in group G, is negligible in λ. In other words, the (t,ϵ)-CDH assumption holds in group G if there is no algorithm that takes at most t running time and can solve CDHP with at least a nonnegligible probability ϵ.

3 | TGKA PROTOCOL

Suppose n participant PCs $U_1,U_2,\ldots,U_n$ attempt to create a cluster by running a protocol (π). Each PC is provided with a (public, private) key pair. In our protocol, we refer by *session* a running instance. Each participant is allowed to run multiple sessions concurrently. An ith instance of the protocol is represented as Π_U^i, where U is the corresponding user or participant. We define two identities, the session identity sid_U^i, which is the session dependent information computed by user U at its ith instance using the shared information in that session, and the partner identity pid_U^i, which is a set of identities of the participants who are involved in generation of the session key with Π_U^i. We say an instance Π_U^i *accepts* when it computes a valid session key sk. We say instances Π_U^i and $\Pi_{U'}^j$(for $\Pi_U^i \neq \Pi_{U'}^j$) are partnered iff (i) they have both accepted (ii) $sid_U^i = sid_{U'}^j$ and (iii) $pid_U^i = pid_{U'}^j$. We further define the term *freshness*.

Definition 4 (Freshness)**.**
An instance Π_U^i is referred to be fresh if it satisfies the following conditions:

1. If the instance Π_U^i is accepted, neither U_i nor any of its partnered instances can query Reveal key oracle.
2. No participant is allowed to query Corrupt and Reveal Ephemeral Key simultaneously.
3. In a partnered instance between U_i and U_j, if an adversary $\mathcal{A}$ corrupts U_j, any message sent from U_j to U_i must actually come from U_j.
4. The instance Π_U^i or any of its partner has not been asked a Decryption Key Reveal query after their acceptance.

We precisely present here two-round interactive authenticated twofold key agreement protocol divided into **Setup**, **KeyGen**, **Key Agreement**, and **Key Computation** phases. The **Key Computation** phase consists of subphases to derive shared secret key as well as asymmetric key pair. We assume the existence of a PKG who generates long-term private keys for the users. The PKG can be a tamper-proof secure IP core and the private keys are delivered to users via a secure channel. The users interact among them using their private keys to share session dependent information *sid*, which leads to compute the session key.

3.1 | Security model of TGKA protocol

Similar to the existing security models, we model the secrecy in terms of indistinguishability of encrypted ciphertext (for a chosen message, against derived group public key) from a random string in the ciphertext space. Here, we extend the existing asymmetric security model with the privilege of revealing the ephemeral key. To realize this standard security frame, the following game between the challenger C and the adversary $\mathcal{A}$ is defined as follows:

Setup: On input security parameter 1^λ, the challenger C runs **KeyGen**(1^λ) to generate the public parameter *Params* and the system key pair (pk,msk) and gives the adversary $\mathcal{A}$ the public key pk. msk is the master secret of the system and kept secret with challenger C.

Queries: $\mathcal{A}$ can adaptively make the following queries:

- Execute(Π_U^i): This query models the honest execution of the protocol π. In other words, this query returns all the transcripts/messages exchanged among participants. Moreover, the participants are selected by the adversary itself. This models passive attacks.
- Send(Π_U^i, m): This query sends a message m to instance Π_U^i and outputs the reply generated by instance Π_U^i. For the initiation of the protocol, the message may be (*sid*,*pid*), where *sid* is the session identity and *pid* is the partner identity. If the message is not of intended format, this query returns null. Moreover, *Execute* query can be substituted by making *Send* query repeatedly.
- Corrupt(U_i): This query models the reveal of long-term secret key. The participant is honest iff adversary $\mathcal{A}$ has not made any *Corrupt* query. Recall that the reveal of long-term secret key will model forward secrecy.
- Reveal Key (Π_U^i): When the oracle is accepted, this query outputs the group session key.
- Ephemeral Key Reveal(Π_U^i): This query models the reveal of ephemeral key of participant U_i for instance Π_U^i.
- Encryption Key Reveal (Π_U^i): When this oracle is accepted, it outputs group encryption key.
- Decryption Key Reveal (Π_U^i): To model known key security, this oracle allows the group decryption key as output.
- Test(Π_U^i): This query models secrecy and can be made only once during the execution of protocol π. The adversary $\mathcal{A}$ selects two messages (m_0,m_1) and a fresh instance. During the Test query, the challenger C randomly selects a bit $b \overset{\$}{\leftarrow} \{0,1\}$ and returns a ciphertext C_b corresponding to the selected message m_b.

Guess: $\mathcal{A}$ outputs its guess b' for b.

The adversary succeeds in breaking the security if $b^{'} = b$. We denote this event by $Succ_{\mathcal{A}}$ and define $\mathcal{A}$'s advantage as $Adv_{\mathcal{A}}(1^{\lambda}) \overset{\text{def}}{=} |2Pr[Succ_{\mathcal{A}}] - 1|$.

Definition 5 (AKE security)**.**
Let $\mathcal{A}_{ake}$ be an adversary against AKE security. It is allowed to make queries to the Execute, Send, RevealKey, Ephemeral Key Reveal, and Corrupt oracles. It is allowed to make a single Test query to the instance Π_U^i at the end of the phase and given the challenge session key $sk_{ch,b}$ (depending on bit b). Finally, $\mathcal{A}_{ake}$ outputs a bit $b^{'}$ and wins the game if (1) $b = b^{'}$ and (2) the instance Π_U^i is fresh till the end of the game. The advantage of $\mathcal{A}_{ake}$ is $Adv_{\mathcal{A}_{ake}} = |2Pr[Succ_{\mathcal{A}_{ake}}] - 1|$. The protocol is called AKE secure if the adversary's advantage $Adv_{\mathcal{A}_{ake}}$ is negligible.

In the following, we recall the MA-security considering both types of adversaries, outsiders and insiders. An outsider adversary may compromise the long-term private key of all parties except one. An outsider adversary is successful in KCI attack if it can impersonate an uncorrupted instance (in our case, the ephemeral key) of an uncorrupted party to an uncorrupted instance of any of the corrupted parties. The adversary's goal is to break the confidentiality of the session private key and to break the MA-security. An adversary is called insider adversary if it succeeds in corrupting a party and participating in a protocol session representing the corrupted party. An insider adversary is successful in breaking KCI security if it succeeds to impersonate an uncorrupted instance of an uncorrupted party A to another uncorrupted instance of another party B. The only goal of an insider adversary is to break the MA-security.

Definition 6 (MA-security [with outsider KCIR])**.**
Let $\mathcal{A}_{ma,out}$ be an outsider adversary against MA-security. Let pid_U^i be a set of identities of participant in the group with whom Π_U^i wishes to establish a session key and sid_U^i denotes a session id of an instance Π_U^i. $\mathcal{A}_{ma,out}$ is allowed to make queries to the Execute, Send, RevealKey, EphemeralKey Reveal, and Corrupt oracles. $\mathcal{A}_{ma,out}$ breaks the MA-security with outsider KCIR notion if at some point there is an uncorrupted instance Π_U^i with the key sk_U^i and another party U' that is uncorrupted when Π_U^i accepts such that there are no other insiders in pid_U^i and the following conditions hold:

- there is no instance $\Pi_{U'}^{i'}$ with $(pid_{U'}^{i'}, sid_{U'}^{i'}) = (pid_U^i, sid_U^i)$ or,
- there is an instance $\Pi_{U'}^{i'}$ with $(pid_{U'}^{i'}, sid_{U'}^{i'}) = (pid_U^i, sid_U^i)$, which has accepted with $sk_{U'}^{i'} \neq sk_U^i$.

Definition 7 (MA-security [with insider KCIR])**.**
Let $\mathcal{A}_{ma,in}$ be an insider adversary against MA-security. It is allowed to query Execute, Send, RevealKey, Emphemeral Key Reveal, and Corrupt oracles. It breaks the MA-security with insider KCIR if at some point there is an uncorrupted instance Π_U^i that has accepted with the secret key sk_U^i and another party U' that is uncorrupted when Π_U^i accepts and

- there is no instance $\Pi_{U'}^{i'}$ with $(pid_{U'}^{i'}, sid_{U'}^{i'}) = (pid_U^i, sid_U^i)$ or,

- there is an instance $\Pi^{i'}_{U'}$ with $(pid^{i'}_{U'}, sid^{i'}_{U'}) = (pid^{i}_{U}, sid^{i}_{U})$ that has accepted with $sk^{i'}_{U'} \neq sk^{i}_{U}$.

Furthermore, consider a public key encryption scheme $E = (\textbf{Setup}, \textbf{KeyGen}, \textbf{Encryption}, \textbf{Decryption})$, where $param \leftarrow \textbf{Setup}(1^{\lambda})$, $(pk,sk) \leftarrow \textbf{KeyGen}(param)$, $c \leftarrow \textbf{Encryption}(param,pk,m)$, and $m \leftarrow \textbf{Decryption}(sk,c)$.

Definition 8 (IND-CCA security)**.**
Let $\mathcal{A}_{ind}$ be a PPT adversary. For a bit $b \in \{0,1\}$, the IND-CCA security experiment $Exp^{IND\text{-}CCA}_{\mathcal{A}_{ind},E}$ is as follows: (1) $param \leftarrow \textbf{Setup}(1^{\lambda})$; (2) $pk' \leftarrow \textbf{KeyGen}(param)$; (3) $\mathcal{A}_{ind}$ is given access to the left-right oracle $L - R_{pk'}(\cdot,\cdot)$ on input two messages of equal length, m_0, m_1. The oracle returns the encryption of m_b under the key pk'; (4) $\mathcal{A}_{ind}$ has access to the decryption oracle $\mathcal{O}\textbf{Decryption}$ on a ciphertext by its choice except the ciphertext output by $L - R_{pk'}$ oracle; (5) Finally, $\mathcal{A}_{ind}$ outputs a bit b'.

We say, an encryption scheme is secure if the following advantage of the adversary in negligible:

$$Adv^{IND\text{-}CCA}_{\mathcal{A}_{ind},E} = \left| Pr\left[Exp^{IND\text{-}CCA\text{-}0}_{\mathcal{A}_{ind},E} = 1\right] - Pr\left[Exp^{IND\text{-}CCA\text{-}1}_{\mathcal{A}_{ind},E} = 1\right]\right|.$$

The IND-CPA security is defined in a similar way as IND-CCA, except for the access to decryption oracle, which is not given to an IND-CPA adversary.

Our construction also involves a digital signature scheme, which is required to be Unforgeable under Chosen Message Attacks (UNF-CMA) secure. A digital scheme consists of three algorithms, (**KeyGen**, **Sign**, **SVrfy**), where $(pk,sk) \leftarrow \textbf{KeyGen}(1^{\lambda})$, $\sigma \leftarrow \textbf{Sign}(sk,m)$, $1/0 \leftarrow \textbf{SVrfy}(pk,\sigma,m)$. This security notion is defined as follows.

Definition 9 (UNF-CMA security)**.**
Let $\mathcal{A}_{unf}$ be a PPT adversary. The security experiment $Exp^{UNF\text{-}CMA}_{\mathcal{A}_{unf}}$ is defined as follows. (1) The challenger samples $(pk,sk) \leftarrow \textbf{KeyGen}(1^{\lambda})$. It initializes a set S of queried messages. (2) $\mathcal{A}_{unf}$ has access to the signing oracle $\mathcal{O}\textbf{Sign}$ on a message m of her choice. (3) $\mathcal{A}_{unf}$ picks a message m^* and outputs a forgery $\sigma^*(m^*,pk)$. (4) $\mathcal{A}_{unf}$ wins if $\textbf{SVrfy}(\sigma^*,m^*,pk) = 1$ and $m^* \notin S$, ie, that message was not queried to the signing oracle before. We say, a signature scheme is unforgeable if the following advantage of $\mathcal{A}_{unf}$ is negligible in security parameter.

$$Adv^{UNF\text{-}CMA}_{\mathcal{A}_{unf}} = \left| Pr\left[Exp^{UNF\text{-}CMA}_{\mathcal{A}_{unf}} = 1\right]\right| \leq \epsilon(\lambda).$$

4 | IDENTITY-BASED TWOFOLD GROUP KEY AGREEMENT PROTOCOL

The presented key agreement protocol is suitable to run by PCs in order to derive a shared session key, to securely communicate inside the zones, whereas a common group encryption key and an individual decryption key for the communication are suitable with other zones. We assume the presence of a PKG, or the long-term secrets have already been stored in a secure memory. The PCs have been referred as users in our scheme. A TGKA protocol consists of the following algorithms.

Setup(1^{λ}): On input security parameter 1^{λ}, the PKG generates the system parameters *Params* in the following steps:

- Chooses an elliptic curve group G of prime order q. Let P be a generator of group G.
- Let $e: G \times G \rightarrow G_T$ be an admissible bilinear map, where G_T is a cyclic multiplicative group of order q.
- Computes system's public key as $P_{pub} = sP$ by choosing a master secret $s \xleftarrow{\$} \mathbb{Z}^*_q$.
- Chooses cryptographic hash functions $H_1 : \{0,1\}^* \times G \rightarrow \mathbb{Z}^*_q$, $H_2 : \{0,1\}^* \times G \times G \rightarrow \mathbb{Z}^*_q$, $H_3 : \{0,1\}^* \rightarrow \mathbb{Z}^*_q$, $H_4 : G_T \rightarrow \{0,1\}^c$, and $H: \{0,1\}^* \rightarrow \{0,1\}^k$.
- Finally, publishes the system parameters $Params = \{G,q,H_1,H_2,H_3,H_4,H,e,P_{pub}\}$ and keeps the master key secret.

KeyGen(s): The PKG performs the following for all the group members:

- Chooses $r_i \xleftarrow{\$} \mathbb{Z}^*_q$ and computes $R_i = r_i P$.
- Computes the private key for user U_i as $x_i = r_i + sH_1(ID_i,R_i)$.
- The user U_i can verify the private key as $x_i P = R_i + H_1(ID_i,R_i)P_{pub}$.

Key Agreement(x_i,pid): This protocol runs in the following two rounds.
Round 1: Each user $U_i (1 \leq i \leq n)$ does the following:

- Chooses $b_i \xleftarrow{\$} \mathbb{Z}_q^*$ and computes $l_i = H_3(b_i, x_i)$ and $L_i = l_i P$.
- Chooses a random string $k_i \in \{0,1\}^\lambda$ of length λ. Each user, except U_n, computes $H(k_i)$. The user U_n masks the randomness as $\tilde{k}_n = H(k_n, x_n)$, where x_n is long-term secret of U_n. Now, he computes $H(\tilde{k}_n)$.
- Broadcasts the tuple $< L_i, H(k_i), H(\tilde{k}_n), R_i >$.

Round 2: On receiving the message $< L_j, H(k_j), H(\tilde{k}_n), R_j >$, each user U_i performs the following:

- Computes $U_{ij} = l_i L_j$.
- Each user, except U_n, computes $K_{ij} = H(U_{ij}) \oplus k_i$. The user U_n computes $mask = H(U_{ij}) \oplus \tilde{k}_n$.
- Each participant computes $L = L_1 \| L_2 \| \ldots \| L_n$.
- Each participant computes $M_i = e(l_i x_i P_{pub}, P)$ and $\delta_{i,j} = l_i x_i P_{pub} + l_i R_j$ for $1 \le j \le n$.
- Choose another random number $t_i \in \mathbb{Z}_q^*$ and compute $T_i = t_i l_i P$. Moreover, compute the signature on $<L, M_i, T_i>$ as $\sigma_i = t_i l_i + x_i H_2(ID_i, M_i, L, T_i, pid)$.
- Broadcast $<K_{ij}, mask, \delta_{i,j} (1 \le j \le n, j \ne i), \sigma_i, T_i, M_i>$.

Key Computation: The key computation derives two different keys, one for symmetric encryption among group participants, and other for any unrestricted sender intended to share something among group participants without joining the group.

Shared Secret Key. Upon receiving $<K_{ji}, mask, \delta_{j,i}, \sigma_j, T_j, M_j>$, each user verifies the received signatures as

$$\sigma_i P = T_i + \left(R_i + H_1(ID_i, R_i) P_{pub}\right) H_2(ID_i, M_i, L, T_i, pid).$$

Each user U_i then computes $\tilde{k}_j = H(U_{ji}) \oplus K_{ji}$ and $\tilde{k}_n = mask \oplus H(U_{ij})$. Here, $U_{ij} = l_i L_j = l_i l_j P = l_j l_i P = l_j L_i = U_{ji}$.

Each user U_i checks correctness of k_i as $H(k_j) = H(\tilde{k}_j)$ for $(1 \le j \le n, j \ne i)$ and computes session identity $sid = H(k_1) \| H(k_2) \| \ldots \| H(\tilde{k}_n)$.

Finally, the shared session key is computed as $sk = H(k_1 \| k_2 \| \ldots \| \tilde{k}_n \| sid \| pid)$.

Asymmetric Key Pair. The common group encryption key (accessible to anyone) can be computed as $K_1 = \Sigma_{i=1}^n L_i$, $K_2 = \Pi_{i=1}^n M_i$.

The encryption key is $EK = (K_1, K_2)$. The decryption key, individual for each participant, can be computed as $\delta_i = \delta_{i,i} + \Sigma_{j=1}^{n, j \ne i} \delta_{j,i}$.

Encryption(***Params***, *EK*, *message*): Any unrestricted sender can encrypt a message m of length c by randomly selecting $\eta \in \mathbb{Z}_q^*$ and computing the ciphertext as $c_1 = \eta P$, $c_2 = \eta K_1$ and $c_3 = m \oplus H_3(K_2^\eta)$.

Decryption(***Params***, δ_i, *ciphertext*): Any group participant possessing the group decryption key δ_i can decrypt the ciphertext (c_1, c_2, c_3) as $m = c_3 \oplus H_4(K_2^\eta) \Leftrightarrow m = c_3 \oplus H_4(e(\delta_i, c_1).e(R_i^{-1}, c_2))$.

Correctness:

$$\begin{aligned} e(\delta_i, c_1).e\left(R_i^{-1}, c_2\right) &= e(\delta_i, \eta P).e\left(R_i^{-1}, \eta K_1\right) \\ &= e(\delta_i, P)^\eta.e(R_i, K_1)^{-\eta} \\ &= e\left(\Sigma_{i=1}^n (l_i x_i P_{pub} + l_i R_i), P\right)^\eta.e\left(R_i, \Sigma_{i=1}^n l_i P\right)^{-\eta} \\ &= e\left(\Sigma_{i=1}^n (l_i x_i P_{pub} + l_i R_i), P\right)^\eta \\ &= \Pi_{i=1}^n M_i^\eta. \end{aligned}$$

The message is $m = c_3 \oplus H_4(K_2^\eta)$.

Remark 1. The resulting encryption scheme is indistinguishable under chosen identity and plaintext attack (IND-ID-CPA).[45] A generic conversion by Fujisaki and Okamoto[46] can be used to transform it into IND-ID-CCA secure encryption scheme.

For the verification of all the received messages in one step, batch verification can be used as follows.

Batch verification: All the participants can verify the received signatures σ_i in one step as follows (details in the following section):

$$\left(\sum_{i=1}^n v_i \sigma_i\right) P = \sum_{i=1}^n v_i T_i + \left(\sum_{i=1}^n (v_i x_i h_{i2})\right) P.$$

4.1 | Batch verification of signature

An independently existential UF-ACMA secure signature scheme does not guarantee its security in batch signature. Recall that the signature algorithms proposed by Schnorr[19] and Hess[47] are not secure when used as batch signature. Recently, in a noted contribution by Horng et al,[48] an efficient method was introduced to derive batch signature.

Adopting the concept of small exponent test,[48] each member chooses a random vector $v = (v_1, v_2, \dots, v_n)$, where v_i ranges between 1 and 2^t, each v_i is random, to make sure the property of nonrepudiation. To avoid any computational overhead, t is a very small value with error probability at most 2^{-t}. Any forged signature can be easily detected with a glitch of probability 2^{-t}.

Correctness:

$$\left(\sum_{i=1}^{n} v_i \sigma_i\right) P = \left(\sum_{i=1}^{n} v_i (l_i + x_i h_{i2})\right) P$$
$$= \sum_{i=1}^{n} v_i \left(T_i + (R_i + h_{i1} P_{pub}) h_{i2}\right),$$

where $h_{i1} = H_1(ID_i, R_i)$ and $h_{i2} = H_2(ID_i, M_i, L, T_i, pid)$.

4.2 | Realization of the proposed protocol

To realize the proposed TGKA protocol in MPSoC architecture, all the PCs (in any zone) will run the GKA protocol and derive a shared session key sk. This shared session key is used to encrypt the communication inside the zone. For real-time implementation, any lightweight symmetric encryption such as AES or PRESENT cipher can be used. Moreover, the proposed GKA protocol is an AGKA, therefore malicious PC can be identified and discarded. Furthermore, all the secure zones derive a group encryption key $EK = (K_1, K_2)$ and a unique decryption key δ_i for each PC. The group encryption key can be derived by any outside entity to send encrypted message to secure zone. This feature enables secure communication from any PC on the integrated circuit (IC) to any other PC.

The three varieties of solutions discussed in Section 1.2.3 are vulnerable to various security threats. The first solution comprising pairwise session keys can never be efficiently scalable on a large MPSoC platform. Another solution where a key pool is distributed to all the IPs at design time will have the storage overhead proportional to the MPSoC size. Otherwise, it would be difficult to find a common key among IPs. The solution based on Diffie-Hellman key exchange is vulnerable to man-in-the-middle attack. To overcome these drawbacks, we present an AGKA protocol. In contrast to the existing GKA solutions, our solution provides authentication of the IP cores as well as secure communication inside and outside security zones. Moreover, the IPs involved in the security zones are still accessible by other IPs on the chip.

5 | PERFORMANCE ANALYSIS

The proposed construction is a fusion of symmetric as well as ASGKA protocols. Therefore, the performance of our protocol should be compared with a combination of symmetric and ASGKA protocols (SGKA+ASGKA) as if they were implemented separately. Since there is no such existing hybrid GKA protocol, therefore we will take two best ID-based GKA protocols (one SGKA and other ASGKA) into consideration. In contrast to run two separate protocols, our hybrid approach can especially be time saver for parameters setup, key generation, and secure delivery of private keys.

Although, there are one-round authenticated ASGKA protocols available, whereas designing one-round authenticated SGKA protocol is still an open problem. Therefore, our twofold protocol can save at least one round because any combination of SGKA and AGKA will take minimum three rounds. Moreover, when we compare the efficiency of our proposed TGKA protocol with any combination of recent ID-based SGKA+AGKA protocols, our protocol is far more efficient. Table 1 presents the comparative computational analysis of both the variants of GKA protocols and the twofold protocol. The computation cost indicated in table covers the key agreement and key computation phase, determining the cost of each participant.

From Table 1, it is clear that Sharma et al.[49] GKA protocol is the best choice for symmetric, whereas the most suitable asymmetric GKA might depend upon the choice of signature algorithm. However, in any case, our twofold protocol seems the best efficient option to replace two separate protocols.

TABLE 1 Performance comparison of TGKA with SGKA+ASGKA protocol

Scheme	GKA	Rounds	Computation Cost
Zhang et al[32]	Asymmetric	1	$(n+1)E+3e+1S^*+nV$
Zhao et al[34]	Asymmetric	3	$4E+3S^*+2nV$
Li et al[36]	Asymmetric	1	$(2n+5)E+7e$
Teng et al[28]	Symmetric	2	$(n+9)E+(2n-2)e$
Wu et al[27]	Symmetric	2	$(n+9)E+(3n+2)e$
Sharma et al[49]	Symmetric	2	$(3n+2)E$
TGKA	Twofold	2	$(4n+3)E+1e$

E: Exponentiation in G (or scalar multiplication in a cyclic additive group).
e: Bilinear map.
S: Signature generation.
V: Signature verification.
*: In the ASGKA protocols,[32,34] it is recommended to use any ID-based signature.
Abbreviations: ASGKA, asymmetric group key agreement; SGKA, symmetric group key agreement; TGKA, twofold group key agreement.

6 | SECURITY ANALYSIS

In this section, we consider various aspects of security for the introduced protocol. We analyze the security of the proposed protocol by considering different security properties one-by-one, by the means of following theorems. As discussed in the recent literatures[11,13,15,16,50] that for a GKA protocol, the security of session key, ie, AKE security, and the MA are of prime concern. Furthermore, it is crucial for a secure protocol to achieve the KCI resilience in this connection.

In the theorem below, we present the detailed proof for the AKE security of our protocol (assuming existence of an insider adversary).

Theorem 1. *The proposed TGKA protocol is AKE secure with insider KCIR adversary following Definition 5 assuming existence of random oracle, unforgeability of the underlying signature scheme, the hardness of CDHP ,and IND-ID-CCA security of the underlying encryption scheme. The success probability of adversary $\mathcal{A}_{ake}$ is the following advantage*

$$\left(n^2 Adv_{\mathcal{A}_{cma}} + \frac{(q_e+3q_s+q_h)^2}{2^\lambda} + \frac{q_s^2}{2^\lambda} + nq_{H_3}Adv_{CDH} + \frac{(q_d+q_h)}{2^\lambda}\right)$$

considering n parties in the network, $Adv_{\mathcal{A}_{cma}}$ as the advantage of the adversary $\mathcal{A}_{cma}$ against the unforgeable signature scheme, λ the security parameter, q_e the maximum number of Execute queries, q_s the maximum number of Send queries, $q_h = q_H + q_{H_2} + q_{H_3}$, where q_{H_i} the maximum number of hash H_i (random oracle) queries, q_d the maximum number of Decryption queries that the adversary $\mathcal{A}_{ake}$ can ask.

Proof. For the proof of claimed security, we follow the security model formalized in Section 3.1 and the game hopping technique from the work of Dent[51] in the way as adopted in the work of Gorantla et al.[16] The technique considers a sequence of games between the challenger and the adversary. The advantage of the adversary in winning the game is then shown to be bounded in terms of its advantage in distinguishing between a real and random values. This essentially shows the probability of ability of the adversary, which we define to be the advantage of the adversary, in distinguishing an encrypted ciphertext from a random string in the ciphertext space, as discussed in Section 3.1. To achieve the required security, we prove the probability to be negligible.

Let $\mathcal{A}_{ake}$ be the adversary against the AKE security of our scheme. As discussed in Section 3.1, $\mathcal{A}_{ake}$ is allowed to make Execute, Send, RevealKey, Ephemeral Key Reveal, and Corrupt queries. We construct two algorithms $\mathcal{B}_{cdh}$ and $\mathcal{B}_{ind}$ simulating adversaries of *CDH* problem and IND-CCA security of the underlying encryption scheme. The AKE security ensures the session key security. In particular, we consider KCI resilience against an insider adversary as discussed in the work of Gorantla et al.[16]

For the purpose, let E_i be an event that $\mathcal{A}_{ake}$ wins the ith AKE security game. Furthermore, let ρ_i be the corresponding advantage (probability of success) of $\mathcal{A}_{ake}$ in wining the ith AKE security game. We set $\rho_i = |2Pr[E_i] - 1|$. If an event E', which occurs during $\mathcal{A}_{ake}$'s execution, is detectable by simulator, then E' is independent of E_i. We say that two

successive games Game-g_i and Game-g_{i+1} are identical unless event E' occurs and the probability $Pr[E_{i+1}|E'] = 1/2$. In this case, we have

$$Pr[E_{i+1}] = \frac{1}{2} + Pr[\neg E'] \left(Pr[E_i] - \frac{1}{2} \right)$$

and

$$\rho_{i+1} = Pr[\neg E']\rho_i. \qquad \square$$

The sequence of games are described as follows:

Game-g_0:

This game is the actual AKE security game as introduced in Definition 5. The advantage of the adversary $\mathcal{A}_{ake}$ is given by

$$Adv_{\mathcal{A}_{ake}} = |2Pr[E_0] - 1| = \rho_0.$$

Game-g_1:

The game is same as Game-g_0 except that the simulation fails if an event `Forge` occurs, ie, when

$$|Pr[E_1] - Pr[E_0]| \leq Pr[\texttt{Forge}],$$

and hence, by the above relation and the definition of ρ_i,

$$\begin{aligned} \rho_0 &= |2Pr[E_0] - 1| \\ &\leq |2Pr[E_0] - 2Pr[E_1]| + |2Pr[E_1] - 1| \\ &\leq 2Pr[\texttt{Forge}] + \rho_1, \end{aligned}$$

which means that $\mathcal{A}_{ake}$ issues a Send query with (m_i,σ_i), assuming that user U_i is not corrupted and m_i was not output in the previous instance of U_i. According to the AKE security for KCI attack definition, $\mathcal{A}_{ake}$ can corrupt maximum $n-1$ parties, being remained passive on behalf of those corrupted users. If the event `Forge` occurs, then it shows the successful output of fake signature on behalf of the noncorrupted party as follows: the public key is assigned to one party where the other $n-1$ parties are assumed to be normal according to the protocol. Since $n-1$ parties are corrupt, the secret keys of those $n-1$ parties are known. The only secret key that corresponds to the public key of the UNF-CMA game can be simulated by the signing oracle available from the underlying signature scheme. Hence, the probability that $\mathcal{A}_{cma}$ does not corrupt a party is $\geq 1/n$, also the probability that $\mathcal{A}_{cma}$ forge the signature on behalf of the party is $\geq 1/n$. Hence,

$$\begin{aligned} &Adv_{\mathcal{A}_{ake}} \geq \frac{1}{n^2} Pr[\texttt{Forge}] \\ &\Leftrightarrow Pr[\texttt{Forge}] \leq n^2 Adv_{\mathcal{A}_{cma}}. \end{aligned}$$

Game-g_2:

This game is the same as the previous except that the simulation fails if an event `Collision` appears. Similarly, as above, the least probability of `Collision` can be counted as

$$|Pr[E_2] - Pr[E_1]| \leq Pr[\texttt{Collision}],$$

and hence, by the above relation and the definition of ρ_i,

$$\begin{aligned} \rho_1 &= |2Pr[E_1] - 1| \\ &\leq |2Pr[E_1] - 2Pr[E_2]| + |2Pr[E_2] - 1| \\ &\leq 2Pr[\texttt{Collision}] + \rho_2, \end{aligned}$$

this is the case when at least one of the random oracles, considered to respond the Send query, produces a collision. As the value of 3 hash functions, namely, H, H_2, and H_3, would require to be picked as random oracle to respond each Send query, the maximum number of random oracle queries is $(q_e + 3q_s + q_h)$. Where q_e, q_s, and q_h are as defined above. Hence, the probability of `Collision` is at most $\frac{(q_e+3q_s+q_h)^2}{2^\lambda}$. Thus,

$$Pr[\texttt{Collision}] \leq \frac{(q_e + 3q_s + q_h)^2}{2^\lambda}.$$

Game-g_3:

The game is the same as the previous one, except that the simulation fails when an event `Repeat` occurs. In this case,

$$|Pr[E_3] - Pr[E_2]| \leq Pr[\texttt{Repeat}],$$

and hence

$$\begin{aligned}\rho_2 &= |2Pr[E_2] - 1| \\ &\leq |2Pr[E_2] - 2Pr[E_3]| + |2Pr[E_3] - 1| \\ &\leq 2Pr[\texttt{Repeat}] + \rho_3.\end{aligned}$$

A `Repeat` event happens when an instance of user U_i chooses a nonce κ_i that was also used by another instance of the user U_i. The maximum of instances that can choose a nonce κ_i is $q_e + q_s$, therefore

$$Pr[\texttt{Repeat}] \leq \frac{(q_s + q_e)^2}{2^\lambda}.$$

Game-g_4:

This game differs from the previous game by $\mathcal{A}_{ake}$'s randomly chosen value ν, as a guess of the session in which the adversary would have requested the Test query, from the set of values, which is bounded by Execute and Send queries, ie, $\nu \in \{1, \ldots, q_s + q_e\}$.

However, the simulation aborts if the adversary $\mathcal{A}_{ake}$ chooses a session that is different than ν to issue a Test query. Otherwise, the event that the adversary picks the right session for being tested happens with the probability of $1/(q_e + q_s)$. It follows immediately that the simulation aborts with probability $1 - \frac{1}{(q_e+q_s)}$. We have

$$\rho_{i+1} = Pr[\neg E']\rho_i,$$

hence

$$\rho_4 = \frac{1}{(q_e + q_s)}\rho_3 \Longleftrightarrow \rho_3 = (q_e + q_s)\rho_4.$$

Game-g_5:

This game differs from previous game depending upon the answers to the Send queries during the Test session.

We assume that, in round 1, the values L_i are randomly chosen from group G. All other calculations are the same as in the previous game. Since in the previous game, the values L_i were computed as $L_i = l_i P$, where l_i were outputs from a random oracle for H_3 queries, as $l_i = H_3(b_i, x_i)$.

Hence, an adversary $\mathcal{A}_{ake}$ can distinguish between Game-g_4 and Game-g_5 only if it queries $x_{i+1}L_i (= x_i L_{i+1})$, for at least one value of i, to the random oracle. Let `Dis` be an event where the adversary succeeds to distinguish the values of a hash functions. Then,

$$|Pr[E_5] - Pr[E_4]| \leq Pr[\texttt{Dis}]$$

and

$$\begin{aligned}\rho_4 &= |2Pr[E_4] - 1| \\ &\leq |2Pr[E_4] - 2Pr[E_5]| + |2Pr[E_5] - 1| \\ &\leq 2Pr[\texttt{Dis}] + \rho_5.\end{aligned}$$

When $\mathcal{A}_{ake}$ can distinguish between the random and real values, then we can also solve the CDH problem as follows. Let U_i be a random party involved in the Test session, then we set $R = aP$ and $S = bP$ and assign $L_i = R$ and $L_{i+1} = S$. The CDH instance is $(P, R = aP, S = bP)$. Let C be a randomly chosen value, and if `Dis` occurs, then the probability that C is a solution of CDH problem is at least $1/(q_{H_3})$.

Furthermore, the minimum probability of C being correct solution to the instance (P, aP, bP) is $\frac{1}{n}$. It follows that $Adv_{CDH} \geq \frac{1}{nq_{H_3}} Pr[\texttt{Dis}]$. Hence,

$$Pr[\texttt{Dis}] \leq nq_{H_3} Adv_{CDH}.$$

Game-g_6:

This game differs from previous game by the simulation of encryption secret key. Here, the algorithm $\mathcal{B}$ is simulated by $\mathcal{A}_{ake}$. Whenever $\mathcal{A}_{ake}$ issues queries to the decryption key reveal oracle, $\mathcal{B}_{ind}$ simulates the answers by invoking its own decryption oracle. Since the security of the encryption scheme is provided under the hardness of CDH problem, the decryption query $\mathcal{B}_{ind}$ invokes the action of $\mathcal{B}_{cdh}$ algorithm, who answers with a random value $\tilde{\delta}_i$, which $\mathcal{B}_{ind}$ on his side provides to $\mathcal{A}_{ake}$ adversary. The hardness of CDH problem assures that $\mathcal{B}_{ind}$ cannot distinguish which message was used in the encryption process of the IND-CPA game. This means that $\mathcal{A}_{ake}$ cannot distinguish the response of $\mathcal{B}'_{ind}$ from random. Thus, $\mathcal{A}_{ake}$ is not able to get any information about the real decryption key, ie, it cannot distinguish δ_i from $\tilde{\delta}_i$, meaning that in best case, the adversary can guess the key with a probability of $1/2^{\lambda}$. That means that $\mathcal{A}_{ake}$ advantage in Game-g_6 differs from the previous advantage by a negligible value only, ie,

$$|Pr[E_6] - Pr[E_5]| \leq \frac{q_d}{2^{\lambda}},$$

and hence

$$\begin{aligned}\rho_5 &= |2Pr[E_5] - 1| \\ &\leq |2Pr[E_5] - 2Pr[E_6]| + |2Pr[E_6] - 1| \\ &\leq 2\frac{q_d}{2^{\lambda}} + \rho_6.\end{aligned}$$

Game-g_7:

The difference of this game to the previous one is that the Test session aborts if $\mathcal{A}_{ake}$ issues a query $(k_1||\ldots||k_n||pid_i||sid_i)$. Since the adversary does not get any information about k_n, it can only guess the value (of k_n), with a probability $1/2^{\lambda}$. Thus, $\mathcal{A}_{ake}$ can request at most $\frac{q_h}{2^{\lambda}}$ correct random oracle queries for the Test session. Hence,

$$|Pr[E_7] - Pr[E_6]| \leq \frac{q_h}{2^{\lambda}},$$

and hence

$$\begin{aligned}\rho_6 &= |2Pr[E_6] - 1| \\ &\leq |2Pr[E_6] - 2Pr[E_7]| + |2Pr[E_7] - 1| \\ &\leq 2\frac{q_h}{2^{\lambda}} + \rho_7.\end{aligned}$$

The advantage ρ_7 is 0, if $\mathcal{A}_{ake}$ does not issue a random oracle query on the correct value $(k_1||\ldots||k_n||pid_i||sid_i)$.

Combining all the probabilities from the above games, the advantage of $\mathcal{A}_{ake}$

$$\left(n^2 Adv_{\mathcal{A}_{cma}} + \frac{(q_e + 3q_s + q_h)^2}{2^{\lambda}} + \frac{q_s^2}{2^{\lambda}} + nq_{H_3} Adv_{CDH} + \frac{(q_d + q_h)}{2^{\lambda}}\right)$$

is negligible.

Remark 2. The AKE security of our protocol assuming existence of an *outsider* adversary can be proved following the same technique as deployed above with a condition that the adversary $\mathcal{A}_{ake}$ must be passive for any party that it corrupts. In other words, in contrast to the AKE security game with insider KCI adversary, in the outsider KCI attack case, the goal of $\mathcal{A}_{ake}$ is to mount KCI attack by impersonating any uncorrupted party to an uncorrupted instance at any of the corrupted party (instead of any outsider party, as considered in the insider KCI attack).

The another important notion of security for GKA protocol is MA. In the following, we prove this security for our protocol.

Theorem 2. *The proposed TGKA protocol is MA secure with KCIR adversary following Definitions 6 and 7 of Section 3.1 assuming the existence of random oracle and unforgeability of the underlying signature scheme. The probability of success of the adversary $\mathcal{A}_{ma}$ (adversary against the MA security of our TGKA protocol) is the following advantage*

$$n^2 Adv_{\mathcal{A}_{cma}} + \frac{(q_e + 3q_s + q_h)^2}{2^{\lambda}} + \frac{q_s^2}{2^{\lambda}}$$

considering n parties in the network, $Adv_{\mathcal{A}_{cma}}$ as the advantage of the adversary $\mathcal{A}_{cma}$ against the unforgeable signature scheme, λ the security parameter, q_e the maximum number of Execute queries, q_s the maximum number of Send queries, $q_h = q_H + q_{H_2} + q_{H_3}$, where q_{H_i} is maximum number of hash H_i (random oracle) queries that the adversary $\mathcal{A}_{ma}$ can ask.

Proof. To show the MA in our protocol, we follow the game hopping technique exactly as considered in the above Theorem 1. For this, let E_i be the event that $\mathcal{A}_{ma}$ violates the definition of MA in Game-g_i. □

The sequence of games are as follows.

Game-g_0:

This is the original MA game as constructed in Definitions 6 and 7.The advantage of the adversary $\mathcal{A}_{ma}$ is

$$Adv_{\mathcal{A}_{ma}} = Pr[E_0].$$

Game-g_1:

This game is the same as Game-g_0 other than a case that the simulation fails if an event `Forge` happens. It is the same event as considered in Game-g_1 of the above Theorem 1. Hence, similarly, like the previous proof

$$|Pr[E_1] - Pr[E_0]| \leq Pr[\texttt{Forge}] \leq n^2 Adv_{\mathcal{A}_{cma}}.$$

Game-g_2:

This is the same as the previous game except that the simulation fails if an event `Collision` appears, with the same meaning of `Collision` as considered in above Theorem 1. Hence,

$$|Pr[E_2] - Pr[E_1]| \leq Pr[\texttt{Collision}] \leq \frac{(q_e + 3q_s + q_h)^2}{2^\lambda}.$$

Game-g_3:

This game is the same as Game-g_2 other than a case that the simulation fails if an event `Repeat` appears, with the same meaning of `Repeat` as considered in above Theorem 1. Hence,

$$|Pr[E_3] - Pr[E_2]| \leq Pr[\texttt{Repeat}] \leq \frac{(q_s + q_e)^2}{2^\lambda}.$$

If Game-g_3 does not abort, all the honest partnered parties compute the same key, so $Pr[E_3] = 0$.

Combining all the above probabilities, the advantage of $\mathcal{A}_{ma}$ is $n^2 Adv_{\mathcal{A}_{cma}} + \frac{(q_e+3q_s+q_h)^2}{2^\lambda} + \frac{q_s^2}{2^\lambda}$, which is negligible in λ.

7 | CONCLUSION

This paper introduces a novel TGKA protocol to be suitable for the discussed electronic communication. We have considered real-time issues such as the leakage of long-term secret key or faulty random number generation to leak ephemeral secrets. We hope the proposed GKA protocol will provide a basis and ready reference for future work and motivate researchers to achieve a protocol in one round, which is still a problem of interest in this domain. The expansion and

shrinking of existing security zones will be considered as a future work. Furthermore, in order to evaluate the expected benefits of the proposed solution, there is a need to implement this solution on real-time MPSoC platform.

ACKNOWLEDGMENT

This research has been performed in the context of Self-Organising circuits For Interconnected, Secure and Template computing (SOFIST) project, supported by Project ARC (Concerted Research Action) of Federation Wallonie-Bruxelles.

ORCID

Gaurav Sharma https://orcid.org/0000-0003-2842-3788

REFERENCES

1. Sharma G, Kuchta V, Sahu RA, Ellinidou S, Markowitch O, Dricot JM. A twofold group key agreement protocol for NoC based MPSoCs. In: Proceedings of the 16th Annual Conference on Privacy, Security and Trust (PST); 2018; Belfast, UK.
2. Wu Q, Mu Y, Susilo W, Qin B, Domingo-Ferrer J. Asymmetric group key agreement. In: *Advances in Cryptology - EUROCRYPT 2009: 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2009.
3. Bellare M, Rogaway P. Entity authentication and key distribution. In: *Advances in Cryptology - CRYPTO' 93: 13th Annual International Cryptology Conference Santa Barbara, California, USA August 22-26, 1993 Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 1993.
4. Shamir A. Identity-based cryptosystems and signature schemes. In: *Advances in Cryptology: Proceedings of CRYPTO 84*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 1984.
5. Diffie W, Hellman ME. New directions in cryptography. *IEEE Trans Inf Theory*. 1976;22(6):644-654.
6. Burmester M, Desmedt Y. A secure and efficient conference key distribution system. In: *Advances in Cryptology - EUROCRYPT'94: Workshop on the Theory and Application of Cryptographic Techniques Perugia, Italy, May 9-12, 1994 Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 1994.
7. Ingemarsson I, Tang D, Wong C. A conference key distribution system. *IEEE Trans Inf Theory*. 1982;28(5):714-720.
8. Steiner M, Tsudik G, Waidner M. Key agreement in dynamic peer groups. *IEEE Trans Parallel Distrib Syst*. 2000;11(8):769-780.
9. Joux A. A one round protocol for tripartite Diffie–Hellman. In: *Algorithmic Number Theory: 4th International Symposium, ANTS-IV Leiden, The Netherlands, July 2-7, 2000. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2000.
10. Barua R, Dutta R, Sarkar P. Extending Joux's protocol to multi party key agreement. In: *Progress in Cryptology - INDOCRYPT 2003: 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2003.
11. Bresson E, Chevassut O, Pointcheval D. Provably authenticated group Diffie-Hellman key exchange-the dynamic case. In: *Advances in Cryptology - ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9-13, 2001 Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2001.
12. Bresson E, Chevassut O, Pointcheval D. Dynamic group Diffie-Hellman key exchange under standard assumptions. In: *Advances in Cryptology - EUROCRYPT 2002: International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam, The Netherlands, April 28 - May 2, 2002 Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2002.
13. Bresson E, Chevassut O, Pointcheval D, Quisquater J-J. Provably authenticated group Diffie-Hellman key exchange. In: Proceedings of the 8th ACM Conference on Computer and Communications Security; 2001; Philadelphia, PA.
14. Katz J, Yung M. Scalable protocols for authenticated group key exchange. In: *Advances in Cryptology - CRYPTO 2003: 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2003.
15. Katz J, Shin JS. Modeling insider attacks on group key-exchange protocols. In: Proceedings of the 12th ACM Conference on Computer and Communications Security; 2005; Alexandria, VA.
16. Gorantla MC, Boyd C, Nieto JMG. Modeling key compromise impersonation attacks on group key exchange protocols. In: *Public Key Cryptography - PKC 2009: 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2009.
17. Zhao J, Gu D, Gorantla MC. Stronger security model of group key agreement. In: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security; 2011; Hong Kong.
18. Tseng Y-M, Tsai T-T, Huang S-S. Enhancement on strongly secure group key agreement. *Security Commun Netw*. 2015;8(2):126-135.
19. Schnorr CP. Efficient identification and signatures for smart cards. In: *Advances in Cryptology - CRYPTO' 89 Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 1989.
20. Choi KY, Hwang JY, Lee DH. Efficient ID-based group key agreement with bilinear maps. In: *Public Key Cryptography - PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography, Singapore, March 1-4, 2004. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2004.
21. Zhang F, Chen X. Attack on an ID-based authenticated group key agreement scheme from PKC 2004. *Inf Process Lett*. 2004;91(4):191-193.

22. Shim K-A. Further analysis of ID-based authenticated group key agreement protocol from bilinear maps. *IEICE Trans Fundam Electron Commun Comput Sci*. 2007;90(1):295-298.
23. Wu T-Y, Tseng Y-M, Yu C-W. A secure ID-based authenticated group key exchange protocol resistant to insider attacks. *J Inf Sci Eng*. 2011;27(3):915-932.
24. Wu T-Y, Tseng Y-M, Tsai T-T. A revocable ID-based authenticated group key exchange protocol with resistant to malicious participants. *Computer Networks*. 2012;56(12):2994-3006.
25. Wu T-Y, Tseng Y-M. Towards ID-based authenticated group key exchange protocol with identifying malicious participants. *Informatica*. 2012;23(2):315-334.
26. Wei F, Wei Y, Ma C. Attack on an ID-based authenticated group key exchange protocol with identifying malicious participants. *Int J Netw Secur*. 2016;18(2):393-396.
27. Wu T-Y, Tsai T-T, Tseng Y-M. A provably secure revocable ID-based authenticated group key exchange protocol with identifying malicious participants. *Sci World J*. 2014;2014. Article ID 367264.
28. Teng J, Wu C, Tang C, Tian Y. A strongly secure identity-based authenticated group key exchange protocol. *Sci China Inf Sci*. 2015;58(9):1-12.
29. Teng J, Wu C. A collusion attack on asymmetric group key exchange. *Security Commun Netw*. 2015;8(13):2189-2193.
30. Wu Q, Qin B, Zhang L, Domingo-Ferrer J, Farràs O. Bridging broadcast encryption and group key agreement. In: *Advances in Cryptology - ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2011.
31. Wu Q, Qin B, Zhang L, Domingo-Ferrer J, Manjón JA. Fast transmission to remote cooperative groups: a new key management paradigm. *IEEE/ACM Trans Netw*. 2013;21(2):621-633.
32. Zhang L, Wu Q, Qin B, Domingo-Ferrer J. Identity-based authenticated asymmetric group key agreement protocol. In: *Computing and Combinatorics: 16th Annual International Conference, COCOON 2010, Nha Trang, Vietnam, July 19-21, 2010. Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2010.
33. Zhang L, Wu Q, Qin B, Domingo-Ferrer J. Provably secure one-round identity-based authenticated asymmetric group key agreement protocol. *Information Sciences*. 2011;181(19):4318-4329.
34. Zhao X, Zhang F, Tian H. Dynamic asymmetric group key agreement for ad hoc networks. *Ad Hoc Netw*. 2011;9(5):928-939.
35. Zhang L, Wu Q, Domingo-Ferrer J, Qin B, Chow SS, Shi W. Secure one-to-group communications escrow-free ID-based asymmetric group key agreement. In: *Information Security and Cryptology: 9th International Conference, Inscrypt 2013, Guangzhou, China, November 27-30, 2013, Revised Selected Papers*. Cham, Switzerland: Springer International Publishing; 2013.
36. Li M, Xu X, Guo C, Tan X. AD-ASGKA–authenticated dynamic protocols for asymmetric group key agreement. *Secur Commun Netw*. 2016;9(11):1340-1352.
37. Sepúlveda J, Gogniat G, Flórez D, Diguet J-P, Zeferino C, Strum M. Elastic security zones for NoC-based 3D-MPSoCS. In: Proceedings of the 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS); 2014; Marseille, France.
38. Sepúlveda J, Flórez D, Gogniat G. Reconfigurable group-wise security architecture for NoC-based MPSoCs protection. In: Proceedings of the 28th Symposium on Integrated Circuits and Systems Design; 2015; Salvador, Brazil.
39. Young CP, Chia CC, Chen LB, Huang J. On-chip-network cryptosystem: a high throughput and high security architecture. In: Proceedings of the 2008 IEEE Asia Pacific Conference on Circuits and Systems; 2008; Macao, China.
40. English T, Popovici E, Keller M, Marnane WP. Network-on-chip interconnect for pairing-based cryptographic IP cores. *J Syst Archit*. 2011;57(1):95-108.
41. Sepúlveda J, Flórez D, Gogniat G. Reconfigurable security architecture for disrupted protection zones in NoC-based MPSoCs. In: Proceedings of the 10th International Symposium on Reconfigurable Communication-Centric Systems-on-Chip (ReCoSoC); 2015; Bremen, Germany.
42. Sepúlveda J, Flórez D, Gogniat G. Efficient and flexible NoC-based group communication for secure MPSoCs. In: Proceedings of the International Conference on ReConFigurable Computing and FPGAs (ReConFig); 2015; Mexico City, Mexico.
43. Sepulveda J, Flórez D, Immler V, Gogniat G, Sigl G. Hierarchical group-key management for NoC-based MPSoCs protection. *J Integr Circuits Syst*. 2016;11(1):38-48.
44. Sepulveda J, Flórez D, Immler V, Gogniat G, Sigl G. Efficient security zones implementation through hierarchical group key management at NoC-based MPSoCs. *Microprocess Microsyst*. 2017;50:164-174.
45. Zhang L, Wu Q, Domingo-Ferrer J, Qin B, Dong Z. Round-efficient and sender-unrestricted dynamic group key agreement protocol for secure group communications. *IEEE Trans Inf Forensics Secur*. 2015;10(11):2352-2364.
46. Fujisaki E, Okamoto T. Secure integration of asymmetric and symmetric encryption schemes. In: *Advances in Cryptology - CRYPTO' 99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15-19, 1999 Proceedings*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 1999.
47. Hess F. Efficient identity based signature schemes based on pairings. In: *Selected Areas in Cryptography: 9th Annual International Workshop, SAC 2002 St. John's, Newfoundland, Canada, August 15-16, 2002 Revised Papers*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2002.
48. Horng S-J, Tzeng S-F, Pan Y, et al. B-SPECS+: batch verification for secure pseudonymous authentication in VANET. *IEEE Trans Information Forensics Security*. 2013;8(11):1860-1875.
49. Sharma G, Sahu RA, Kuchta V, Markowitch O, Bala S. Authenticated group key agreement protocol without pairing. In: *Information and Communications Security: 19th International Conference, ICICS 2017, Beijing, China, December 6-8, 2017, Proceedings*. Cham, Switzerland: Springer International Publishing; 2017.

50. Bresson E, Manulis M. Securing group key exchange against strong corruptions. In: Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security; 2008; Incheon, South Korea.

51. Dent AW. A note on game-hopping proofs. *IACR Cryptol ePrint Arch*. 2006;2006:260.

How to cite this article: Sharma G, Kuchta V, Anand Sahu R, et al. A twofold group key agreement protocol for NoC-based MPSoCs. *Trans Emerging Tel Tech*. 2019;30:e3633. https://doi.org/10.1002/ett.3633