Efficient 2-Round General Perfectly Secure Message Transmission: A Minor Correction to Yang and Desmedt's Protocol*

Qiushi Yang and Yvo Desmedt

Department of Computer Science, University College London, UK
{q.yang, y.desmedt}@cs.ucl.ac.uk

Abstract. At Asiacrypt '10, Yang and Desmedt proposed a number of perfectly secure message transmission protocols in the general adversary model. However, there is a minor flaw in the 2-round protocol in an undirected graph to transmit multiple messages. A small correction solves the problem. Here we fix the protocol and prove its security.

1 Brief Introduction

The aim of perfectly secure message transmission (PSMT) is to transmit messages from a sender S to a receiver R in a network graph with *perfect privacy* and *perfect reliability*. Suppose a *Byzantine adversary* exists in the network, perfect privacy means that the adversary learns no information about the message, and perfect reliability means that the receiver R can output the messages correctly.

We consider the general adversary model, in which the adversary is characterized by an *adversary structure* $\mathcal{A}$ [1]. Our protocol uses the following techniques: *linear code*, *pseudo-basis and pseudo-dimension* and *randomness extractor*. Since the goal of this paper is to fix a small part of Yang and Desmedt's Asiacrypt paper, we refer to [2] for the other details, such as the model, the settings, etc.

2 Old 2-Round Undirected Protocol

Here we copy the 2-round undirected protocol for multiple message transmission in an undirected network graph [2, pp. 460].

2-round undirected protocol for $\ell = wt^{\mathcal{A}}(n - sz^{\mathcal{A}} - 1)$ messages $s_1, \ldots, s_\ell$

Round 1 - R to S:

1. R chooses $wt^{\mathcal{A}}n$ random k-vectors $\mathbf{r}_1, \ldots, \mathbf{r}_{wt^{\mathcal{A}}n} \in \mathbb{F}^k$, and for each $1 \le i \le wt^{\mathcal{A}}n$, S encodes $\mathbf{r}_i$ to get codeword $\mathbf{c}_i = EC(\mathbf{r}_i) = (c_{i1}, \ldots, c_{ih})$.
2. For each $1 \le i \le n$, R sends vectors $\mathbf{r}_{i+0\cdot wt^{\mathcal{A}}}, \mathbf{r}_{i+1\cdot wt^{\mathcal{A}}}, \ldots, \mathbf{r}_{i+(wt^{\mathcal{A}}-1)wt^{\mathcal{A}}}$ via path w_i. R also sends codewords $\mathbf{c}_1, \ldots, \mathbf{c}_{wt^{\mathcal{A}}n}$ via W with respect to ψ.

Round 2 - S to R:

1. S receives $wt^{\mathcal{A}}$ k-vectors $\mathbf{r}'_{i+0\cdot wt^{\mathcal{A}}}, \mathbf{r}'_{i+1\cdot wt^{\mathcal{A}}}, \ldots, \mathbf{r}'_{i+(wt^{\mathcal{A}}-1)wt^{\mathcal{A}}}$ on each path w_i ($1 \le i \le n$), and also receives $wt^{\mathcal{A}}n$ h-vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{wt^{\mathcal{A}}n}$ from W. For each $1 \le i \le wt^{\mathcal{A}}n$, let $\mathbf{x}_i = (x_{i1}, \ldots, x_{ih})$.
2. For each $1 \le i \le wt^{\mathcal{A}}n$, S uses the pseudo-basis construction scheme to construct a pseudo-basis B from $\mathbf{x}_1, \ldots, \mathbf{x}_{wt^{\mathcal{A}}n}$. Let b be the pseudo-dimension of B, then $b \le wt^{\mathcal{A}}$.

* This result was originally going to appear in the full version of [2]. However, as required by some recent studies of this model, we show this correction on Cryptology ePrint Archive in advance.

3. For each $1 \le i \le wt^{\mathcal{A}}n$, S encodes $\mathbf{r}'_i$ to get codeword $\mathbf{c}'_i = EC(\mathbf{r}'_i) = (c'_{i1}, \ldots, c'_{ih})$. S then constructs a set D_i such that for each $1 \le j \le h$, iff $x_{ij} \neq c'_{ij}$, then $(c'_{ij}, j) \in D_i$.
4. For each $1 \le i \le wt^{\mathcal{A}}n$, S decodes $r'_i = DC(\mathbf{r}'_i)$. S then constructs a set T such that iff $|D_i| \le wt^{\mathcal{A}}$, then $r'_i \in T$. S uses the randomness extractor to get $(z_1, \ldots, z_\ell) = RE(T)$, and for each $1 \le i \le \ell$, S computes $\sigma_i = s_i + z_i$.
5. S broadcasts the pseudo-basis B and $\sigma_1, \ldots, \sigma_\ell$. For each $1 \le i \le wt^{\mathcal{A}}n$, if $|D_i| > wt^{\mathcal{A}}$, then S broadcasts "ignore i"; else, then S broadcasts D_i.

Recovery Phase
1. R finds the final error locator F from B.
2. For each D_i that R receives on W, R constructs an h-vector $\mathbf{c}''_i = (c''_{i1}, \ldots, c''_{ih})$ such that for each $1 \le j \le h$, if $(c'_{ij}, j) \in D_i$, then $c''_{ij} = c'_{ij}$; else, then $c''_{ij} = c_{ij}$. R then decodes the information r''_i of $\mathbf{c}''_i$ such that for any $j \in F$, c''_{ij} is not used for decoding. R puts r''_i in a set T'.
3. R uses the randomness extractor to get $(z'_1, \ldots, z'_\ell) = RE(T')$, and for each $1 \le i \le \ell$, R computes $s'_i = \sigma_i - z'_i$. **End.**

The original design of this protocol is to enable $c''_{ij} = c'_{ij}$ for each $j \notin F$ $(1 \le j \le h)$ in the Recovery Phase. However, due to the existence of the *invalid error vector* [2], it is possible that $c'_{ij} \neq c_{ij}$ for some $j \notin F$ and $(c'_{ij}, j) \notin D_i$. In this case $c''_{ij} = c_{ij} \neq c'_{ij}$. This may make the decoding unreliable. A minor correction can solve this problem, thus we fix this protocol in the next section.

3 Fixed 2-Round Undirected Protocol

Here we give a fixed PSMT protocol which guarantees that $T' = T$, and hence the protocol is perfectly reliable. The protocol is almost the same as the original one. The *only* modifications are in Step 3 of Round 2 and Step 2 of the Recovery Phase. We emphasize the modifications using ***bold*** font and footnotes.

Fixed 2-round undirected protocol for $\ell = wt^{\mathcal{A}}(n - sz^{\mathcal{A}} - 1)$ messages $s_1, \ldots, s_\ell$

Round 1 - R to S:
1. R chooses $wt^{\mathcal{A}}n$ random k-vectors $\mathbf{r}_1, \ldots, \mathbf{r}_{wt^{\mathcal{A}}n} \in \mathbb{F}^k$, and for each $1 \le i \le wt^{\mathcal{A}}n$, S encodes $\mathbf{r}_i$ to get codeword $\mathbf{c}_i = EC(\mathbf{r}_i) = (c_{i1}, \ldots, c_{ih})$.
2. For each $1 \le i \le n$, R sends vectors $\mathbf{r}_{i+0\cdot wt^{\mathcal{A}}}, \mathbf{r}_{i+1\cdot wt^{\mathcal{A}}}, \ldots, \mathbf{r}_{i+(wt^{\mathcal{A}}-1)wt^{\mathcal{A}}}$ via path w_i. R also sends codewords $\mathbf{c}_1, \ldots, \mathbf{c}_{wt^{\mathcal{A}}n}$ via W with respect to ψ.

Round 2 - S to R:
1. S receives $wt^{\mathcal{A}}$ k-vectors $\mathbf{r}'_{i+0\cdot wt^{\mathcal{A}}}, \mathbf{r}'_{i+1\cdot wt^{\mathcal{A}}}, \ldots, \mathbf{r}'_{i+(wt^{\mathcal{A}}-1)wt^{\mathcal{A}}}$ on each path w_i $(1 \le i \le n)$, and also receives $wt^{\mathcal{A}}n$ h-vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{wt^{\mathcal{A}}n}$ from W. For each $1 \le i \le wt^{\mathcal{A}}n$, let $\mathbf{x}_i = (x_{i1}, \ldots, x_{ih})$.
2. For each $1 \le i \le wt^{\mathcal{A}}n$, S uses the pseudo-basis construction scheme to construct a pseudo-basis B from $\mathbf{x}_1, \ldots, \mathbf{x}_{wt^{\mathcal{A}}n}$. Let b be the pseudo-dimension of B, then $b \le wt^{\mathcal{A}}$.
3. For each $1 \le i \le wt^{\mathcal{A}}n$, S encodes $\mathbf{r}'_i$ to get codeword $\mathbf{c}'_i = EC(\mathbf{r}'_i) = (c'_{i1}, \ldots, c'_{ih})$. S then constructs a set D_i such that for each $1 \le j \le h$, iff $x_{ij} \neq c'_{ij}$, ***then*** $(c'_{ij}, x_{ij}, j) \in D_i$.[1]
4. For each $1 \le i \le wt^{\mathcal{A}}n$, S decodes $r'_i = DC(\mathbf{r}'_i)$. S then constructs an ordered set T such that iff $|D_i| \le wt^{\mathcal{A}}$, then $r'_i \in T$. S uses the randomness extractor to get $(z_1, \ldots, z_\ell) = RE(T)$, and for each $1 \le i \le \ell$, S computes $\sigma_i = s_i + z_i$.

[1] The only difference is that each tuple $(c'_{ij}, x_{ij}, j) \in D_i$ has 3 elements now. In the old protocol the entry x_{ij} was not involved. A careful re-reading shows that a pair, i.e., $((c'_{ij} - x_{ij}), j)$, can also be used, but here we use the 3-tuple for a simpler presentation.

5. S broadcasts the pseudo-basis B and $\sigma_1, \ldots, \sigma_\ell$. For each $1 \le i \le wt^{\mathcal{A}}n$, if $|D_i| > wt^{\mathcal{A}}$, then S broadcasts "ignore i"; else, then S broadcasts D_i.

Recovery Phase

1. R finds the final error locator F from B.
2. For each D_i that R receives on W, R constructs an h-vector $\mathbf{c}''_i = (c''_{i1}, \ldots, c''_{ih})$ such that for each $1 \le j \le h$, ***if*** $(c'_{ij}, x_{ij}, j) \in D_i$,[1] ***then*** $c''_{ij} = c'_{ij} - (x_{ij} - c_{ij})$;[2] else $c''_{ij} = c_{ij}$.[3] R then decodes the information r''_i of $\mathbf{c}''_i$ such that for any $j \in F$, c''_{ij} is not used for decoding. R puts r''_i in a set T'.
3. R uses the randomness extractor to get $(z'_1, \ldots, z'_\ell) = RE(T')$, and for each $1 \le i \le \ell$, R computes $s'_i = \sigma_i - z'_i$. **End.**

Theorem 1 *The fixed 2-round undirected protocol is a PSMT protocol for multiple messages.*

Proof. Without loss of generality, we assume that the adversary corrupts the set of paths $\{w_1, \ldots, w_t\} \in \mathcal{A}$; i.e., $t \le sz^{\mathcal{A}}$.

First we prove that the protocol is perfectly private. In Round 1, the adversary can learn $wt^{\mathcal{A}}t$ random k-vectors:

$$\mathbf{r}'_{i+0\cdot wt^{\mathcal{A}}}, \mathbf{r}'_{i+1\cdot wt^{\mathcal{A}}}, \ldots, \mathbf{r}'_{i+(wt^{\mathcal{A}}-1)wt^{\mathcal{A}}}$$

for $1 \le i \le t$. With the pseudo-basis B broadcast in Round 2, the adversary can learn (at most) extra b codewords, and hence extra b random k-vectors. Now if a pair $(c'_{ij}, x_{ij}, j) \in D_i$, then either $\mathbf{r}'_i$ or x_{ij} is corrupted, or both are corrupted. Either way, the adversary knows c'_{ij} already before the broadcast in Round 2. That is, the broadcast in Round 2 does not reveal any extra information. Thus in total, the adversary can learn at most $wt^{\mathcal{A}}t + b$ $(\le wt^{\mathcal{A}}(sz^{\mathcal{A}}+1))$ random k-vectors that R has chosen in Round 1. Since $wt^{\mathcal{A}}n - (wt^{\mathcal{A}}t + b) \ge wt^{\mathcal{A}}(n - sz^{\mathcal{A}} - 1) = \ell$, there are at least ℓ k-vectors that remain secret. For any k-vector $\mathbf{r}_i$ that remains secret, it is straightforward that $|D_i| \le wt^{\mathcal{A}}$, and hence $r'_i \in T$ and r'_i is secret to the adversary. Thus the adversary has no knowledge on at least ℓ elements in T. We can then use the randomness extractor to get ℓ perfectly private randomnesses. That is, there are enough number of secret pads $z_1, \ldots, z_\ell$ to be used to encrypt the messages, thus the protocol is perfectly private.

Next we prove that the protocol is perfectly reliable. First, we show that for each D_i that R receives, R gets $r''_i = r'_i$. First, for each $1 \le i \le wt^{\mathcal{A}}$, we have $\mathbf{x}_i = \mathbf{c}_i + \mathbf{e}_i$ where $\mathbf{e}_i$ is an error vector. From Theorem 2 of [2], we know that the information of $\mathbf{c}_i$ can be decoded from $\mathbf{x}_i$ if the final error locator F is given. Let $\mathbf{e}_i = (e_{i1}, \ldots, e_{ih})$, for each $1 \le j \le h$, we have $x_{ij} = c_{ij} + e_{ij}$. Now in the Recovery Phase, if $(c'_{ij}, x_{ij}, j) \in D_i$, then $c''_{ij} = c'_{ij} - (x_{ij} - c_{ij}) = c'_{ij} - e_{ij}$; else (which means $x_{ij} = c'_{ij}$), $c''_{ij} = c_{ij} = x_{ij} - e_{ij} = c'_{ij} - e_{ij}$. Thus in either case, for each $1 \le j \le h$, we have $c''_{ij} = c'_{ij} - e_{ij}$, and hence $\mathbf{c}''_i = \mathbf{c}'_i - \mathbf{e}_i$. Therefore, as we showed above, if the final error locator F is given, then the information of $\mathbf{c}'_i$ can be decoded from $\mathbf{c}''_i$. Thus R can get $r''_i = r'_i$ for each D_i received, and simultaneously get $(z'_1, \ldots, z'_\ell) = (z_1, \ldots, z_\ell)$ to recover the messages with perfect reliability. □

Since we only changed the number of elements from 2 to 3 in each vector of each D_i, the transmission complexity (TC) of the protocol remains $O(hn\ell)$ as shown in [2].

References

1. M. Hirt and U. M. Maurer. Player simulation and general adversary structures in perfect multiparty computation. *J. Cryptology*, 13(1):31–60, 2000.
2. Q. Yang and Y. Desmedt. General perfectly secure message transmission using linear codes. In *Proc. Asiacrypt '10*, volume 6477 of *LNCS*, pages 448–465, 2010.

[2] The only difference is that if $(c'_{ij}, x_{ij}, j) \in D_i$, then the fixed protocol computes $c''_{ij} = c'_{ij} - (x_{ij} - c_{ij})$ instead of $c''_{ij} = c'_{ij}$.

[3] Note that c'' is not a codeword. Instead, it is a corrupted decoding-end-vector, but correct information can be decoded from it.