Simple and Communication Complexity Efficient Almost Secure and Perfectly Secure Message Transmission Schemes

Yvo Desmedt[1,2] Stelios Erotokritou[1] Reihaneh Safavi-Naini[3]

[1] Department of Computer Science
University College London, UK

[2] Research Center for Information Security (RCIS)
AIST, Japan

[3] Department of Computer Science
University of Calgary, Canada

May 4, 2010

OVERVIEW

1. Introduction
2. A campaign for better notations
3. A first 1-phase $(0, 0, \gamma)$-secure protocol
4. Old protocols in new “barrels”
5. Efficient Perfectly Secure Message Transmission
6. Conclusions

1. INTRODUCTION

This talk is in the intersection of network security and cryptography.

After WWI, designers of networks wanted to guarantee reliability of a network against an attacker that destroys t nodes.

The problem was then generalized to the case nodes, deny or forward incorrect information (see Hadzilacos 1984 and Dolev 1982).

The issue became important to cryptography when the privacy requirement was added (see Dolev-Dwork-Waarts-Yung, 1993).

Since then lots of papers in the area (see survey paper by Desmedt, BT Tech. Journal, 2006) have appeared. There are

several more recent papers, e.g., by Kurosawa-Suzuki (ICITS 2007) and Kurosawa-Suzuki (Eurocrypt 2008).

Kurosawa-Suzuki (Eurocrypt 2008) have perfect reliability and perfect privacy with optimal (order wise) transmission complexity.

Some definitions:

Communication Complexity: number of bits the sender sends to communicate 1 bit plaintext.

Transmission Complexity: number of bits sender sends divided by the length of the message.

One can wonder which of these two measures is the most important.

Google Search gives:

- Communication Complexity: 93,000 hits

Google Search gives:

- Communication Complexity: 93,000 hits
- Transmission Complexity: 1,560 hits

Google Search gives:

- Communication Complexity: 93,000 hits
- Transmission Complexity: 1,560 hits

But what about Google Scholar?

Google Search gives:

- Communication Complexity: 93,000 hits
- Transmission Complexity: 1,560 hits

But what about Google Scholar?

- Communication Complexity: 13,400 hits!
- Transmission Complexity: 190 hits

Google Search gives:

- Communication Complexity: 93,000 hits
- Transmission Complexity: 1,560 hits

But what about Google Scholar?

- Communication Complexity: 13,400 hits!
- Transmission Complexity: 190 hits

Why we agree with the majority:

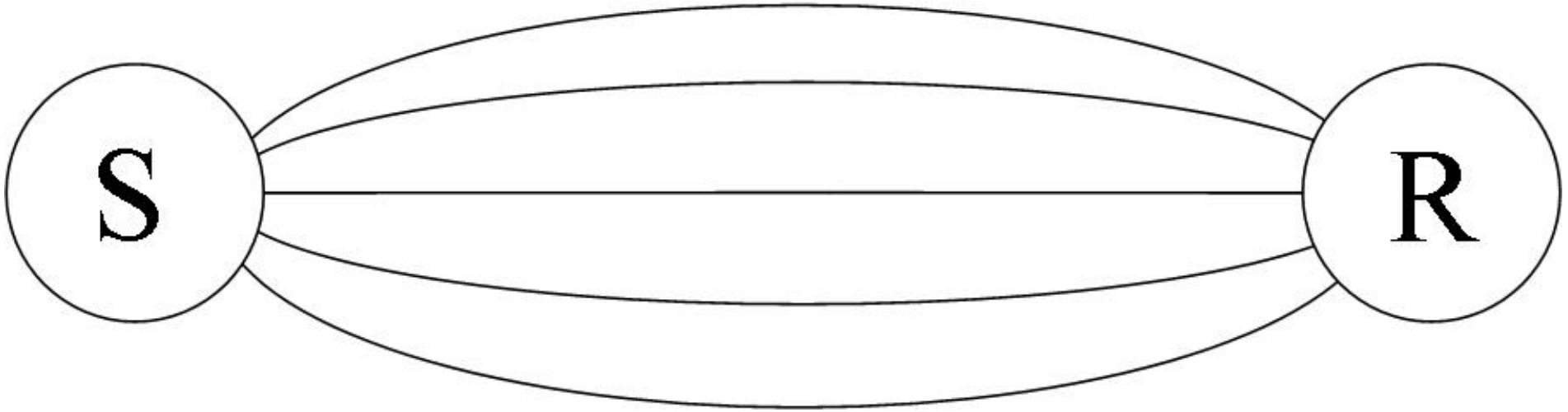
Why we agree with the majority:

- Perfectly Secure Message Transmission protocols are expensive. They need a transmission complexity of at least $2t + 1$. So, they will only be used in exceptional circumstances, such as if most public key systems would be broken. So, the message sent will likely be short as sending a new key for a conventional cryptographic scheme. Afterwards, one switches to classical cryptography.

- Even if one would assume Perfectly Secure Message Transmission (and its variants) be used in practice, the bound is meaningless in practice. Indeed, to achieve this rate, messages are made artificially long. However, in many applications, as ssh, packages are short!

So, we are the first to focus on communication complexity.

Note: we use standard techniques as: secret sharing, interaction and vertex disjoint paths, being:



2. A CAMPAIGN FOR BETTER NOTATIONS

The classical notation is from Franklin and Wright and defines (ϵ, δ)-security, as:

1. Let $\delta < \frac{1}{2}$. A message transmission protocol is δ-reliable if, with probability at least 1 - δ, B terminates with $M^B = M^A$.

2. ϵ refers to the privacy that is achieved, see Franklin-Wright.

A protocol is (ϵ, δ)-secure if it is ϵ-private and δ-reliable. A message transmission protocol is perfectly reliable if it is 0-reliable (similar for privacy).

Note: strange notation, since, e.g., 0-reliable means no errors!

However, standard!

Kurosawa-Suzuki introduced almost secure, meaning:

A (1-phase, n-channel) message transmission scheme is (t, δ)-secure if the following conditions are satisfied

Privacy: The adversary learns no information on M^A (better than guessing).

General Reliability: The receiver outputs $M^B = M^A$ or $\perp$ (failure). The receiver thus never outputs a wrong secret.

Failure: $Pr(\text{Receiver outputs } \perp) < \delta$.

The two definitions cannot be compared!

So, we campaign to use $(\epsilon, \delta, \gamma)$-security, where

γ-availability: when with probability at least $1 - \gamma$, B accepts a message, i.e. B rejects with probability γ.

δ-authenticity:

$$\delta = P(M^A \neq M^B | ReceiverAccepts).$$

ϵ-privacy: as defined by Franklin-Wright.

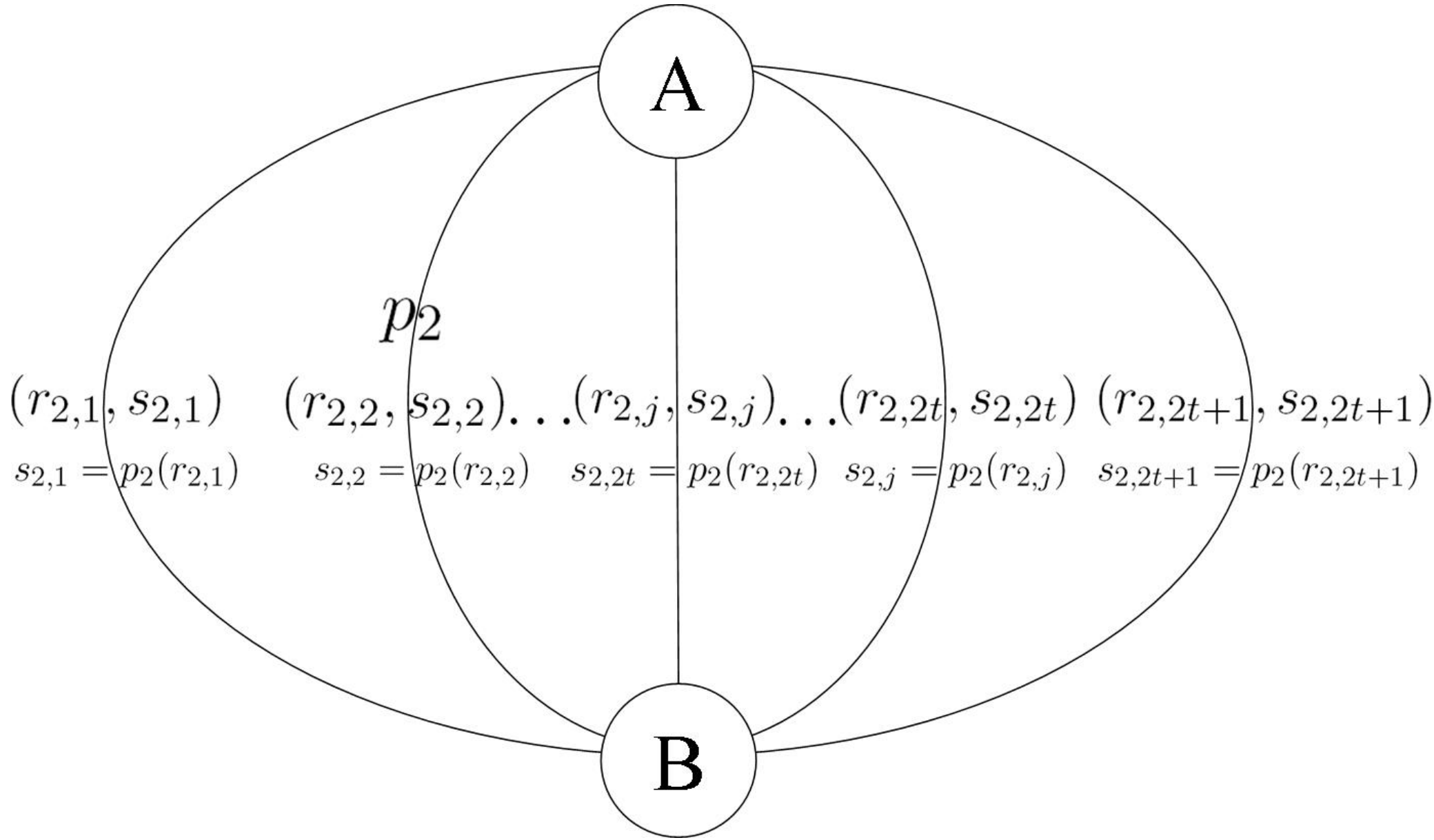
3. A FIRST 1-PHASE $(0, 0, \gamma)$-SECURE PROTOCOL

Denote M^A the secret message A wants to transmit. Let $n = 2t + 1$.

Step 1 The sender chooses shares $(s_1, \ldots, s_n)$ of M^A from a Shamir's $(t+1)$-out-of-n secret sharing scheme.

Step 2 For each s_i, the sender chooses a random polynomial p_i such that $p_i(0) = s_i$ (degree at most t) and random $r_{i,j}$.

Step 3 The sender transmits (e.g., for $i = 2$), as following


A
p_2
$(r_{2,1}, s_{2,1})$ $(r_{2,2}, s_{2,2}) \ldots (r_{2,j}, s_{2,j}) \ldots (r_{2,2t}, s_{2,2t})$ $(r_{2,2t+1}, s_{2,2t+1})$
$s_{2,1} = p_2(r_{2,1})$ $s_{2,2} = p_2(r_{2,2})$ $s_{2,2t} = p_2(r_{2,2t})$ $s_{2,j} = p_2(r_{2,j})$ $s_{2,2t+1} = p_2(r_{2,2t+1})$
B

The receiver executes the following:

Step 1 For all i: B checks the number of times $p_i^B(r_{i,j}^B) = s_{i,j}^B$ $(1 \leq j \leq n)$. If only t times or less, wire i is FAULTY.

Step 2 For all non-FAULTY wires i: B computes $p_i^B(0)$.

Step 3 B checks whether there exists a polynomial p^B of degree at most t such that for all non-FAULTY i: $p^B(x_i) = p_i^B(0)$, where x_i is public and comes from Shamir's secret sharing.
If so, then accept $M^B = p^B(0)$, else reject.

Theorem 1. *This protocol achieves* $(0, 0, \gamma)$ *security for* $q \geq ct(t+1)$ *when* t *tends towards infinity and* c *an appropriate constant (in function of* γ*).*

Proof: Privacy: trivial.

Authenticity: $t+1$ wires are honest, and so their wires will not be declared non-faulty and so $s_i^A = p_i^B(0)$. If for some i', not declared faulty, $s_{i'}^A \neq p_{i'}^B(0)$, then B will reject.

Availability: Observe that a wire B declared non-FAULTY might be dishonest, when the adversary is very lucky. The adversary could modify:

- $p_i(x)$ into $p'_i(x)$, and
- $r_{i,j}$ and $p_i(r_{i,j})$ into $r'_{i,j}$ and $p'_i(r'_{i,j})$ for all j that are dishonest.

However, to be declared non-FAULTY, the adversary needs that

$p_i(x') = p'_i(x')$ for at least one value $x' = r_{i,j}$ where j is honest and $p_i \neq p'_i$ (indeed, otherwise the attack fails).

Let us call A the event that:

> the adversary succeeds that $p_i(x') = p'_i(x')$ for at least one value $x' = r_{i,j}$ where j is honest.

and let us call B the event that $p_i \neq p'_i$. Since the adversary knows both p_i and p'_i, he can check whether they are different or not. So, the adversary will win with probability

$$\text{prob}(A \mid B) = \frac{\text{prob}(A, B)}{\text{prob}(B)}.$$

Let us first analyze $\mathrm{prob}(A, B)$.

Since the degree of the polynomial is at most t, up to t values x might exist such that $p_i(x') = p'_i(x')$. So, $\mathrm{prob}(A, B) =$

$$1 - \mathrm{prob}(\text{at least one honest share is the same}) - \mathrm{prob}(p_i = p'_i) \leq$$

$$1 - \left(1 - \frac{t}{q}\right)^{t+1} - \left(\frac{1}{q}\right)^{t+1},$$

which is obviously less than

$$1 - \left(1 - \frac{t}{q}\right)^{t+1}. \qquad (1)$$

When $q = ct(t+1)$, then (1) becomes

$$1 - \left(1 - \frac{1}{c(t+1)}\right)^{t+1}$$

which is roughly $1 - e^{-c^{-1}}$. So, $\text{prob}(A, B) \leq 1 - e^{-c^{-1}}$.

Moreover, $\text{prob}(B) \geq 1 - \left(\frac{t}{q}\right)^{t+1}$, which when $q = ct(t+1)$ becomes $\text{prob}(B) \geq 1 - \left(\frac{1}{c(t+1)}\right)^{t+1} \geq 1 - 1/c$, for t large enough. So,

$$\text{prob}(A \mid B) \leq \frac{1 - e^{-c^{-1}}}{(1 - \frac{1}{c})}.$$

γ-Availability will definitely be achieved if

$$1 - e^{-c^{-1}}/(1 - 1/c) < \gamma.$$

Note: above assumes the adversary only changes one p_i into p'_i.

However, the adversary controls t wires, so can change up to t.

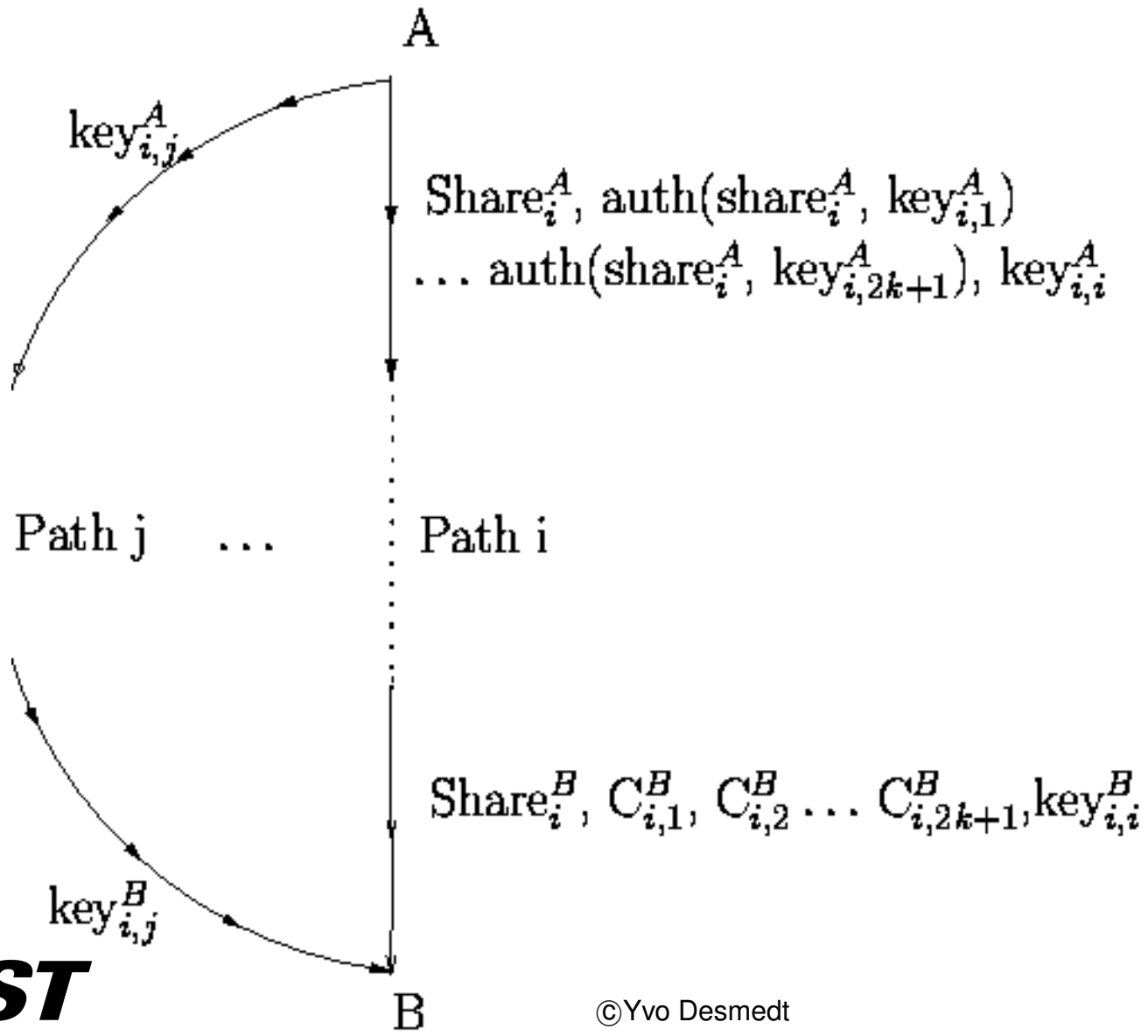
One can proof that when $q = O(t^2)$, that the best strategy is to only modify one p_i (see final paper). □

So, the communication complexity of this protocol is $O(t^2 \log_2 t)$.

4. OLD PROTOCOLS IN NEW "BARRELS"

Desmedt-Wang Eurocrypt 2002 protocol:

A makes shares from the secret using a $t+1$-out-of-$2t+1$ perfect secret sharing scheme. Then, for each i ($1 \leq i \leq 2k+1$), for each j:



If $|\{C^B_{i,j} : C^B_{i,j} = \mathrm{auth}(\mathrm{Share}^B_i, \mathrm{key}^B_{i,j})\}| \geq t+1$, then B accepts Share^B_i. Then from accepted shares B reconstructs the secret.

Above predates the concept of "almost secure" message transmission protocol.

Can trivially be modified into an $(0, 0, \gamma)$-secure one, as follows:

If from the accepted shares one can compute two possible secrets, then the receiver rejects.

Above runs in polynomial time, while Kurosawa-Suzuki (ICITS 2007) requires exponential time.

Theorem 2. *When using an authenticitation scheme in which the probability of a successful substitution is less or equal to* $1/q$*, then this protocol achieves* $(0, 0, \gamma)$ *security for* $q \geq ct(t+1)$ *when* t *tends towards infinity and* c *is appropriately chosen.*

Proof: Privacy: as in Desmedt-Wang, i.e., trivial.

Authenticity: similar as in the proof of Theorem 1.

Availability: the attacker will on all t wires she controls modify the shares, and make on these t wires consistent MACs. She needs at least that at one other wire, one of these keys will lead to a correct MAC. So, the probability is:

$$1 - (1 - \frac{1}{q})^{t(t+1)}.$$

Choosing $q = ct(t+1)$, then we obtain results similar to these in Theorem 1 □

5. Efficient Perfectly Secure Message Transmission

Step 1 The receiver does the following for $i, j := 1, \ldots, n$:

1. The receiver selects random element r_i.

2. The receiver constructs a $(t+1)$-out-of-n secret sharing scheme of r_i using the random polynomial p_i of degree at most t to obtain n shares $(s_{1i}, s_{2i}, \ldots, s_{ni})$.

3. The receiver sends polynomial p_i on wire w_i and share s_{ij} is sent on wire w_j.

Step 2 The sender does the following

1. The sender constructs a $(t+1)$-out-of-n secret sharing scheme

of M^A to obtain n shares $(m_1, m_2, \ldots, m_n)$.

2. For $i := 1, \ldots, n$ the sender receives polynomial p_i from wire w_i. The sender evaluates $p_i(0)$ as r_i. The sender calculates the value $d_i := r_i \oplus m_i$. These are termed correcting information.

3. For $i := 1, \ldots, n$ using the i^{th} shares received from each wire, error shares are identified. s_{ij} received from wire w_j is an error share if $s_{ij} \neq p_i(x_j)$.

4. The tuple of all identified error shares, called es_{ij}, is sent to the receiver via broadcast.

5. The correcting information - $(d_1, d_2, \ldots, d_n)$, is sent to the receiver via broadcast.

Step 3 The receiver does the following:

1. The receiver makes the following checks to identify the set of active wires of the first phase.

   Case 1: If the value of error share es_{ij} is different to the corresponding share sent out by the receiver in Phase 1 then wire j is identified as a faulty wire.

   Case 2: If the value of error share es_{ij} is equal to the corresponding share sent out by the receiver in Phase 1 then wire i is identified as a faulty wire.

   The set of honest wires (indicated as $HONEST$) is also constructed.

2. Using $HONEST$ the receiver computes shares of the secret

message M^A. This is done by computing $m_{w_i} := r_{w_i} \oplus d_{w_i}$ where $w_i \in HONEST$.

3. Using the computed shares from the step above, the receiver interpolates and obtains the secret message.

Theorem 3. *The above protocol achieves perfectly secure message transmission ($(0,0,0)$-security).*

For the proof, see the proceedings.

The communication complexity is $O(n^3 \log n)$.

However, using the technique of generalized broadcast, introduced by Srinathan-Narayanan-Rangan (Crypto 2004), we can reduce the

communication complexity to $O(n^2 \log n)$.

For details: see the proceedings.

6. CONCLUSIONS

We presented several protocols that require polynomial (in t) computation complexity and communication complexity. Our protocols require $O(n^2 \log n)$ communication complexity.

It is trivial to show that one needs to send at least (roughly) $n \log_2 n$ bits. The $\log_2 n$, comes from the bounds on secret sharing schemes.

Open problems: are there protocols with an $O(n \log n)$ communication complexity that achieve

- perfectly secure message transmission using 2 phases?
- $(0, 0, \gamma)$-secure message transmission using 1 phase?

Note: the problem has been solved when requiring a 1 phase $(0, 0, 0)$-secure message transmission protocol.

We expect to have a more detailed and corrected version of our paper on the IACR e-Print Archive around July 1, 2010.