Principles of Information Security

Indranil Pradhan

2019202008

MTech in Computer Science and Information Security

[R]

Given a digraph G (a directed network, that nodes A and B are part) of which up to any e nodes may be corrupted by an adversary, what is a necessary and sufficient condition on G such that Oblivious Transfer between A and B is possible, given the condition that initially, no node knows the public-key of any of the other nodes.

Answer:- Here it is given a digraph (a directed network, that nodes A and B are part) of which up to any e nodes may be corrupted by an adversary. It is needed to find out a necessary and sufficient condition on G such that Oblivious Transfer between A and B is possible, given the condition that initially no node knows the public key of any of other nodes.

Here is a n-party graph. n represents the number of nodes. This oblivious transfer let G allow e-secure computation of oblivious transfer correlation between all pairs of parties, if an f only if the complement of G does not contain the bipartite graph $K_{n-e,n-e}$ as a subgraph.

In the absence of practical real-world protocols for secure computation which are secure in the presence of any number of dishonest parties, there is a need for relaxations that are meaningful and yet provide significant performance benefits.

Oblivious transfer (OT) protocol between a sender and a receiver, by which the sender transfers some information to the receiver, the sender remaining oblivious, however, to what information the receiver actually obtains.

Here n parties are connected to each other with secure private communication point to point channel in synchronize manner between every pairs of parties. Some of the connections are oblivious transfer connection where it is possible perform any number of oblivious transfer operations. The parties which have established oblivious transfer between them are connected by an edge.

Now the question comes what about the parties which are not connected by oblivious transfer. The answer is they can use the existing oblivious transfer by relying on the symmetric key cryptography. So in that case no need to add additional oblivious transfer.

Here assuming the full network of secure channels and this secure channel is achieved by hybrid encryption technique. Here we are assuming of full network is because this one time set up cost is lower than the on time set up cost of oblivious transfer cost of unbounded time using oblivious extension.

A complete graph is necessary for secure computation to deal an adversary that can corrupt e = n-1 parties. And the negative result obtains when complement of the graph contains $K_{n-e,n-e}$ as subgraph.

Let G = (V, E) be an OT graph on n parties P1, . . . Pn, so that any pair of parties Pi , Pj which are connected by an edge may make an unbounded number of calls to an oblivious transfer oracle.

The set of adversaries let A can corrupt e parties. The two parties P and Q among (P_1,P_2,..P_n) can imitate oblivious transfer oracle securely for three conditions stated as below.

1. e< n/2 in honest majorities.

or

2. P and Q are connected by an edge in G. This is a trivial case.

Or

3. let there be partitions Part1, Part2, Part3 of G. It is oblivious transfer possible if Part1, Part2, Part3 does not follow below three conditions.

(a) |Part1| = |Part2| = n − e and |Part3| = 2e − n;

(b) P belongs to Part1 and Q belongs to Part2; and

(c) for every P' belongs to Part1 and Q' belongs to Part2 and it holds that (P', Q') does not belong to E.

when e < n/2 that means we are in honest majorities authorities we can securely imitate oblivious transfer using honest majority information theoretically secure protocols.

For the second condition it is possible to imitate oblivious transfer between P and Q as there is an edge.

The third condition applies when P and Q is not connected by oblivious transfer edge and e >= n/2 and if there is $K_{n-e,n-e}$ bipartite graph as subgraph in the complement of G, it is not possible to tolerate e semi honest corruptions. And it applies that when there is no $K_{n-e,n-e}$ bipartite graph in the complement of G, then it is possible to tolerate e semi honest corruptions.

[Q]

Recall that in Evaluation II, k blocks of data/information is encoded into n blocks such that if any e of the n blocks are corrupted, it is still possible to retrieve the original k blocks of information. Show how to use this to build a robust (torrent like) routing scheme where the sender and the receiver have n different connections/routes and the task is to send k blocks of data successfully even if up to any e of the n connections are corrupt. Further, using a public-key cryptosystem, say El Gamal, design a Robust Oblivious Transfer protocol between a client A (who has the index i) and server B (who has the array) such that A and B are part of a large network and reliably communicate via the above robust (torrent) routing mechanism (you may have to design four different protocols in the four cases outlined below). What do you think would be the maximum tolerable e when (a) the public-keys of A and B are known to all, (b) public-key of A is known to B, but B's public-key is not known to A, (c) public-key of B is known to A, but A's public-key is not known to B and (d) neither party knows the public-key of the other party

Answer:-

Design of the Robust Routing Scheme:-

In Evaluation 2 we have seen that k blocks of data is encoded into n blocks, and if there is e corrupted blocks, still it is possible to retrieve original k blocks of data. Here client and server is connected through n different connections and the task is to send k blocks of data successfully even if up to any e of the n connections are corrupted.

Here the assumptions are

1. n = number of different connections between client and server.
2. e = number of corrupted connections.
3. From evaluation 2 we have seen that it is possible to retrieve k blocks of data if k <= (n-e).

And the following steps are the transmission steps.

First, Let the we divide the data into k blocks data = ($data_{1,}$ $data_2$, ….. $data_n$). Now the steps are followed according to the steps of evaluation 2.

Second, Polynomial is created such that the degree is of the polynomial is k with the k blocks as coefficients of the polynomial (f(x)).

Third, The data blocks are encoded into n blocks, i.e. – (en_1,en_2,….,en_n) where every en_i is {x, f(x), sign(h(f(x))}. Polynomial is signed. h(.) represents the hash function. X represents unique identifier of en_i, which has been chosen randomly from the range of 1 to n.

Fourth, in the final step, in the receiver end the data block is reconstructed using the k or more points.

Design of the Oblivious Transfer Protocol:-

As in the above part the robust routing scheme is designed. It is now possible to assume that I have a robust routing scheme using n connections and it possible to transfer k blocks of the data even if e connections are corrupted.

Assumptions for Oblivious Transfer Protocol are as follows-

Here ELGamal encryption scheme is used. Let B holds the array b=(b_1,b_2,b_3,….,b_k). And A has index i which belongs to the range (0,k-1) and A wants to know about $b_{i.}$

The protocol works as follows-

A first sends a random array a = (a_1,a_2,a_3,…,a_k) where every r_i is encryption of r by the public of B. and given the condition that $r_{i!=j}$ and r is random number.

B decrypts the entire array a, which is sent by A. B then again creates a new array for A such that new_b = [D(r_j) xor b_j] where j belongs to the range (1,k) and sends new_b to A.

A now can obtain b_i by doing (new_b[i] xor i).

So here the index, i which is used by A, B doesn't get to know about it and A also receives its desired b_i but no other $b_{i!=j}$. This is the design of the oblivious transfer protocol.

ELGamal Encryption Decryption Algorithm:-

Suppose A wants to communicate to B.

1. B generates public and private key:
    - B chooses a very large number **q** and a cyclic group $\mathbf{Z}_q$.
    - From the cyclic group $\mathbf{Z}_q$, he choose any element **g** and an element **a** such that gcd(**a**, **q**) = 1.
    - Then he computes **h** = $\mathbf{g}^a$.
    - B publishes **Z**, **h = $\mathbf{g}^a$**, **q** and **g** as his public key and retains **a** as private key.
2. A encrypts data using B's public key:
    - A selects an element **k** from cyclic group **Z** such that gcd(**k**, **q**) = 1.
    - Then she computes **p** = $\mathbf{g}^k$ and **s** = $\mathbf{h}^k$ = $\mathbf{g}^{ak.}$
    - A multiples **s** with **M**.
    - Then A sends (**p**, **M*s**) = ($\mathbf{g}^k$, **M*s**).
3. B decrypts the message:
    - B calculates **s'** = $\mathbf{p}^a$ = $\mathbf{g}^{ak}$.
    - B divides **M*s** by **s'** to obtain **M** as **s = s'.**

Different Scenarios -

a. **the public-keys of A and B are known to all:-**
Here all the public keys of A and B is known to all. As B's keys is known to all so it is possible of encryption using ElGamal encryption scheme and it is also possible for oblivious transfer.
The maximum tolerable e as follows:
e <= (n- k)

b. **Public-key of A is known to B, but B's public-key is not known to A:-**
Here public key of A is known to B but B's public key is known to A. So the problem arises during encryption. It is not possible for A to encrypt without the prior knowledge of B's public key as per the Elgamal encryption scheme as well as it is not possible for oblivious transfer. So, it is needed first to transfer the B's public key A so that A can encrypt. Let the public key of B takes b blocks.
The maximum tolerable e as follows:
e <= min((n-b),(n-k))

c. **Public-key of B is known to A, but A's public-key is not known to B:-**
Here public key of B is known to A but A's public key is not known to B. As B's public key is known to A, then A can proceed with encryption using ElGamal encryption scheme and it is also possible for oblivious transfer.
The maximum tolerable e as follows:
e <= (n-k)

d. **Neither party knows the public-key of the other party:-**
Here neither A nor B knows about public key of the other party. So here it is not possible for A to encrypt without prior knowledge of B's public key using ElGamal encryption as well as it is not possible for oblivious transfer. So it is needed first to transfer B's public key to A so that A can encrypt. Let the size of B's public key is b blocks.
The maximum tolerable e as follows:
e <= min((n-b),(n-k))