A Privacy-Preserved Joint Group Time Scheduling Mechanism for Mobile Social Applications

Yu-Jia Chen, *Student Member, IEEE*, Chia-Yu Lin, *Student Member, IEEE* and Li-Chun Wang, *Fellow, IEEE,*
National Chiao Tung University, Taiwan
Email :allan920693.cm99g@nctu.edu.tw , sallylin1987.cm01g@nctu.edu.tw and lichun@cc.nctu.edu.tw

***Abstract*— In this paper, we develop a privacy preserving mechanism for a scheduling service for a group of users, which can compute the joint available time based on users' calendar information. We also propose a weighted voting game with partially homomorphic encryption (WVGPHE) scheme to protect the privacy of users' calendar information as well as users' weight. Even with the privacy protection, our experimental results show that the proposed scheme is still very computational and bandwidth efficient enough to provide immediate group meeting in mobile communications environments.**

General Terms: Design, Experimentation

Keywords: Mobile social network, weighted voting system

I. Introduction

Mobile social networks (MSNs) have been continuing growth rapidly in recent years. In MSNs, many friends can share their calendars to arrangement a meeting, thereby forming a so-called group-ware calendar system (GCS). In GCS of MSN, the meeting organizer can find out the joint available time through the electronic calendars of group users in a very short time, instead of calling every friends one by one. Augar is an example of this system [1]. The calendar merge-up tasks to determine the joint available time can be implemented for a large-scale user calendar data sizes for MSN on cloud computing platform.

However, users' calender data are private information, and thus these data should be kept secret in GCS. In [2], the author suggested that users could restrict the shared data by explicitly using privacy settings or used cryptic and context-sensitive entries techniques to make audience understand the meaning of the appointments. However, these techniques can only prevent violating privacy from other users, but not the servers themselves.

Moreover, in current MSNs, a user are sometimes weighted according to his/her social status. For example, every user has a friend ranking list in Facebook [3]. The ranking list of a user shows the ranking of user's friends whom he/she interacts most frequently. However, in GCS of MSNs, the meeting organizer cannot set the weight for each user, leading to an important user in that group be possibly absent in a meeting.

To overcome these limitations of organizing meeting service in GCS of MSNs, we develop a group time scheduling service for MSNs. In this service, the meeting organizer can not only set the weights of the users according to the social level or his/her preference, but also can find the joint available time of a group of users without revealing users' calender information. Such a service can be modeled as a weighted voting system with secret ballot. The major challenges of designing this kind of service are listed as follows:

- Since the weight setting is on behalf of the preference/importance of a user, users' weight should be kept secret from users and servers. The considered scenario is thus different from current weighted voting system, in which each user can know his/her weight.
- The group time scheduling service is executed on mobile devices, thereby requiring a light-weight algorithm to complete the execution in a short period of time.

In this paper, we propose a weighted voting game with partially homomorphic encryption scheme (WVGPHE scheme) to protect the privacy of the users' calendar information as well as the users' weight in their social group. We propose to use homomorphic encryption techniques to protect the calendar information as well as users' weight. In particular, we propose a table lookup method by applying partially homomorphic encryption and a role analysis for the weighted voting game to improve the computation/bandwidth efficiency. Our experiment results shows that the proposed mechanism can provide an effective and fast group time scheduling service in a mobile communications environment.

The rest of the paper is organized as follows: Section II introduces the related work of electronic voting system. Section III describes the system model and security requirement of the group time scheduling service. Preliminaries and definitions are given in Section IV. Section V details WVGPHE scheme. Section VI shows the performance evaluation of WVGPHE scheme. Finally, Section VII concludes the paper.

II. RELATED WORK

In a voting system, the voters elect a winner from a few candidates. At first, the candidates and the number of expected winners are declared by the authorities. Then every voter can elect the candidates using ballots. At last, some talliers count the ballots and announce the voting result. One of the most essential properties of a voting system is the privacy of users' votes. Secret-ballot voting has been studied extensively in the literature. Two main methods have been applied to develop a secret ballot protocol: mix network [4]–[8] and homomorphic voting [9]–[14]. In the mix network scheme, ballots are shuffled in a network of servers which takes a set of ciphertext as input and deliver the corresponding plaintext with a certain permutation as output. The vote privacy is preserved since the authorities cannot linked the ballot to the voter. Voting schemes using homomorphic encryption are called homomorphic voting in this paper. Homomorphic encryption allows computing operations for encrypted data. However, constructing an efficient encryption scheme with both additively and multiplicatively homomorphic remains a tantalizing open question. Thus, current homomorphic voting employs additively homomorphic encryption (e.g. , Paillier encryption) to encrypt the ballots and exploits the additively homomorphism to recover the sum of the votes with a single decryption. Since none of the votes is decrypted, the vote privacy is preserved. It is widely believed that the homomorphic voting is more suitable for a small number of candidates or choices, such as a "YES/NO" voting [15]. In the practical electronic voting system, the ballots are sometimes tailed in proportion to the voters' weights. Such a system can be analyzed using weighted voting game (WVG), which provides a model of decision-making in many political and economical bodies, such as the International Monetary Fund [16]. In a WVG, each player has a weight, and a coalition of players wins the game if the sum of the weights exceed a certain threshold. Since each voter's weight is determined by his/her private information (e.g. , shares), it is necessary to keep a voter's weight secret, in addition to his/her votes. To realize the secrecy of weights, a shuffling protocol for server to shuffle a list of encrypted votes is proposed in [17]. Our work is inspired by [18], which developed a weighted voting protocol with secret weights using homomorphic encryption. However, the authors in [18] adopted the model that each user can know his/her weight. In contrast, we consider the scenario that only an activity organizer can define the weights of the users. Unlike the previously described works focusing on improving the computational efficiency in the tallying server, our work aims at exploiting a lightweight voting scheme for mobile social applications.

III. SYSTEM MODEL AND REQUIREMENT

The system architecture is composed of three components: the cloud server, the trusted key distribution server, and the mobile users. Fig. 1 illustrates the system architecture for the proposed scheme. The cloud server is assumed to have infinite computing resources, while mobile users has limited computation resources. Mobile users are partitioned into many groups, where an activity organizer of a group can define the j-th user's weight w_j of a group and a threshold q of an activity. A calendar vector $\mathbf{v}$ of a user represent the status of a user's calendar

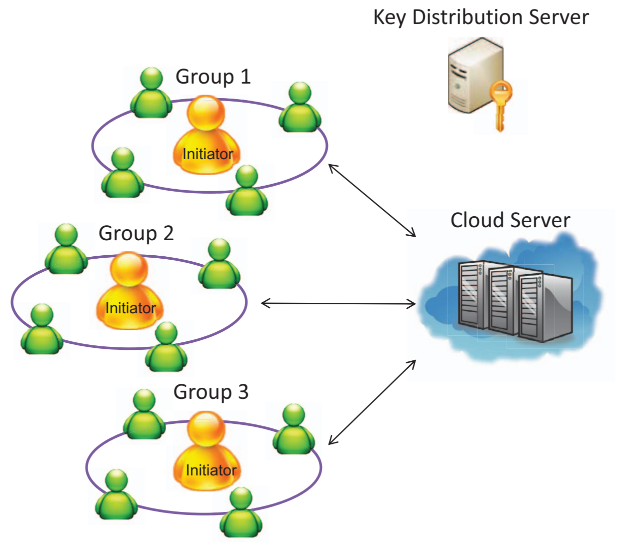

Fig. 1. System Architecture of WVGPHE scheme.

in a time interval. The value of is "0" means that the user has reserved schedule (i.e, busy) in the i-th time slot, while the value of is "1" means that the user has no plan (i.e, free). A time slot is called a **joint available time slot** if $\sum_j w_j v_i \geq q$. Users are not able to access the joint available time of other groups. For adversary model, we assume that the cloud server as well as the mobile users are honest-but-curious and noncollusive [19]. Also, an organizer is assumed to be trusted in the voting (i.e, he/she will not manipulate voting result). The goal of the proposed scheme is to compute the joint available time slot for a organizer without the need of revealing any users' weights and the calendar vector to the server.

IV. PRELIMINARIES AND DEFINITIONS

A. *Weighted Voting Games*

A **weighted voting game** (WVG) $G = [q : \mathbf{w}]$ is given by a vector of players' **weights** $\mathbf{w} = (w_1, ..., w_n) \in (R^+)^n$ and a **quota** $q \in R^+$. In these games, a coalition is winning if the sum of its total weight exceeds the threshold quota q. Formally, weights w_i assigned to each player $i \in N$. By convention, we take $w_i \geq w_j$ if $i < j$. Let $W(N)$ be $\sum_{i \in N} w(i)$. For any $C \subseteq N$ we have $v(C) = 1$ if $W(C) > q$, and $v(C) = 0$ otherwise. We assume that $W(N) \geq q$, i.e., the grand coalition is winning. These two WVGs are **equivalent** if there is a way for all the player of the first system to exchange places with the player of the second system and preserves all winning coalitions. For example, $[5 : (4, 1)]$ and $[4 : (3, 3)]$ are equivalent because each player requires other player unanimous support to pass the quota.

A player i is called a **dummy** if he contributes nothing to any coalition. That is, for any $C \subseteq N$ we have $v(C \cup \{i\}) = v(C)$. A dummy player has no influence on the result of the vote. A player i is **critical** in a winning coalition C if the removal from that coalition would make it a losing coalition. That is, $v(C) = 1, v(C\backslash\{i\}) = 0$. A player i is a **dictator** if he/she is present in in every winning coalition and absent from every losing coalition. That is, for player i to be the dictator, $w(i) \geq q$ and $\sum_{2 \leq i \leq n} w(i) < q$. It is obvious that if a dictator exist, he/she is unique.

A common interpretation of the power index of a player is the probability of having a significant role in the game. One of the most prominent power index is the Banzhaf index [20]. This index has been widely used for the purpose of measuring the ability of a player in a WVG to determine the outcome of the vote. The Banzhaf index depends on the number of coalitions in which a player is critical out of all possible coalitions. The **Banzhaf index** $\beta_i(G)$ of a player i in a game G is computed as follows:

$$\beta_i(G) = \frac{1}{2^{n-1}} \sum_{C: i \notin C} [v(C \cup \{i\}) - v(C)]$$

Two WVGs are equivalent if the Banzhaf index of each player in the first system is the same as that of the player in the second system.

B. Homomorphic Encryption

The homomorphic property of public-key cryptosystems is often applied in current cryptographic protocols since operations can be performed without revealing the plaintext. Let $E(m)$ be a ciphertext of a plaintext m. Then, a **homomorphic encryption** satisfies the following equation:

$$E(m_1) \otimes E(m_2) = E(m_1 \oplus m_2)$$

, where $(\otimes, \oplus)$ are some group operation in the ciphertext and plaintext space respectively. For example, if $\oplus$ represents modular addition, we call such a scheme **additively homomorphic**. A scheme is **partially homomorphic** if it allows homomorphic computation of only one operation on plaintexts ($\oplus$ represents modular addition or modular multiplication). We denote $E_k(m, r)$as the encryption of plaintext m with public key k and a random number r.

V. WEIGHTED VOTING GAME WITH PARTIALLY HOMOMORPHIC ENCRYPTION SCHEME

The proposed WVGPHE scheme are composed of four phases: preparing, setup, voting and tallying. Preparing phase is off-line, in which voting can be done before an activity organizer runs a group time scheduling service. The remaining three phases are done in an on-line manner.

A. Preparing Phase

During this phase, we build the knowledge of WVGs, including the role and the power of each player in a WVG. At first, an activity organizer sends huge amounts of WVG data to the cloud server. Then, the cloud server computes the knowledge of each WVG and sends the outcome to the organizer. Finally, the database of the knowledge of WVGs are constructed. The detailed protocol is as follows:

1) An organizer sends WVG vector $\mathbf{G} = [G_1, ..., G_m]$ to the cloud server.
2) The cloud server computes role vector $\mathbf{r}(\mathbf{G})_i = [r(G_1)_i, ..., r(G_m)_i]$ for the i-th player and Banzhaf index vector $\boldsymbol{\beta}(\mathbf{G})_i = [\beta(G_1)_i, ..., \beta(G_m)_i]$ where $r(G_j)_i = \infty$ if the i-th player of the game G_j is a dictator;$r(G_j)_i = 1$ if the i-th player of game G_j is a non-dummy player;$r(G_j)_i = 0$ if the i-th player of game G_j is a dummy. Then sends $(\mathbf{r}(\mathbf{G}), \boldsymbol{\beta}(\mathbf{G}))$ to the organizer.

B. Setup Phase

When an organizer needs to execute a group time scheduling service, he/she first specifies a time range, a time length, an effective invitee set and a WVG for this activity. The effective invitee set and the WVG are sent to the key distribution server. The time length defines the duration of the meeting/activity. The WVG, which has been analyzed in the preparing phase, is used to define the weight of each invitee and the quota of the activity. Given a WVG, an organizer can select its equivalent WVG (by comparing Banzhaf index vector). The effective invitee set is constructed according the role vector of the WVG and the initial invitee set. The pseudocode of constructing an effective invitee set is given in Algorithm 1. The organizer also send the time range, the time length and the effective invitee set to the cloud server. Then the key distribution server sends the group key and the time range to the invitees in the effective invitee set and then constructs a increment table of each invitee. The purpose of the increment table is to transform the weighted (multiplicative) operation to the addictive operation based on the following homomorphic property:

$$\begin{aligned} & E(m \times w) \\ & = E(m \times (w-1) + m) \\ & = E(m \times (w-1)) \otimes E(m) \end{aligned}$$

The pseudocode of constructing an increment table is given in Algorithm 2. Finally, the increment table is sent to the cloud server. The following details the proposed protocol:

1) An organizer generates a public group key k and selects time slot vector $\mathbf{T} = (T_1, ..., T_n)$, time length l, an initial invitee set $\mathbf{I}$, and a WVG $\mathbf{G}$. Then computes effective invitee set $\mathbf{I_e}$ by Algorithm 1. Send $(\mathbf{G}, \mathbf{I_e}, k)$ to the key distribution server and $(\mathbf{T}, l, \mathbf{I_e})$ to the cloud server.
2) For the each effective invitee $I_e(i)$, the key distribution server selects a random number r_i. Then computes the increment table $\mathbf{A}_i$ by Algorithm 2. Send $(\mathbf{T}, k, r_i)$ to $I_e(i)$ and $\mathbf{A}_i$ to the cloud server.

C. Voting Phase

In this phase, the effective invitees encrypt the calendar data using the key sent by the key distribution server. Then the encrypted calendar data (secret ballot) is sent to the cloud server. The detailed protocol is described as follows:

1) $I_e(i)$ compute encrypted calendar vector $\mathbf{v'}_i = [E_k(v_1, r_i), ..., E_k(v_n, r_i)]$, and then send $\mathbf{v'}_i$ to the cloud server.

Algorithm 1

```
1: Input: Initial invitee set $\mathbf{I}$ and role vector $\mathbf{r}(G)$
2: Output: Effective invitee set $\mathbf{I_e}$
3: Set $\mathbf{I_e} = \emptyset$
4: Select $i \in \mathbf{I}$
5: if $r(G)_i = \infty$ then
6:     $\mathbf{I_e} = \{i\}$, go to step 7
7: else
8:     if $r(G)_i = 1$ then
9:         $\mathbf{I_e} = \mathbf{I_e} \cup \{i\}$ and $\mathbf{I} = \mathbf{I} - \{i\}$ , go to step 6
10:    end if
11: else
12:    go to step 6
13: end if
```

Algorithm 2

```
1: Input: Weight $w$, random number $r$, public key $k$, and effective invitee set $\mathbf{I_e}$
2: Output: A $2 \times 2$ Increment table $\mathbf{A}$
3: Generate a random bit $b \in \{0, 1\}$ and a select a shift $\varepsilon < \frac{1}{|\mathbf{I_e}|}$ where $|\mathbf{I_e}|$ denote the number of elements in $\mathbf{I_e}$.
4: if $b = 0$ then
5:     $\mathbf{A} = \begin{bmatrix} E_k(0, r) & E_k(1, r) \\ E_k(\varepsilon, r) & E_k(w-1, r) \end{bmatrix}$
6: else
7:     $\mathbf{A} = \begin{bmatrix} E_k(1, r) & E_k(0, r) \\ E_k(w-1, r) & E_k(\varepsilon, r) \end{bmatrix}$
8: end if
9: Return $\mathbf{A}$
```

D. Tallying Phase

In the final phase, the cloud server adds an increment to each encrypted calendar data by looking up its increment table. After that, the cloud server computes the encrypted vote result by summing the encrypted calendar data within a time length. The pseudocode of constructing an encrypted vote result is given in Algorithm 3. The encrypted vote result is then sent to the organizer. Finally, the encrypted vote results are decrypted by using the organizer's secret key. The detailed protocol is explained as follows:

1) The cloud server computes encrypted vote result vector $\mathbf{C'}$ by Algorithm 3. And then send $\mathbf{C'}$ to the organizer.
2) The organizer decrypts $\mathbf{C'}$ using his/her secret key and gets vote result $\mathbf{C}$. The joint available time slot is the time range from T_k to T_{k+l-1} which satisfies $C(k) \geq q$.

Algorithm 3

```
1: Input: Encrypted calendar vector $\mathbf{v'}_i$, increment table $\mathbf{A}_i$, time length $l$, size of effective invitee set $|\mathbf{I_e}|$, and size of time slot vector $|\mathbf{T}|$
2: Output: Encrypted vote result vector $\mathbf{C'}$
3: for $i := 1$ to $|\mathbf{I_e}|$ do
4:     for $j := 1$ to $|\mathbf{T}|$ do
5:         if $v'_i(1) = A_i(1, 1)$ then
6:             $v'_i(j) = v'_i(j) \otimes A_i(2, 1)$
7:         else
8:             $v'_i(j) = v'_i(j) \otimes A_i(2, 2)$
9:         end if
10:    end for
11: end for
12: for $k = 1$ to $|\mathbf{T}| - l + 1$ do
13:     $C'(k) = \sum_{i=1}^{|I_e|} \sum_{j=k}^{k+l-1} v'_i(j)$
14: end for
15: Return $\mathbf{C'}$
```

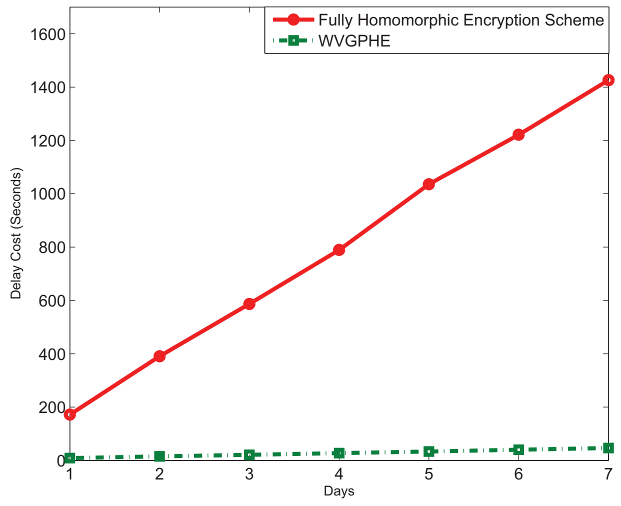

Fig. 2. Voting Delay cost of WVGPHE scheme and fully homomorphic encryption scheme.

VI. PERFORMANCE EVALUATION

We compare the delay and bandwidth performance of the proposed scheme with fully homomorphic encryption scheme. We adopt Paillier encryption [21] for the additively homomorphic encryption in our scheme. The fully homomorphic encryption scheme is implemented based on the work in [22] for the initial concept and that in [23] for the integer-based implementation approach. We conduct the experiment by running on a personal computer as the cloud server. The personal computer is equipped with Intel Core i5 processor running at 2.4 GHz, 8 GB of RAM. We use HTC Desire with Android 2.1 to be the mobile device. The key length is 3080 bits (which is equivalent to AES-128 in security strength).

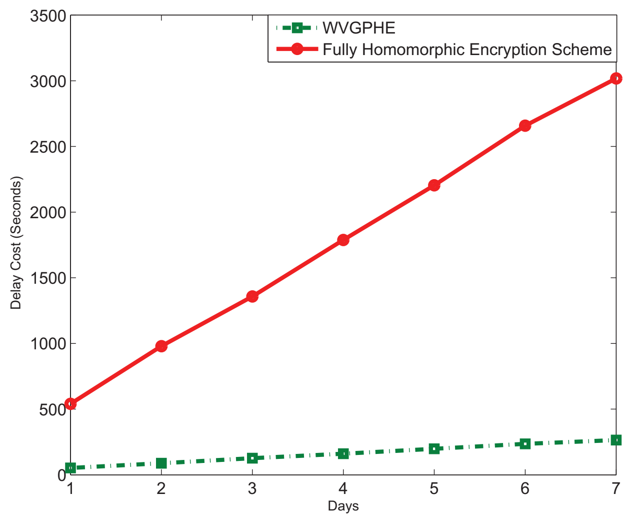

Fig. 3. Service Delay Cost of WVGPHE scheme and fully homomorphic encryption scheme.

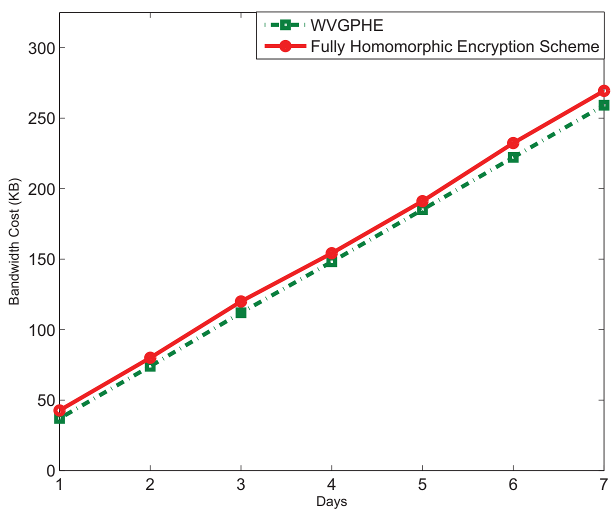

Fig. 4. Bandwidth Cost of WVGPHE scheme and fully homomorphic encryption scheme.

A. Delay Cost

Figure 2 shows the voting delay cost of WVGPHE scheme and the fully homomorphic encryption scheme on mobile devices. The voting delay cost is the execution time of the voting phase for the mobile device. The result shows that WVGPHE scheme can reduce about 95% of delay cost than the fully homomorphic encryption scheme. Fig. 3 shows the performance comparison in terms of of service delay cost. The service delay cost is the execution time of the entire service (four phases) with the calendar data of 7 days and a group of 20 users. The result shows that WVGPHE scheme can finish the service within 5 minutes.

B. Bandwidth Cost

Figure 4 shows the bandwidth cost for the encrypted calendar data. The bandwidth cost of WVGPHE scheme is slightly reduced compared with fully homomorphic encryption scheme. Also, the proposed scheme can avoid the redundant communication and computation for the WVG by analyzing the role of each user. Even though the probability for a WVG to have a dictator is not high, the probability of having at least one dummy in a weighted voting game with a small number of player is very high. This probability can reach about 50% for 4, 5 or 6 players [24]. In other words, our scheme can reduce at least 10% of the bandwidth cost averagely.

VII. Conclusion

In this paper, we investigated the privacy issues in GCS of MSN from the aspects of users' calender information and users' weights in their corresponding social group. We develop a group time scheduling service to determine the joint available time based on users' calendar information without revealing the information of their calender and social status. To expedite the convergence of computing the encrypted joint available meeting time, we propose a weighted voting game with partially homomorphic encryption (WVGPHE) scheme. Our service mechanism consists of a novel table lookup method to apply partially homomorphic encryption technique to the weighted voting system. The analysis of WVG plays a role to reduce the redundant computation and communication in the scheduling service. Our experiments shows that the proposed scheme can provide group meeting services in mobile environments within a very short time, thereby improving the user experience on the encrypted mobile social applications. One possible future research direction of interest is to design a secure voting protocol with distributed key distribution servers since a centralized key server becomes both a single point of failure and an attractive attack target.

References

[1] J. Tullio, J. Goecks, E. Mynatt, and D. Nguyen, "Augmenting shared personal calendars," in *Symposium on User Interface Software and Technology: Proceedings of the 15 th annual ACM symposium on User interface software and technology*, 2002.

[2] L. Palen, "Social, individual and technological issues for groupware calendar systems," in *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, 1999.

[3] "Bookmarklet lets facebook users view the friend rankings list of those they interact with most," http://www.insidefacebook.com/2011/08/18/friend-rankings/.

[4] D. Chaum, "Secret-ballot receipts: True voter-verifiable elections," *Security & Privacy, IEEE*, vol. 2, no. 1, pp. 38–47, 2004.

[5] A. Fujioka, T. Okamoto, and K. Ohta, "A practical secret voting scheme for large scale elections," in *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*, 1993.

[6] M. Abe, "Mix-networks on permutation networks," *Advances in cryptology-ASIACRYPT*, pp. 258–273, 1999.

[7] R. Aditya, B. Lee, C. Boyd, and E. Dawson, "An efficient mixnet-based voting scheme providing receipt-freeness," *Trust and Privacy in Digital Business*, pp. 152–161, 2004.

[8] B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang, and S. Yoo, "Providing receipt-freeness in mixnet-based voting protocols," *Information Security and Cryptology-ICISC*, pp. 245–258, 2004.

[9] A. Kiayias and M. Yung, "Self-tallying elections and perfect ballot secrecy," in *Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems: Public Key Cryptography*, 2002.

[10] K. Peng and F. Bao, "Efficient vote validity check in homomorphic electronic voting," *Information Security and Cryptology–ICISC*, pp. 202–217, 2009.

[11] B. Lee and K. Kim, "Receipt-free electronic voting scheme with a tamper-resistant randomizer," *Information Security and Cryptology XICISC*, pp. 389–406, 2003.

[12] I. Damgård and M. Jurik, "A generalisation, a simpli. cation and some applications of paillier's probabilistic public-key system," in *Public Key Cryptography*, 2001.

[13] K. Peng and F. Bao, "Efficient proof of validity of votes in homomorphic e-voting," in *Fourth International Conference on Network and System Security*, 2010.

[14] R. Araújo, S. Foulle, and J. Traoré, "A practical and secure coercion-resistant scheme for remote elections," *Frontiers of Electronic Voting*, vol. 7311, 2007.

[15] K. Peng, R. Aditya, C. Boyd, E. Dawson, and B. Lee, "Multiplicative homomorphic e-voting," *Progress in Cryptology-INDOCRYPT*, pp. 1403–1418, 2005.

[16] D. Leech, "Voting power in the governance of the international monetary fund," *Annals of Operations Research*, vol. 109, pp. 375–397, 2002.

[17] T. Saisho, T. Saito, H. Doi, and S. Tsujii, "On the security of electronic weighted voting schemes," *Trans. of Inform. Process. Soc. of Japan*, vol. 44, pp. 1913–1923, 2003.

[18] N. F. Toru Nakanishi, Shinji Nakatake and Y. Sugiyama, "An efficient weighted voting protocol with secret weights," *International Symposium on Information Theory and its Applications (ISITA)*, 2004.

[19] A. Bain, J. Mitchell, R. Sharma, D. Stefan, J. Zimmerman, S. Chakraborty, and A. Kumar, "A domain-specific language for computing on encrypted data (invited talk)," in *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, 2011.

[20] P. Dubey and L. Shapley, "Mathematical properties of the banzhaf power index," *Mathematics of Operations Research*, vol. 4, pp. 99–131, 1979.

[21] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, 1999.

[22] C. Gentry, "Fully homomorphic encryption using ideal lattices," *Proceedings of the 41st annual ACM symposium on Theory of computing*, 2009.

[23] N. Smart and F. Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes," *Public Key Cryptography–PKC*, pp. 420–443, 2010.

[24] F. Barthélémy, D. Lepelley, and M. Martin, "On the likelihood of dummy players in weighted majority games," *Social Choice and Welfare*, pp. 1–17, 2011.