Topology-Hiding Broadcast Based on NTRUEncrypt

Bo Mi[1,2], Dongyan Liu[2]
[1] College of Information Science and Engineering, Chongqing Jiaotong University
Chongqing, 400074 - China
[e-mail: mi_bo@163.com]
[2] College of Civil Engineering, Chongqing University
Chongqing, 400044 - China
[e-mail: liudy@cqu.edu.cn]
*Corresponding author: Bo Mi

Received July 10, 2015; revised October 13, 2015; accepted October 28, 2015; published January 31, 2016

Abstract

Secure multi-party computation (MPC) has been a research focus of cryptography in resent studies. However, hiding the topology of the network in secure computation is a rather novel goal. Inspired by a seminal paper [1], we proposed a topology-hiding broadcast protocol based on NTRUEncrypt and secret sharing. The topology is concealed as long as any part of the network is corrupted. And we also illustrated the merits of our protocol by performance and security analysis.

Keywords: Topology-hiding, Multi-party Computation, NTRUEncrypt, Homomorphism, Broadcast

The paper is subsidized by Natural Science Foundation Project of CQ CSTC(cstc2011jjA30010) and Science Foundation of Chongqing Jiaotong University (20101cjc016).

http://dx.doi.org/10.3837/tiis.2016.01.025 **ISSN : 1976-7277**

1. Introduction

The basic problem of multi-party computation was phrased well over thirty years ago. Following the seminal works of Yao and Goldreich [2,3], numerous advances of MPC have been made ranging from general completeness results [4,5] to efficient solutions for specific problems [6-8]. As Goldwasser predicted, the field of multi-party computation is today where public-key cryptography was ten years ago [9], its competence of corporative computing with inputs concealed made it an extremely powerful tool and rich theory for real-life usage. Albeit secure multi-party computations are usually carried out on computer networks, one important question is always neglected: MPC over an underlying communication network whose structure might be, in itself, sensitive information. Taking social networks or vehicular networks for example, revealing the network topology could have severe privacy implications on nodes relationships or locations.

Aiming at the goal of designing an MPC protocol which computationally hides the underlying network topology, a seminal paper presented by Tal Moran [1] in early 2015 formally defined topology-hiding MPC in both indistinguishability game-based and simulation-based ways. Briefly speaking, if we model a network by a graph $G=(V,E)$ that is not fully connected, the game-based definition can be summarized as follows:

An adversary corrupts $A \subset V$ *and sends two network topologies* G_0, G_1 *on vertices* V. *These graphs must be so that the neighborhoods of* A *are the same. The challenger then picks* G_b *at random and returns the collective view of the parties in* A *resulting from the execution of the protocol on* G_b. *The adversary outputs* b' *and wins if* $b'=b$. *We say a protocol is secure against chosen topology attack if no static and computationally bounded adversary can win the above game with probability negligibly greater than if it simply guesses* b'.

For more details, a through reading of abovementioned paper [1] is highly recommended.

Besides the definition of topology-hiding MPC, a conceptual broadcast protocol which is secure as long as the adversary does not corrupt any whole neighborhoods of the network graph is also presented in [1]. However, Tal Moran simply assumed that there exists a secret-sharing scheme which can be used to hide the sensitive information among nodes in local neighborhood and left its realization behind. Inspired by Tal Moran's idea on topology-hiding broadcast, we exploit the semi-homomorphic property of modified NTRUEncrypt and devise an efficient topology-hiding broadcast protocol.

2. Related Work

In fact, the interplay between multi-party computation and networking is disregarded by most of cryptographic MPC literatures thus we refer the works in [1] as first contributions in such topic. Besides that, the only literature we can find related to topology-hiding MPC is that of Hinkelmann and Jakoby [10] who focused on the information theoretic setting. They observed that two nonadjacent nodes v_i and v_j in G can communicate with each other only when some intermediate v_z knows that it sits between v_i and v_j. The observation implies that, in the information theoretic setting, any MPC protocol must inherently leak information about

G to an adversary. Fortunately, they also derived a positive result: given some minimal amount of network information to leak, one can construct an MPC protocol which leaks no further information [10]. Leverage on such interesting possibility and Tal Moran's innovative idea, we caught the sight of NTRUEncrypt as an instrument to actualize our scheme.

As for NTRUEncrypt [11], it is a relatively new public key cryptosystem which is developed around 1996 by three mathematicians. Up till 2013, literatures can be found that introduce new parameters to resist currently known attacks and increase its computation power [12-14]. Since both encryption and decryption use only simple polynomial multiplications in NTRUEncrypt, it is more competitive to be used in resource-constraint applications such as mobile devices or smart-cards. And that is the reason we exploit it as our building block besides its semi-homomorphic property.

The rest of the paper is organized as follows. We first describe the system model and a naive broadcast protocol where topology-hiding is not achieved yet. Then we give some preliminaries about the modified NTRUEncrypt in section 4. In section 5, a topology-hiding MPC protocol, which is secure even if the adversary is capable of corrupt any part of the network graph, is presented. It is worth mentioning that our scheme dose not strictly comply with Tal Moran's [1] framework and deviates from it to some extents. Security analyses and performance evaluations are given in section 6. And we conclude the paper in section 7.

3. System model and topology-exposure problem

We model a network by an undirected graph $G=(V,E)$ which is not fully connected. For $v_i \in V$, its closed neighborhood is denoted as $N(v_i)=\{v_j \in V,(v_i,v_j)\in E\}\cup\{v_i\}$. A static and computationally bounded adversary that controls some subset of parties is considered in our model. The adversary is semi-honest and sees all the messages sent by the honest parties for each round.

In order to flood a broadcast to the whole network, we first describe a naive broadcast protocol for a message which can be represented as a bit string:

1. In the first round, the sender broadcast the message m to all of its neighbors, and every other party broadcasts a string of 0s to their own neighbors instead. Note that all these broadcasts are of equal lengths.

2. As for successive rounds, every party computes the bitwise OR of all broadcasts received from their neighbors, and sends the result back to them.

After k rounds, it is obvious that any node which is k hops or less from the sender will be sending the message, and after $diam(G)$ rounds all parties will agree on message m, where $diam(G)$ is the diameter of the graph.

However, though the broadcast functionality is realized in this naive protocol, the network topology would be exposed on it: every party can easily figure out its distance from the sender by counting the rounds until it receives a non-zero bit string, and even the direction where the sender lies in can be deduced by noting which neighbor first sent a non-zero message. In order to sort the topology-exposure problem out, we fall back on a semi-honest server which helps the neighboring nodes to securely compute the bitwise OR of the broadcasts and keep intermediate outcomes concealed. The topological structure is also confidential to the server in our scheme and more details can be found in section 5.

4. Preliminaries on NTRUEncrypt

In our protocol, the NTRU encryption algorithm which is based on the shortest vector problem in a lattice is exploited for semi-homomorphic computation. It works on a truncated polynomial ring $R = Z[X]/(X^N - 1)$ with convolution multiplication and all polynomials in the ring have integer coefficients and degree at most $N-1$:

$$a = a_0 + a_1 X^1 + a_2 X^2 + \cdots + a_{N-2} X^{N-2} + a_{N-1} X^{N-1}. \quad (1)$$

For each system, three integer parameters (N, p, q) are specified, where N represents the maximal degree $N-1$ for all polynomials in the truncated ring R, p and q are two moduli, respectively. It is always assumed that N is prime, q is much larger than p, and p is coprime to q. To send a secret message to the receiver, two key polynomials f and g with degree at most $N-1$ and with coefficients in $\{-1,0,1\}$ must be generated in advance. The polynomial f must also satisfy the additional requirement that the inverses modulo q and modulo p exist, which means that $f \cdot f_p = 1 \bmod p$ and $f \cdot f_q = 1 \bmod q$ must hold. If the chosen f is not invertible, we have to go back and try another f. After that, we preserve f and f_p as secret key, while compute $h = pf_q g$ as the public key.

In order to encrypt a message m, we represent it as a binary or ternary string and put it in the form of a polynomial whose coefficients belong to {-1,0,1}. Then we randomly choose a binding polynomial r with small coefficients and achieve the ciphertext as

$$c = r \cdot h + m (\bmod q). \quad (2)$$

During the decryption process, the receiver first multiplies the encrypted message c and part of his private key f. After she obtains

$$\begin{aligned} a &= f \cdot c (\bmod q) \\ &= p \cdot r \cdot g + f \cdot m, \end{aligned} \quad (3)$$

whose coefficients lie within the interval $[-q/2, q/2]$ since the polynomials r, g, f, m and prime p all have coefficients that are small compared to q, she can compute

$$\begin{aligned} b &= a (\bmod p) \\ &= f \cdot m (\bmod p) \end{aligned} \quad (4)$$

and

$$\begin{aligned} m &= f_p \cdot b (\bmod p) \\ &= f \cdot f_p \cdot m (\bmod p) \end{aligned} \quad (5)$$

to decrypt the ciphertext.

We note that if we constrain the polynomial coefficients of m in the interval $[-p/2, p/2]$ (instead of {-1,0,1}) with a sufficiently large modulus p, the plaintext can also be recovered properly. Thus we exploit this property and devise a semi-homomorphic scheme which can be used in our topology-hiding protocol.

Denote the coefficients of two bit strings m_1, m_2 as $(a_{1,1}, a_{1,2}, \cdots a_{1,N-2}, a_{1,N-1})$ and

$(a_{2,1}, a_{2,2}, \cdots a_{2,N-2}, a_{2,N-1})$ where $a_{1,i} + a_{2,i} \in [-p/2, p/2]$ for $0 \le i \le N-1$, the sum of their ciphertext can be written as

$$c_1 + c_2 = (r_1 + r_2) \cdot p \cdot f_q \cdot g + m_1 + m_2 (\mathrm{mod}\, q). \tag{6}$$

Then we decrypt the result as mentioned above, we obtain the added plaintext $m_1 + m_2 (\mathrm{mod}\, p)$ whose coefficients are $(a_{1,1} + a_{2,1}, a_{1,2} + a_{2,2}, \cdots a_{1,N-2} + a_{2,N-2}, a_{1,N-1} + a_{2,N-1})$.

5. Topology-hiding broadcast protocol

In this section we describe a protocol for topology-hiding broadcast against a semi-honest adversary. There are three objectives in designing our protocol: (1) the topology can not be revealed even if the adversary is capable of corrupting any part of the network graph; (2) despite the employment of a semi-honest third party, he can not derive any topology information from what he received; (3) even with prerequisite of hiding intermediate results of OR-operation, all nodes will be aware of the message at end. For clarity, we formulate our protocol in three phases.

Initialization phase: This phase aims at distributing secret shares among neighboring nodes and the server. First, we figure out a key-pair of NTRUEncrypt, assign the private key SK to the server while publish the public key PK to all parties. A positive integer d which is greater than the degree of any node is also allocated to all nodes as well as the server. Then for each node v_i, we assign it a randomly chosen polynomial k_i^1 whose degree is at most $N-1$ with coefficients belong to $\{-1,0,1\}$. Assuming that there exists a polynomial $k_i^{1\prime}$ where $\sum_{j \in N[v_i]} k_j^1 + k_i^{1\prime} (\mathrm{mod}\, p) = 0$ and $N[v_i]$ stands for the closed neighborhood of node v_i, we deliver $k_i^{1\prime}$ to v_i as well. It is worth noting that the polynomials k_i^1 and $k_i^{1\prime}$ denote the initial values of parameters k_i^t and $k_i^{t\prime}$ which will be updated in each step of Broadcast phase.

***Broadcast phase*:** In this phase, the aforementioned naive broadcast protocol is improved to a topology-hiding version. Depending on the depth of broadcast(only broadcast to a limited range, e.g. local information query) or network diameter $diam(G)$ (broadcast to the whole network), a series of rounds will be executed in the following manner.

For each round, we assume that the intermediate result of OR-operation as in naive broadcast protocol is represented as m_i for node v_i. It is worth noting that such intermediate result should be kept secret from any nodes or server except for the first and last rounds. Therefore, any node v_i is only endowed with an encrypted version $E_{PK}(m_i + k_i^t)$ as the input of current round t. As for the first round, we encode the original broadcast message as a polynomial $m_i = m_s$ whose coefficients belong to $\{0,1\}$ for sender v_s and assign polynomials $m_i = 0$ to any other nodes $v_i (i \ne s)$, thus all nodes can trivially figure out their inputs as $E_{PK}(m_i + k_i^1)$ by homomorphically combining $E_{PK}(m_i)$ with $E_{PK}(k_i^1)$.

At the beginning of each round t, every node v_i broadcasts its message $E_{PK}(m_i + k_i^t)$ to neighboring nodes and homomorphically sum its receives (including its own message) together with $E_{PK}(dk_i^{t+1})$ and $E_{PK}(k_i^{t'})$ as

$$\begin{aligned} & E_{PK}\left(\sum_{j\in N[v_i]} m_j + \sum_{j\in N[v_i]} k_j^t + dk_i^{t+1} + k_i^{t'})\right) \\ &= \sum_{j\in N[v_i]} E_{PK}(m_j + k_j^t) + E_{PK}(dk_i^{t+1}) + E_{PK}(k_i^{t'}), \end{aligned} \tag{7}$$

where all nodes exchange the coefficients of polynomial k_i^t in a same permutation to upgrade it to be k_i^{t+1}. Then it sends the result to the server who can recover the plaintext as

$$\sum_{j\in N[v_i]} m_j + dk_i^{t+1} = D_{Sk}\left(E_{PK}\left(\sum_{j\in N[v_i]} m_j + \sum_{j\in N[v_i]} k_j^t + dk_i^{t+1} + k_i^{t'})\right)\right), \tag{8}$$

since $\sum_{j\in N[v_i]} k_j^t + k_i^{t'} (\mathrm{mod}\ p) = 0$. We noticed that the coefficients of $\sum_{j\in N[v_i]} m_j$ are seated within $\{0,1,\cdots,\deg(v_i)+1\}$, where $\deg(v_i)$ denotes the degree of node v_i. So the OR-operation of corresponding coefficients for neighboring m_j can be calculated as $\left\lceil \sum_{j\in N[v_i]} m_j / d \right\rceil$ ($\lceil h \rceil$ stands for the ceiling operation of coefficients in polynomial h). However, after decoding the ciphertext, server can easily deduce the distance between sender and the current node, by discerning the existence of fractional part in $(\sum_{j\in N[v_i]} m_j + dk_i^{t+1})/d$. To sort this out, node v_i forge other n ciphertexts $E_{PK}\left(m_i^c + dk_i^c\right), c = \{1,2,\cdots,n\}$ applying some fake sums of intermediate results m_i^c and parameters k_i^c. Herein, the polynomials m_i^c are chosen as 0 with half probability, whereas for any other m_i^c, we uniformly choose a number from $\{1,2,\cdots,d\}$ and assign it to part of the polynomial's coefficients. k_i^c are fabricated by permute the coefficients of polynomials k_i^{t+1}. After that, we permute the counterfeits together with the genuine one and send the results to the server. Server then decrypts the results and reseals them as $E_{PK}\left(\left\lceil \left(m_i^c + dk_i^c\right)/d \right\rceil\right), c = \{0,1,\cdots,n\}$, in permuted order, where $E_{PK}\left(\left\lceil \left(m_i^0 + dk_i^0\right)/d \right\rceil\right) = E_{PK}\left(\left\lceil (\sum_{j\in N[v_i]} m_j + dk_i^{t+1})/d \right\rceil\right)$ stands for the genuine one. Since server is unaware of the permutation, he is incapable of deciding whether the broadcast reached current node in terms of the ceiling operation.

After received the resealed information from sever, node v_i pick up the genuine one as the

input of next round and we consider $\left\lceil \sum_{j\in N[v_i]} m_j / d \right\rceil$ in ciphertext as new m_i. We should also remember that p must be large enough so that any coefficient of $\sum_{j\in N[v_i]} m_j + dk_i^{t+1}$ resides within $[-p/2, p/2]$. In the end of current round, we also upgrade $k_i^{t'}$ by permuting its coefficients in the same way as updating k_i^t, making sure that $\sum_{j\in N[v_i]} k_j^{t+1} + k_i^{t+1'} (\text{mod } p) = 0$.

End phase: In order to make sure that any node is aware of the broadcasted message at end, we describe the final round as a separate phase. In the last round e, each node homomorphically sum its receives as

$$E_{PK}\left(\sum_{j\in N[v_i]} m_j + \sum_{j\in N[v_i]} k_j^e + k_j^{e'}\right) = \sum_{j\in N[v_i]} E_{PK}(m_j + k_j^e) + E_{PK}(k_j^{e'}), \quad (9)$$

where $t = e$ stands for the last round, without appending $E_{PK}(dk_i^{e+1})$ to it. Then it sends the ciphertext directly to server who decodes it as $\left\lceil \sum_{j\in N[v_i]} m_j / d \right\rceil$ and sends the result back. In terms of the received polynomial, node can trivially derive the original message from it.

The detailed protocol is shown in **Table 1.**

Table 1. Protocol at node v_i for each round t, where m_i stands for the current plaintext message and $m_i^c + dk_i^c = \sum_{j\in N[v_i]} m_j + dk_i^{t+1}$ for $c = 0$

Node v_i	**Server**
Secret shares: PK: public key d **:** an integer bigger than maximum node degree k_i^t : secret polynomial, upgraded as previously defined $k_i^{t'}$: secret polynomial, upgraded the same way as k_i^t	PK: public key SK: private key d **:** an integer bigger than maximum node degree
Protocol for each round t **:** (1) Broadcast $E_{PK}(m_i + k_i^t)$ to its neighbors (2) upon receiving broadcasts from its closed neighbors v_j, v_i upgrade k_i^t and computes $\sum_{j\in N[v_i]} E_{PK}(m_j + k_j^t) + E_{PK}(dk_i^{t+1}) + E_{PK}(k_i^{t'})$, (3) v_i permutes the result with counterfeits	

$E_{PK}\left(m_i^c + dk_i^c\right), c = \{1, 2, \cdots, n\}$ and submits them to the server.

(4) Decrypt the received messages, reseal them as $E_{PK}\left(\left\lceil \left(m_i^c + dk_i^c\right)/d \right\rceil\right), c = \{0, 1, \cdots, n\}$, and send them back in receiving order.

(5) pick up the genuine one as current intermediate result and upgrade $k_i^{t'}$.

In the last round $t = e$, v_i computes $\sum_{j \in N[v_i]} E_{PK}(m_j + k_j^e) + E_{PK}(k_i^{e'})$ instead and send it directly to the server, server decipher it as $\left\lceil \sum_{j \in N[v_i]} m_j / d \right\rceil$ and directly send it back.

6. Performance and Security analysis

As for the performance of our proposed protocol, we implement it with an agent-based modeling tool which is called Repast Symphony [15]. Considering that our scheme is an instantiation of topology-hiding broadcast for the first time, we also realize the protocol described in Section 4 of paper [1] by RSA cryptosystem and NTRUEncrypt (since it is the only similar protocol we know for topology-hiding broadcasting) for comparison. For more details, a thorough reading of [1] is highly recommended. In fact, though our protocol is inspired by Moran's seminal work [1], it differentiates from his framework such as secret shares and cryptography scheme. It is also worth mentioning that Moran simply ignored the issue on how to deal with the recovered key from neighboring key shares. Therefore, we assign the tasks of reconstructing the secret key as well as encryption-decryption operations of his protocol to a server, which implies that the topology will be explored to a third-party. In the simulation, nodes are connected to constitute a Watts-Strogatz network [16] for reality. Some parameters are chosen as in **Table 2.**

Table 2. Simulation parameters

parameter	**values**
Maximal degree of polynomials $N-1$	346
Modulus q	512
Modulus p	83
Number of counterfeit ciphertexts n	6
RSA key length	1024
Number of nodes	500

We implement the simulator on a PC machine equipped with Intel Core i5-2430M 2.40GHz processor and 3GB memories. The computation burden and communication overhead of server for each round per node is estimated in average of 50 tests.

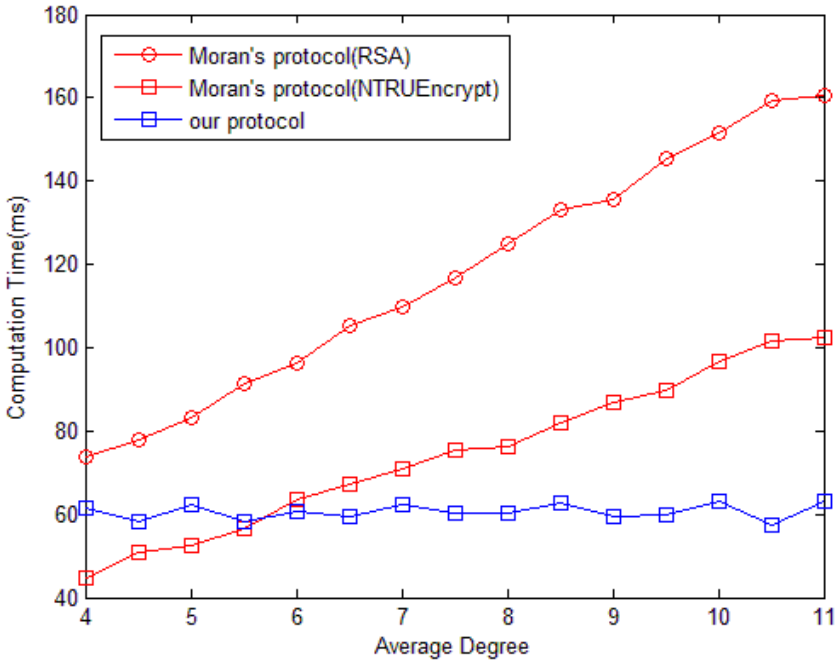

Fig. 1. Computation burden of server for each round per node

As shown in **Fig. 1**, it is obvious that our scheme is preferable than Moran's protocol especially when average node degrees are bigger. This is due to the reason that the decryption-encryption operations in Moran's protocol must be executed $\deg(v_i)+1$ (number of nodes in closing neighborhood of node v_i) times, while the execution times of NTRU algorithm in our scheme is constant as $n+1$. Even if we employed NTRUEncrypt to realize Moran's protocol for efficiency, the degradation of its performance is still conspicuous as average degree increases. Note that though homomorphic sum and counterfeits forging must be carried out in each node before sending in our scheme, it is unnecessary to worry about the extra burden since polynomial computations are efficient. When the scale of network is large, we can deploy a series of servers to distribute the computational burden to localized agents. As for communication overhead, each node firstly broadcasts its current intermediate result to neighbors in our scheme, which are $N\log_2 q$ bits, and the permuted polynomials subsequently submitted to server are $(n+1)N\log_2 q$ bits in all. The messages send back from server are the some amount as it received. Assuming that the key length of RSA or NTRUEncrypt are K bits (K is equal to 1024 for RSA and $N(2+\lceil \log_2 p \rceil)=3123$ for NTRUEncrypt in our experiment) in Moran's protocol, sever must retrieve at least $(\deg(v_i)+1)K$ bits to recovery the private key for node v_i, which is bitwise exclusive OR of secret shares. Meanwhile, the ciphertexts it sends and receives are both $(\deg(v_i)+1)K$ bits

when implemented by RSA and $(\deg(v_i)+1)N\lceil \log_2 q \rceil$ bits when implemented by NTRUEncrypt. **Fig. 2** illustrated that, due to longer ciphertexts (also longer key shares in NTRUEncrypt version of Moran's scheme), protocols based on NTRUEncrypt brought about more communication overhead. However, since the communication overhead only relates to the number of counterfeits in our protocol, it will not increase along with average degree as in Moran's scheme.

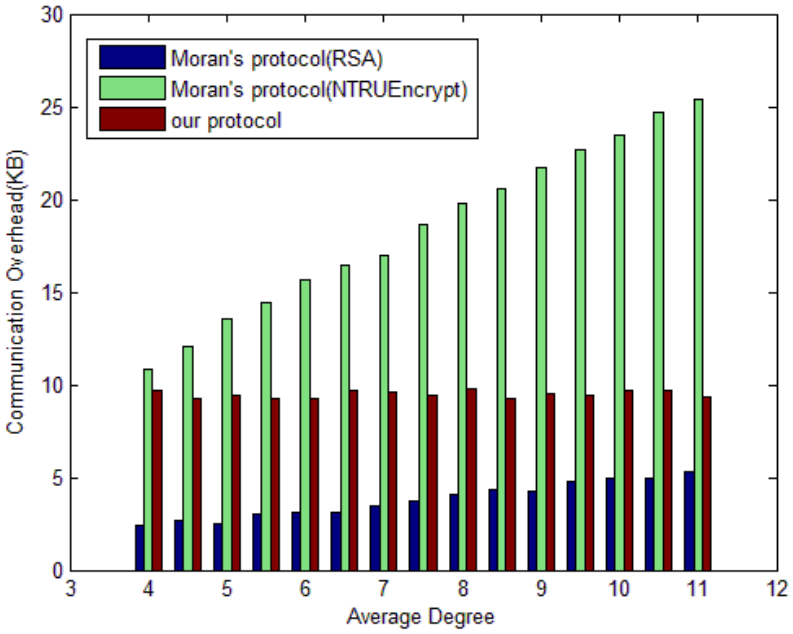

Fig. 2. Communication overhead of server for each round per node

There after, we give two theorems about the security of our scheme.

Theorem 1 Our proposed protocol is topology hiding as long as any part of the network is corrupted.

Proof. We leverage on the game-based security definition of topology-hiding proposed by Moran **[1]** for our proof.

Let G be a set of graphs. Let Π be our protocol running over any of the communication graphs in G where each node v_i gets an input $m_i \in \{0, m_s\}$. Herein, m_s stands for the input of broadcaster.

Assuming that an adversary A arbitrarily chooses a corrupt subset S, inputs m_j for the corrupted parties $v_j \in S$ and, for $k \in \{0,1\}$, two graphs $G_k = (V_k, E_k) \in G$, such that $S \subset V_0 \cap V_1$ and $N_{G_0}[S] = N_{G_1}[S]$. Then it outputs $(S; G_0, G_1; \{m_j\})$ to a competitor C. If $S \not\subset V_0 \cap V_1$ or if some input m_j is invalid C wins automatically.

Now C chooses a random $b \in \{0,1\}$ and runs Π in the communication graph G_b. A receives the collective view of all parties in S during the protocol execution.

Finally A must output $b' \in \{0,1\}$. If $b' = b$ we say that A wins the security game. Otherwise A loses.

According to the forementioned game rule, the game-based security definition of topology-hiding can be described as

Definition 1 We say that an MPC protocol Π is *Indistinguishable under Chosen Topology Attack (INDCTA secure)* over G if for any probabilistic polynomial time bounded adversary A there exists negligible function $\mu(\cdot)$, such that for every n it holds

$$\left|\Pr(A \text{ Wins}) - \frac{1}{2}\right| < \mu(n). \tag{10}$$

As for our protocol, the collective view of adversary A after gaming can be represented as a triad $\text{VIEW}_{G_b}^{\Pi} = (m_{j\in S}; E_{PK}(m_i + k_i^t), i \in N_{G_b}[S]; m_s)$, where $E_{PK}(m_i + k_i^t)$ stand for the encrypted intermediate messages computed by the server or received from the closed neighborhood of corrupted nodes. Since the adversary is unaware of the secret key and NTRUEncrypt is probabilistic, he can not reveal these ciphertexts or even tell which of them correspond to a same plaintext. That is to say, adversary A can not win the game by distinguishing $\text{VIEW}_{G_0}^{\Pi}$ from $\text{VIEW}_{G_1}^{\Pi}$ other than simply guessing and our scheme is topology-hiding in game-based definition.

Compared with the naive protocol described in section 3, our scheme prevents any corrupted nodes from deducing the distance or direction of broadcaster since they are ignorant of whether the intermediate messages are zero or not.

Theorem 2 The server is unaware of any information about the network topology.

Proof. The server receives a series of permuted intermediate results except for the last round. Since the genuine message is obscured with both zero and non-zero fake messages and server is unaware of the permutation order, it can not tell whether the genuine result is zero or not by discerning the existence of fractional part during ceiling operation. Moreover, server is also incapable of tracking the position of genuine messages in permutations since k_i^c is introduced and updated periodically. Therefore, server gains no more information than any node dose.

It is also worth mentioning that, the private key of any node is secretly shared within its neighbors in Moran's protocol, so the identities of neighbors can not be hidden from it and the network must be static. Meanwhile, Moran didn't address the problem of how to deal with the recovered secrets key, which means it can be reused by some party once figured out. However, the abovementioned defects do not exists in our scheme.

7. Conclusion and further work

The main contribution of this paper consists on the introduction of a specific topology-hiding broadcast protocol. We revised the naive scheme described in [1] and resort to the semi-homomorphic property of NTRUEncrypt as well as secret sharing to prevent any party from deducing topology information by intermediate results. By performance and security analysis, we illustrated the merits of our protocol compared with Moran's scheme. However, our protocol will be no longer secure if the server colludes with any node, thus we consider its solution as a further work. Moreover, we didn't take malicious models into account in our

scheme, and we leave the solutions of topology-hiding broadcast in malicious environments to the future.

References

[1] T. Moran, "Topology-hiding computation," in *Proc. of 12th Theory of Cryptography Conference*, pp. 23-25, 2015. Article (CrossRef Link)

[2] A. C. C. Yao, "Protocols for secure computations," in *Proc. of FOCS*, vol. 82, pp. 160–164, 1982. Article (CrossRef Link)

[3] O. Goldreich, S. Micali and A. Wigderson, "How to play ANY mental game," in *Proc. of the nineteenth annual ACM conference on Theory of computing*, pp. 218-229, 1987. Article (CrossRef Link)

[4] M. Ben-Or, S. Goldwasser and A. Wigderson, "Completeness theorems for noncryptographic fault-tolerant distributed computations," in *Proc. of 20th Annual ACM Symposium on Theory of Computing*, pp. 1–10, 1988. Article (CrossRef Link)

[5] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *Proc. of 42nd Annual Symposium on Foundations of Computer Science*, pp. 136–147, 2001. Article (CrossRef Link)

[6] M. Naor, B. Pinkas and R. Sumner, "Privacy preserving auctions and mechanism design," in *Proc. of 1st ACM Conf. on Electronic Commerce*, pp. 129–139, 1999. Article (CrossRef Link)

[7] R. Cramer, I. Damgard and Y. Ishai, "Share conversion, pseudorandom secret-sharing and applications to secure computation," in *Proc. of the Second Theory of Cryptography Conference*, pp. 342–362, 2005. Article (CrossRef Link)

[8] P. Bogetoft, K. Boye, H. Neergaard-Petersen and K. Nielsen, "Reallocating sugar beet contracts: Can sugar production survive in Denmark?" *European Review of Agricultural Economics*, vol. 34, no. 1, 2007. Article (CrossRef Link)

[9] S. Goldwasser, "Multi-party computations: Past and present," in *Proc. of 16th annual ACM Symposium on Principles of distributed computing*, pp. 21-24, 1997. Article (CrossRef Link)

[10] M. Hinkelmann, A. Jakoby, "Communications in unknown networks: Preserving the secret of topology," *Theor. Comput. Sci.*, vol 384, no. 2, pp. 184-200, 2007. Article (CrossRef Link)

[11] J. Hoffstein, J. Pipher and J. H. Silverman, "NTRU: A Ring Based Public Key Cryptosystem," *Lecture Notes in Computer Science*, vol. 1423, pp. 267-288, 1998. Article (CrossRef Link)

[12] J. Hermans, F. Vercauteren and B. Preneel, "Speed Records for NTRU," *Lecture Notes in Computer Science*, vol. 5985, pp. 73-88, 2010. Article (CrossRef Link)

[13] D. Stehle, R. Steinfeld, "Making NTRU as secure as worst-case problems over ideal lattices," in *Proc. of EUROCRYPT 2011. LNCS*, vol. 6632, pp. 27–47, 2011. Article (CrossRef Link)

[14] J. W. Bos, K. Lauter, J. Loftus and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *Proc. of IMACC 2013, LNCS*, vol. 8308, pp. 45–64, 2013. Article (CrossRef Link)

[15] M. J. North, E. Tatara, N. T. Collier and J. Ozik, "Visual Agent-based Model Development with Repast Simphony," in *Proc. of the Agent 2007 Conference on Complex Interaction and Social Emergence*, 2007. Article (CrossRef Link)

[16] D. J. Watts, S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440-442, 1998. Article (CrossRef Link)

Bo Mi received the P. H. degree in computer science in Chongqing University in 2009. His current research interests include multi-party computation, data privacy and secure vehicular ad hoc networks.

Dongyan Liu received the P. H. degree in civil engineering in Chongqing University in 1993. His current research interests include intelligent transportation and topology analysis.