



**IP-FP6-015964**

**AEOLUS**

**Algorithmic Principles for Building Efficient Overlay Computers**

**Deliverable D4.3.3**

**Algorithmic solutions for secure and rationally secure distributed computation.**

---

Responsible Partner: Università degli Studi di Salerno (I)

Report Preparation Date: September 2007

Contract Start Date: 01/09/05 Duration: 48 months

Project Co-ordinator: University of Patras (EL)



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Secure sharing of computational power</b>	<b>2</b>
2.1	Certified Access to Information: Different Viewpoints . . . . .	2
2.2	Certified access to a database . . . . .	3
2.3	Our Algorithmic Solution . . . . .	4
2.3.1	Verifiable Deterministic Envelopes . . . . .	5
2.3.2	Performance evaluation . . . . .	8
<b>3</b>	<b>Time stamping</b>	<b>10</b>
3.1	Classical results . . . . .	10
3.2	Universally Composable Time-stamping . . . . .	11
3.3	Our Algorithmic Solutions . . . . .	12
<b>4</b>	<b>Security Challenges in Wireless Networks</b>	<b>16</b>
4.1	Group Key Management in Wireless Sensor Networks . . . . .	16
4.2	Key Management in Ad-Hoc Networks . . . . .	17
4.3	Group Key Establishment Protocols in WSN . . . . .	18
4.4	Cryptographic Properties . . . . .	20
4.5	Secure Data Aggregation . . . . .	21
4.6	Our Algorithmic Solutions. . . . .	22
4.6.1	Key Management . . . . .	22
4.6.2	Key Pre-distribution . . . . .	23
4.6.3	Secure Data Aggregation . . . . .	24
<b>5</b>	<b>Commitment</b>	<b>27</b>
5.1	Our algorithmic solution: Hybrid trapdoor commitments . . . . .	28
5.2	Our algorithmic solution: Non-Malleable Commitments . . . . .	30
5.2.1	Applications to auctions . . . . .	32
<b>6</b>	<b>Secret sharing</b>	<b>34</b>
<b>7</b>	<b>Distributed Oblivious Transfer</b>	<b>36</b>
7.1	Applications . . . . .	37
7.2	Our Algorithmic Solutions . . . . .	38
<b>8</b>	<b>Web-Service Security</b>	<b>42</b>
8.1	Secure metering schemes . . . . .	43
8.1.1	Our algorithmic solution . . . . .	44
8.2	Validating orchestration of web services . . . . .	44
8.2.1	Our algorithmic solution . . . . .	45



# 1 Introduction

Secure distributed computation is an important techniques for designing secure protocols. Roughly speaking, the aim of secure distributed computation is to design *secure* protocols for the following scenario. We have  $n$  mutually distrusting parties  $P_1, \dots, P_n$  and  $P_i$  has private input  $x_i$ . The  $n$  parties wish to compute a fixed functionality  $F(x_1, \dots, x_n)$  of their private inputs without revealing any information about their private inputs. A typical example is a sealed-bid auction in which the input of each party is his bid and the function  $F(x_1, \dots, x_n)$  is simply the MAX-functionality. A secure distributed protocol for the MAX-functionality allows thus to compute the maximum bid in an auction without having to reveal the other bids, which is a desirable privacy property. At the end of the protocol each bidder is guaranteed that the outcome of the protocol is correct.

It is a classical result (called the *Feasibility Theorem*) that it is possible to evaluate any efficiently computable function in a secure way. The research conducted in this WP moves from two important drawbacks suffered by the classical constructions.

**Practical construction for specific problems.** Due to its generality, the protocols obtained from the Feasibility Theorem are not practical although the running time of all parties is polynomial. It is therefore an important research line to design practical protocols for special cases of the general feasibility theorem of practical relevance. In the second year we have considered important special cases that are of interest to global computing and that are relevant to the research conducted in other SPs. Specifically, we looked at protocols for *securely outsourcing computation* and *digital time-stamping* and we have studied how techniques from secure distributed computation can be used for securing *web services* and *workflow*. Moreover, global computers are often extended to allow communication with computationally weak devices (e.g., mobile and wireless devices). The need thus arises to re-design the protocols to keep into account the computational limitations of such devices.

**Securely composing protocols.** The security of the protocols obtained via the Feasibility Theorem only hold if the protocol is executed in isolation or if a central authority schedules protocol execution so to limit the concurrent execution of protocols. These assumptions are not justified in a global computer where it is often the case that protocols are executed concurrently, that a party is engaged in the execution of more than one protocol and no central authority is present. It is therefore an important research problem to design protocols that remain secure in global computers; that is, when each party may be executing several instances of the same protocol concurrently. We have thus looked at an important cryptographic primitive, the commitment, and given efficient constructions for it that remain secure even in global scenaria. Moreover, we have investigated the notion of hierarchical secret sharing that constitute a building block of protocols for secure distributed computation.

## 2 Secure sharing of computational power

The advances in communication technology of the last decades have made it possible the emergence of global-scale computer networks. Networks of such a large scale are currently used to share information for fast dissemination and efficient access. It has been observed that most of the nodes in a global size network are idle most of the time and it is very tempting to use the idle cycles of this huge computer to our advantage. The typical scenario that we envision consists of a user that *outsources* its computation to the network in order to harness the computational power that other nodes of the network are willing to share. Several researchers have considered the challenging issues related to load balancing and service discovery that naturally arise in this context and are object of research in other AEOLUS SPs. In this WP, we concentrate on the security issues related to the scenario outlined above and, more specifically, on the security issues relate on outsourcing computation on *private* data. Sharing computation across the network has been successfully achieved for several computation-intensive tasks like number-theoretic computation [15] or search for signs of extraterrestrial intelligence [16]. We notice that in both cases the input to the shared computation are public and security issues are trivial or non-existing. In other applications in which the outsourced computation is to be carried out on public data, most of the security issues are directly related to the problem of user authentication: is the user entitled to use the network for his own computation? In our work we have considered cases in which the input of the computation to be shared is to be retained private. Specifically, we have addressed the case in which the computation to be shared is the computation of the cryptographic data structure needed for a cryptographic primitive (the Certified Information Access primitive) that has been studied by AEOLUS researchers in the previous year [65] and has been the object of a (centralized) implementation (see Deliverable D4.3.1 for Year I).

### 2.1 Certified Access to Information: Different Viewpoints

In a global scenario, it is often the case that one entity has to access information owned by a different entity. This poses a number of security issues both from the database side and the user side.

For example, the owner of a database may require that only authorized users have access to the information or part of it. This is an instance of the well-known access control problem. Once a user has been granted access to the database, one wants to make sure that the user cannot infer additional knowledge by analyzing the answers to the issued queries. In other words, authorized users can obtain the information they required, but they cannot infer any other information from the received answers [108, 141]. On the other hand, a user may require that the database does not gain any information about specific content requested by a user [43, 74, 130, 151, 169]. This means that the database should “blindly” answer all the queries from authorized users in a way that all the requested information can be correctly reconstructed from the answers.

A slightly different approach is considered in the case of privacy-preserving data-mining (PPDM). The primary task of data-mining is to develop models about aggregated data, for example about the habits of the Internet users, about the loyal customers, etc. The main question of privacy-preserving data-mining (PPDM) is: Can we develop accurate models without access to precise information in individual data records? The latter question has proven to be difficult to solve.

Several different approaches for the PPDM have been developed and the main two of them are the randomization approach (based on the well-known statistical Randomized Response Technique) [73] and the cryptographic approach. Different randomization methods have been proposed, and securing of several data-mining algorithms has been undertaken.

The above problems mainly address the confidentiality of the information. In [140], the authors deal with the problem of guaranteeing the consistency of the answer, sent by the database to the user in response of a query, with the information actually contained in the database. We assume the possibility that a database would be willing to give wrong answer to queries issued by a user. In this setting, since the user does not know in advance which is the actual information he will receive from the database, there would be no way to distinguish between a correct answer from a random one.

## 2.2 Certified access to a database

In this WP we have considered a specific example of access to database. The Certified Information Access (CIA for short) is a cryptographic primitive introduced by [88] that provides *certified* access to a database. Specifically, the database owner publishes a snapshot of its current database, which we refer to as the *Public Information*, on a trusted entity. Once the public information is available, any user may issue queries to the database. The database owner gives the user the query result along with a short proof that the result is consistent with the published snapshot. CIA is particularly suited for slowly changing databases like the Domain Name System. In this case a query returns the IP address associated to a name.

One trivial way of implementing CIA is to publish the whole content of the database on a trusted server. A user can thus compare the received reply with the one contained in the public copy of the database (or actually directly query the trusted server). This solution is, of course, neither secure nor efficient. Since the database has to publish its whole content, the confidentiality of the information therein contained is compromised. Furthermore, since all the elements in the database need to be transmitted and stored, the communication and space complexity of this solution is linear in the size of the database

For these reasons the public information should satisfy the following properties:

- *Compactness*: The size of the public information should be considerably smaller than the size of the database.

- *Confidentiality*: The public information should not reveal anything about the actual content of the database.
- *Correctness*: A correct answer to a query should be consistent with the public information with probability one.
- *Soundness*: Any wrong answer to a query will be detected with high probability.

Currently, a way of implementing CIA primitives is by means of a recently introduced cryptographic primitive, namely *mercurial commitments* introduced and studied in [65, 69, 133, 140]. Unfortunately, the implementation of such primitive are computationally intensive. On one hand, the generation of the public information is time consuming also on current servers. On the other hand, although the verification procedure can be easily executed on a PC in few seconds, it still requires much more time on mobile devices.

### 2.3 Our Algorithmic Solution

In [88] we present a distributed architecture for a CIA service. Here, the database owner builds a tree-like cryptographic data structure representing the database. Each node in the tree requires the computation of a few modular exponentiations. The system presented in [88], by using “pre-computation” peers, securely distributes, for each node in the tree, the computation of the modular exponentiations. The database owner is simply required to locally combine partial results obtained by the peers. Pre-computation peers are only required to pre-compute elements of the form  $g^r \bmod p$  where  $g$  and  $p$  are fixed elements and  $r$  is randomly chosen by the peer.

Specifically, the computation associated with a node of the cryptographic data structure requires the computation of pairs  $(x, g^x \bmod p)$  for randomly chosen  $x$ , where  $g$  is a fixed generator of the underlying cyclic group  $Z_p$ . We assume that several peers compute pairs  $(r, g^r \bmod p)$ . Whenever a user needs to compute a pair  $(x, g^x \bmod p)$  it queries two peers and obtains  $(r_1, g^{r_1} \bmod p)$  from the first and  $(r_2, g^{r_2} \bmod p)$  from the second. The user then computes the pair  $(x, g^x \bmod p)$  as  $(x, g^x \bmod p) = (r_1 + r_2, g^{r_1} \cdot g^{r_2} \bmod p)$ . In other words, we have outsourced the computation of a modular exponentiation at the cost of one modular multiplication (and one modular addition). The outsourcing is secure since neither of the two peers has any information about  $x$ . If a higher security level is desired then the user might query more than one peer and combine the result by modular multiplication.

We stress that even this simple scenario has an important security feature that makes the outsourcing of computation non-trivial. Indeed in outsourcing the computation of the pair  $(x, g^x \bmod p)$  we have to make sure that nothing about  $x$  is leaked for otherwise the cryptographic data structure being constructed loses its security properties. This is the reason why we cannot simply split the computation among peers but we to do so in such a way that the secrecy of  $x$  is guaranteed.

In [40] we continue on the same line of research and we present an efficient distributed implementation of a CIA service that allows the database owner to *completely* outsource the tree computation and not simply the construction of the nodes of the tree-like cryptographic data structure. In the former solution the database owner securely distributes the computation but she locally combines the partial results. In the presented solution the entire database is sent to an external computation entity and the database owner simply obtains tree representing the database. We have implemented a prototype that uses the PUBWCL library [45] that is currently being integrated in the overlay computing platform developed by AEOLUS.

Clearly, the information in the database need to be hidden before they are transferred. Unfortunately, it is not possible to use “simple” (neither randomized nor deterministic) encryption schemes. To this aim, we first present a new primitive, which we call Verifiable Deterministic Envelope (or VDE for short), that allows the “encryption” of all the elements in the database. The “encrypted” version of the database is then transferred to a web-cluster that computes the tree associated to the database and sends it back to the database owner. In this scenario, the database owner still needs to locally compute an “encryption” of each element in the database but then the method presented in [88] can be used to speed the computation.

As in [88] we describe a solution for *static* databases, i.e., databases in which the content does not change. This is however without loss of generality as our main target will be slowly-changing databases for which every time the database is updated a new snapshot is published. For rapidly changing databases, one could use the construction of [133] which is however not very practical.

We assume that there exists a trusted entity that does not collude with the entity holding the database. The trusted party is only required to generate some of the public parameters for the scheme and to store the database’s snapshot. Furthermore, the system comprises a sufficient number of peers whose only role is to compute modular exponentiations. We assume that such peers are honest, that is, they properly execute the protocols, but a small fraction of them may be curious, in the sense that they may collude in order to infer additional information from the messages they have exchanged.

In the next sections we give an overview of the technique used.

### 2.3.1 Verifiable Deterministic Envelopes

Our algorithmic solution is based on the concept of *Verifiable Deterministic Envelope*, or VDE for short which is introduced in [40]. We will show how VDEs can be efficiently constructed and how they can employed to outsource computation.

Informally speaking, a VDE is a deterministic privately computable and verifiable function  $f$ . That is, the function is privately computable in the sense that, given  $x$ , it is not possible to compute  $f(x)$  and nor it is possible, given  $y$  and  $x$ , to check whether  $y = f(x)$ . On the other hand the *owner* of  $f$  can compute  $y = f(x)$  and prove to another

party that the computation has been correctly carried out.

**Definition 1 (Verifiable Deterministic Envelope)** *A Verifiable Deterministic Envelope scheme (VDE for short) is a triple  $(\text{EnvKeyGen}, \text{EnvelopeGen}, \text{EnvelopeOpen})$  of algorithms, involving a sender  $S$  and a receiver  $R$ , described as follows:*

- *A randomized key generation algorithm  $\text{EnvKeyGen}$  executed by the sender. The algorithm  $\text{EnvelopeGen}$ , on input a security parameter,  $k$ , generates a pair  $(pk, sk)$ . The value  $pk$  is made public while  $sk$  is kept secret by the sender.*
- *A deterministic envelope generation algorithm  $\text{EnvelopeGen}$  executed by the sender. The algorithm  $\text{EnvelopeGen}$ , on input a string  $x$ , and the pair  $(pk, sk)$  computes a VDE for  $x$ . In order to simplify the notation, we will denote by  $\text{EnvelopeGen}(x)$  the VDE computed as  $\text{EnvelopeGen}(x, (pk, sk))$ .*
- *An interactive opening protocol  $\text{EnvelopeOpen}$  executed by both the sender and the receiver. The protocol is started by the receiver who provides a string  $x$ . At the end of the protocol, the receiver is able to compute the (correct) value for  $\text{EnvelopeGen}(x)$ .*

The following is a security definition for VDE:

**Definition 2** *A VDE scheme  $(\text{EnvKeyGen}, \text{EnvelopeGen}, \text{EnvelopeOpen})$  is OW-secure if the following holds:*

- a. For any  $x$ , given  $pk$  and  $sk$ , the sender can compute  $\text{EnvelopeGen}(x)$  in polynomial time.*
- b. For any  $x$ , given  $pk$ , the receiver  $R$  by executing the Opening protocol  $\text{EnvelopeOpen}$  can compute in polynomial time the (correct) value  $y = \text{EnvelopeGen}(x)$ .*
- c. For any  $x$ , given  $pk$ , there exists no polynomial time algorithm that computes  $y = \text{EnvelopeGen}(x)$  with non-negligible probability.*
- d. For any VDE  $y = \text{EnvelopeGen}(x)$ , given  $pk$ , there exists no polynomial time algorithm that computes  $x$  with non-negligible probability.*

**A practical construction.** We next describe a practical instantiation of the VDE primitives based on the RSA cryptosystem.

The different phases of the VDE are implemented as follows:

- $\text{EnvKeyGen}(\cdot)$ : The algorithm  $\text{EnvKeyGen}$  outputs the pair  $((n_E, e_E, d_E), (n_S, e_S, d_S))$ , where
  - $n_S < n_E$ ;
  - $(n_E, e_E, d_E)$  are the parameters for the RSA encryption scheme;

- $(n_S, e_S, d_S)$  are the parameters for the RSA signature scheme;
- The parameters  $pk = ((n_E, e_E), (n_S, e_S))$  are published while  $sk = (d_E, d_S)$  is kept secret.
- **EnvelopeGen** $(x, (pk, sk))$ : Given a value  $x$ , the sender computes  $e = E(\text{Sig}(x, d_S), e_E) = (x^{d_S} \bmod n_S)^{e_E} \bmod n_E$ . The value  $e$  is sent to the receiver.
- **EnvelopeOpen**: Let  $x$  be a key. The Opening protocol works as follows:
  - The Receiver  $R$  sends  $x$  to the  $S$ .
  - The Sender  $S$  computes  $g = \text{Sig}(x, d_S) = x^{d_S} \bmod n_S$  and sends  $g$  to  $R$ .
  - The receiver checks whether  $g^{e_S} = x \bmod n_S$  and computes **EnvelopeGen** $(x, (pk, sk)) = g^{e_E} \bmod n_E$ .

**Using VDE to outsource computation.** An implicit assumption is that the DB Owner uses a *deterministic* algorithm to associate the elements in the database to the leaves of the tree, e.g., the value  $v = D(x)$  is associated to the leaf identified by the binary representation of the key  $x$ . Such a deterministic approach allows the user to *verify* that the received proof corresponds to *a specific path* that, in turn, corresponds to the queried key. Notice that the above argument does not rule out the possibility of (pseudo-)randomly assigning keys to tree leaves. However, if the DB Owner uses a randomized strategy for such assignment, for each key in the database, there should exist a “certified” way of binding the key along with the randomness used to create the path in the tree.

Thus, a deterministic placement of database elements to tree leaves guarantees the security of the CIA service. On the other hand, if the DB Owner is willing to outsource the tree construction, the problem of confidentiality of information contained in the database has to be taken into account. Notice, however, that there exist two conflicting requirements. On one hand, the DB Owner needs to hide the information contained in the database from the entity that constructs the tree. Specifically, it should not be possible to check whether there exists in the database some value associated to a specific key. On the other hand, the USER should be able to verify that the answer received for a given query actually corresponds to the submitted key.

A first solution could be to (deterministically) encrypt the keys of the elements in the database (along with the values associated to them). If the DB Owner uses a symmetric key encryption scheme, she can securely hide all the information from the computing entity. Unfortunately, after the tree has been constructed, she needs to publish the key used to encrypt the keys in order to allow the users to properly verify the answers.

A second possible approach could be to encrypt each key in the database using a public key encryption scheme. In this case, the leaf associated to the key is identified by the binary representation of the encryption of the key itself. This approach allows the USER to compute by herself the “permuted” value of the key. Unfortunately the entity

that computes the tree can execute the same operation herself by obtaining information on the existence of some specific key in the database.

Another possible solution could be to place each element in the database in the leaf identified by the (binary representation of the deterministically computed) signature of the key. Notice that also this solution does not prevent the computing entity to check whether or not some key belongs to the database by simply “verifying” if there exists a valid signature for the key under consideration.

For the above reasons, we need a way of deterministically permuting the keys in the database in a way that the computing entity will not be able (a) to compute the key given its permuted value and (b) to compute the permuted value of any key of its choice. Furthermore, the user, by interacting with the DB Owner, should be able (a) to compute the permuted value associated to a given key and (b) to verify that the computed value is the one used by the DB Owner during the tree construction phase.

As stated in the previous section, such properties can be achieved by using a VDE scheme for permuting the elements in the database. Given such a permutation, the user can, by interacting with the DB Owner, compute the value associated to each key. On the other hand, the computing entity, given the “permuted” database, cannot tell whether or not a key is therein contained without interacting with the DB Owner.

### **2.3.2 Performance evaluation**

In this section we briefly discuss the performance obtained by a prototype implementation of our protocol for securely outsourcing the computation of the cryptographic data structure associated with the CIA protocol. Figure 1 show the running time of the local computation as function of the size of the database both when the computation is outsourced and when it is carried out locally. As we can see even for modest sized database (10000 elements) the running time of the local implementation is about 3 hours. In contrast the running time of the outsourced computation is so little that it does not even show on the graph. If we turn our attention to Figure 2, we see that the running time for 10000 elements is about 6 minutes. We can thus conclude that our experiments show that it is possible to harness the power of a global computer without having to sacrifice security.

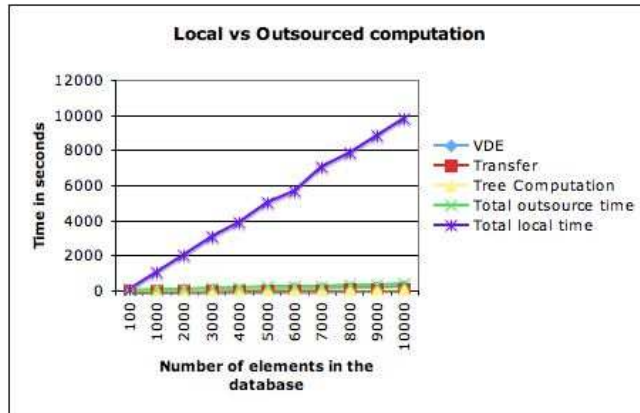


Figure 1: Local vs. outsourced

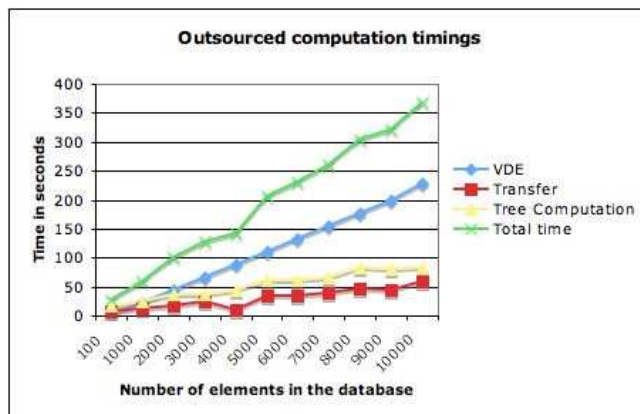


Figure 2: Outsourced

### 3 Time stamping

Time-stamping is used to determine which of certain two bit-strings existed earlier. Here the existence of a bit-string really means that it had entered the knowledge set of some party who had a vested interest in showing that it is as old as possible. The existence of bit-strings typically denotes that some event has occurred, and the most common usage of time-stamps is to find out whether signing some message occurred before or after the invalidation of (the certificate of) the signature key. Both of these events produce a bit-string — the signature and the revocation notice (signed by the certification authority or somebody delegated by it). The recipient of the signature (a party that obtains some rights as the consequence of signing the message) tries to show the earliest possible creation time for the signature, while the signer wants to show the earliest possible time for the eventual compromise of the signing key. Used in such form, time-stamps are essential in ensuring the non-repudiation of digital signatures. Besides non-repudiation, there are other possible applications for time-stamps, e.g. in bidding, where a bid is valid if and only if it has been received before a preset time.

We will give an overview of the classical approaches for time-stamping and continue with our work on universally composable time-stamping.

#### 3.1 Classical results

A naïve way to time-stamp bit-strings is to introduce a trusted third party — the Time-Stamping Server (TSS) — that the other parties query with the bit-strings for which they want to obtain time-stamps. In response to a query  $x$ , the TSS appends the current date and time  $t$  to  $x$ , signs  $(x, t)$  and returns the result. A time-stamp is deemed valid if it contains a valid signature of the TSS. Two time-stamps are compared by comparing the times included in them.

The naïve scheme has two distinct shortcomings. First, it includes a trusted party. There are no system-internal means to check that the TSS is behaving according to the specification. In particular, nothing prevents it *back-dating* the documents — including a time in a time-stamp that is earlier than the current time. Second, a time-stamp is valid only if it contains a valid signature. A signature is valid only if the corresponding key has not been revoked. The revocation of signature keys is considered to be common a occurrence. Indeed, time-stamps themselves were introduced to cope with this.

The naïve scheme augments the bit-strings with *absolute* time (if the TSS is honest) — each time-stamp can be pin-pointed at the axis of time. In *relative* time-stamping, one can only compare two time-stamps but cannot find the absolute time of their creation. All known time-stamping schemes that do not require trusted parties only provide relative time-stamps. The TSS is still a part of the system, but only to facilitate the communication between parties, not to make authoritative statements. The first scheme without trusted third parties was proposed by Haber and Stornetta [115]. In this scheme, as well as in all subsequent schemes, the security against back-dating is achieved by *linking* the time-

stamps together — by letting each time-stamp contain the cryptographic digest of the previous time-stamp. If the digests of time-stamped documents are  $X_1, X_2, \dots$ , then the time-stamp for the  $n$ -th document  $X_n$  is  $L_n = H(X_n, L_{n-1})$ . To compare two time-stamps,  $L_n$  and  $L_m$  where  $m < n$ , the *linking information*  $(X_{m+1}, \dots, X_n)$  has to be retrieved and the equality

$$L_n = H(X_n, H(X_{n-1}, \dots H(X_{m+1}, L_m) \dots))$$

has to be verified.

Note that during a verification the same amount of work has to be done as the TSS did between issuing the time-stamps  $L_m$  and  $L_n$ . By directly linking each time-stamp not only with a previous, but also with some earlier time-stamp the number of hash function invocations can be reduced to  $O(\log n)$  [54].

Where can the linking information be retrieved from? One obvious source is the TSS (who has to maintain an archive in this case) but it becomes a single point of failure in this case. Publishing the database of  $X$ -s ensures that everybody has access to it, but it may be prohibitively expensive. It turns out that time-stamping may be done in rounds (whose lengths may be in order of hours or days) and a digest of the database of  $X$ -s may be published only at the end of each round. With each time-stamp  $L_n$  published in the  $r$ -th round, we store linking information from the digest  $R_{r-1}$  of the  $(r-1)$ -st round to  $L_n$  and from  $L_n$  to  $R_r$ . Given two time-stamps from different rounds, we may immediately make sure which one is earlier. Given two time-stamps from the same round, the accompanying linking information will allow us to reconstruct the linking information from one of them to the other [54].

### 3.2 Universally Composable Time-stamping

Time-stamping is an important data integrity protection mechanism the main objective of which is to prove that electronic records existed at a certain time. On the one hand, the scope of applications of time-stamping is very large and the combined risks related to time stamps are potentially unbounded. Hence, the standard of security for time-stamping schemes must be very high. On the other hand, time-stamping is seldom considered as a stand-alone service and therefore it is important to study the security of time-stamping schemes in composition with other protocols and primitives. The *Universal Composability (UC) framework* provides a suitable apparatus for such an approach to security.

Several constructions of time-stamping schemes have been proposed [33, 54, 115, 116], based on collision-resistant hash functions, but only few analytical arguments confirm the security of these schemes. Two early attempts to sketch a security proof [33, 116] were recently shown to be flawed [59] and a new security proof was presented [59] under the assumption that every time stamp has a pre-determined "shape" so that two different time stamps with the same shape will always produce a collision for the underlying hash function. The security proof used an assumption that the number  $N$  of different "shapes" is polynomially bounded and the efficiency of the reduction explicitly depends on  $N$ . It was

shown that every adversary  $A$  for a time-stamping scheme with running time  $\tau$  and with success probability  $\varepsilon$  can be transformed to a collision-finder with running time  $\tau' \approx 2\tau$  and with success probability  $\varepsilon' \approx \frac{\varepsilon^2}{N}$ . Such proofs are not completely satisfactory for two main reasons:

- For obtaining two stamps with the same shape the collision-finder has to *rewind*  $A$ , which means that such proofs cannot be adapted to the standard UC-framework.
- The security guarantees provided by such proofs are low because the reduction is quadratic and because the efficiency  $\varepsilon'$  of the collisions finder is divided by  $N$ , which may be very large in global-scale time-stamping solutions.

In the framework of the AEOLUS project we obtained the following results:

- By slightly modifying the commonly used hash tree structure and by explicitly including a trusted *auditing functionality* into the security model [55, 56] we obtained a universally composable time-stamping scheme. The security reduction for this scheme is almost exact:  $\frac{\tau'}{\varepsilon'} \approx \frac{\tau}{\varepsilon}$ . This is much better compared to  $\frac{\tau'}{\varepsilon'} \approx 2N \frac{\tau}{\varepsilon^2}$  which was achieved in [59].
- We studied the influence of collision-finding attacks on the security of time-stamping schemes [57]. We derived necessary and sufficient conditions for *client-side hash functions* (used to shorten the documents at the client side) and showed that neither collision-resistance nor 2nd preimage resistance is necessary for secure time-stamping. Moreover, we show that *server side hash functions* (used for establishing one way causal relations between time stamps) can even be not one-way. However this is only true if the hash functions are viewed as black boxes. We also took a more detailed look at the structure of practical hash functions by studying the Merkle-Damgård (MD) hash functions and showed that attacks, which are able to find collisions for MD hash functions with respect to *randomly chosen initial states*, also violate the necessary security conditions for client-side hash functions. As a practical consequence, MD5, SHA-0, and RIPEMD are no more recommended to use as *client-side hash functions* in time-stamping. However, there is still no evidence against using MD5 (or even MD4) as *server-side* hash functions.

### 3.3 Our Algorithmic Solutions

In [58] we prove in a non-black-box way that every bounded list and set commitment scheme is *knowledge-binding*. This is a new and rather strong security condition, which makes the security definitions for time-stamping much more natural compared to the previous definitions, which assume *unpredictability* of adversaries. As a direct consequence, list and set commitment schemes with partial opening property are sufficient for secure time-stamping if the number of elements has an explicit upper bound  $N$ . On the other hand,

white-box reductions are in a sense strictly weaker than black-box reductions. Therefore, we also extend and generalize the previously known reductions. The corresponding new reductions are  $\Theta(\sqrt{N})$  times more efficient, which is important for global-scale time-stamping schemes where  $N$  is very large.

Commitment schemes are basic building blocks in numerous cryptographic protocols. The most important properties of commitment schemes are binding and hiding. A commitment is hiding if it reveals no information about the committed message and binding if it is impossible to change the committed message afterwards without detection. First such schemes for committing a single bit were proposed by Blum [36] and by Brassard *et al* [46] and were proven secure under the hardness of factoring assumption. Later works have significantly improved their efficiency and weakened the underlying complexity theoretic assumptions, see [82, 118] for further references. Here, we study the so called *partially releasable* commitments, in which one can compute a commitment (also called *digest*) for a list  $\mathcal{X} = (x_1, \dots, x_N)$  of bit-strings, so that it is possible to partially open the commitment for every  $x_i \in \mathcal{X}$  without disclosing the other elements of  $\mathcal{X}$ . For opening  $x_i$  it is sufficient to present a decommitment string  $s_i$  (also called *certificate*). Achieving the hiding property is somewhat trivial, as one can always add another layer of commitments. Hence, our main emphasis is on the binding property. List commitments [26, 34, 147] that are only binding are known as *one-way accumulators*.

In particular, we analyze the security of a *time-stamping* protocol, where clients send their requests  $x_1, \dots, x_N$  to a Time-Stamping Server (TSS) who computes the commitment  $c$  and sends the corresponding certificates  $s_1, \dots, s_N$  back to the clients. If  $c$  is published in an authentic way then everybody can verify that  $x_i$  was generated before  $c$  was published. This principle is used in practical time-stamping schemes [116] where  $c$  is computed as the root of a hash tree. List commitment schemes were believed to be exactly what one needs for such kind of time-stamping. However, Buldas *et al.* [59] pointed out a flaw in the security proof of [116]. By giving a carefully crafted oracle separation they showed that pure collision-resistance is insufficient to prove that the hash tree time-stamping schemes [116] are secure. In other words, either there are collision-resistant functions that are still insecure for time-stamping, or the security of time-stamping schemes follows from currently unknown complexity-theoretic results. The key point of this paradoxical result is that the number of committed elements is potentially unbounded. We prove that all list and set commitments, where the cardinality of  $\mathcal{X}$  has an explicit bound  $|\mathcal{X}| \leq N$ , are suitable for time-stamping. The proof is given in the exact security framework and is  $\Theta(\sqrt{N})$  times more efficient than the previous reduction [59]. This improvement is especially valuable for global-scale time-stamping schemes in which  $N$  is very large.

We also show that all binding bounded list and set commitments are *knowledge-binding*. This is a new and extremely strong security requirement inspired from the security of time-stamping schemes. Its strength is comparable to the *plaintext awareness* property, which is defined for public key encryption. The knowledge-binding property is also much more intuitive requirement for time-stamping schemes than the previous ones [57, 59], which use

unpredictable probability distributions to model the stream of “new documents” sent to a TSS. Roughly, the knowledge-binding property states that for every efficient TSS, it is possible (by observing the commitment procedure) to efficiently extract the list  $\mathcal{X}$  of all documents that can be opened by the TSS in the future. The dedicated extractor must know only the internal coin tosses of TSS and some public parameters. Consequently, even if the TSS is malicious, it must *know* the whole list  $\mathcal{X}$  before the corresponding commitment is published. This allows to prove the security in the classical *ideal vs real world* comparison framework [110, pp.622–631,697–700].

Moreover, the notion of knowledge-binding commitments can be useful in other cryptographic protocols, because the ability to open a commitment does not change in time and we may skip the proofs of knowledge in the commitment phase. On the other hand, the corresponding security proofs are not black box. This means that once we have an efficient adversary  $A$  that breaks the knowledge-binding condition *we know that there exists* an efficient adversary  $A'$  that breaks the binding property of the corresponding commitment scheme. However, we may have no efficient ways to construct  $A'$ . Therefore, in reality the knowledge-binding property can be violated but the commitment scheme may still be practically binding—the efficient breaking procedure exists but is not known. Black-box security proofs in turn give an efficient procedure for constructing  $A'$  from  $A$ . In this sense, the black-box security proofs give substantially stronger security guarantees for a fixed hash function (e.g. SHA-1) than the non black-box ones. So finding black box proofs for security properties equivalent to knowledge extraction is still an important open question.

In [53] we prove that there are no black-box reductions from Collision-Free Hash Functions to secure time-stamping schemes, which means that in principle secure time-stamping schemes may exist even if there exist no collision-resistant hash functions. We show that there is an oracle relative to which there exist secure time-stamping schemes but no hash function is collision-free. The oracle we use is not new — a similar idea was already used by Simon in 1998 to show that collision-free hash functions cannot be constructed from one-way permutations in a black-box way. Our oracle contains a random hash function family  $f$  and a universal collision-finder  $A$ . We show that hash-tree time-stamping schemes that use  $f$  as a hash function remain secure even in the presence of  $A$ . From more practical view, our result is an implicit confirmation that collision-finding attacks against hash functions will tell us quite little about the security of hash-tree time-stamping schemes and that we need more dedicated research about back-dating attacks against practical hash functions.

Cryptographic hash functions transform a message  $X$  of an arbitrary length into a digest  $h(X)$  of a fixed length. They have several applications, such as electronic signatures, fast Message Authentication Codes (MACs), secure registries, time-stamping schemes, etc. Though the range of hash function applications is growing rapidly, not much is known either about suitable design criteria or about how to formalize the security conditions for hash functions demanded by applications.

Security proofs of applications often assume the *collision-freedom* of hash functions and this gives an impression as if such a strong security requirement was necessary. Recent success in finding collisions for practical hash functions (MD4,MD5, RIPEMD, SHA-0) by Wang *et al.* [175,177,178] and later improvements [163,176] forced us to revisit the security proofs in order to clarify for which practical implementations the collisions are a real threat.

This paper focuses on one particular application of cryptographic hash functions – time-stamping. For a long time it was believed that collision-freedom is a necessary and sufficient condition for the security of hash-tree based time-stamping schemes. In 2004, it was shown by Buldas and Saarepera [59] that collision-freedom is probably *insufficient* to prove that unbounded hash-tree time-stamping schemes [28,116] (without explicit restrictions to the length of hash-chains) are secure. Buldas and Laur [57] then showed that collision-freeness (and even one-wayness) is also *unnecessary*—once there are hash functions that are secure for time-stamping, then there also exist hash functions which are secure for time-stamping but are not even one-way (and hence, not collision-resistant). This result shows that breaking a particular hash function in terms of collisions (even if they are meaningful textual documents) does not necessarily mean that this particular hash function is insecure for time-stamping.

In this paper we go even further. We show that collision-free hash functions cannot be constructed from secure time-stamping schemes in a black-box way. This means that even if one finds a universal collision-finder that ”breaks” all known hash functions, there may still exist hash functions that remain secure for time-stamping. We show that there is an oracle relative to which there exist secure time-stamping schemes but no hash function is collision-free. The oracle we use is not new — a similar construction was already used by Simon [168] to show that collision-free hash functions cannot be constructed from one-way permutations in a black-box way. Our oracle contains a random hash function family  $f: \{0,1\}^{2k} \rightarrow \{0,1\}^k$  and a universal collision-finder  $A$ . To prove the main result of this paper we will show that hash-tree time-stamping schemes with the random hash function  $f$  remain secure in the presence of the universal collision-finding oracle  $A$ .

From the practical point of view, our result is another confirmation that collision-finding attacks against hash functions will tell us quite little about the security of time-stamping schemes that use these ”broken” hash functions. Therefore, to study the security of hash-based time-stamping schemes, it is insufficient to study the feasibility of collision-finding attacks. Instead, we need dedicated research on practical back-dating attacks. To our knowledge, there is only one work published on this issue [125].

## 4 Security Challenges in Wireless Networks

In this section we describe our contributions for the the extensions of global computer to wireless users. Our work has mainly concentrated on security issues in wireless sensor networks (see [159] for a complete overview).

### 4.1 Group Key Management in Wireless Sensor Networks

Wireless Sensor Networks are considered as very large systems comprised of small sized, low-power, low-cost sensor devices that collect detailed information about the physical environment. Each device has one or more sensors (e.g. light, heat, movement, chemical presence, etc.), embedded processors and low-power radios, and is normally battery operated. Examining each such single device individually, might appear to have small utility. The realization of Sensor networks however, lies in using and coordinating a vast number of such devices and allows the implementation of very large sensing tasks. The system is deployed in areas of interest (ranging from homes to inaccessible terrains, disaster places, etc.) making them smart spaces where fine grained monitoring services and applications can be provided [17].

The unique characteristics of this regime give rise to very different design trade-offs than current general-purpose systems. The realization of such efficient, robust and secure ad-hoc networking environments is a challenging algorithmic, systems and technological task. Large numbers of such tiny and resource-constrained devices should self-organize into an ad-hoc network under highly dynamic ambient conditions, carrying out computations locally and engaging into a collaborative computing and communication effort. The required solutions differ significantly, not only with respect to classic distributed computing but also with respect to ad-hoc networking. To further emphasize on the difference consider that in sensor networks (i) the number of interacting devices is extremely large and dense compared to that in a typical ad-hoc network, (ii) the resources of each node are very limited, (iii) there is no fixed infrastructure, (iv) the network topology is unknown before deployment and (v) there is a high risk of physical attacks in unprotected sensor nodes.

Such systems should at least guarantee the confidentiality and integrity of the information reported to the controlling authorities regarding the realization of environmental events. Therefore, key distribution is critical for the protection in wireless sensor networks and the prevention of adversaries from attacking the network. However, key management and establishment can be a difficult task in such networks and may waste the limited energy resources of the devices. The constraints of sensor node hardware influence the type of security mechanisms and protocols that can be hosted on a sensor node platform. Moreover, the ad hoc networking topology makes it susceptible to link attacks ranging from passive eavesdropping to active interference. Therefore, the choice of a key establishment protocol for the creation of a shared, secret key must be done very carefully and should exhibit the following critical properties: (i) *availability*, in the sense that any sensor

node or service of the whole wireless network must be available whenever required, (ii) *key authentication*, assuring only intended nodes can access a key, (iii) *integrity*, ensuring that there is no unauthorized data modification and (iv) *confidentiality*, by providing security measures in order to avoid eavesdropping. Furthermore, some additional requirements are needed for the evaluation of key distribution in wireless sensor networks: (a) *scalability*, in the sense that they should operate efficiently in extremely large networks composed of huge numbers of nodes, (b) *efficiency* with respect to both energy and time and (c) *fault-tolerance* as sensor devices are prone to several types of faults and unavailabilities, and may become inoperative (permanently or temporarily). In this sense, group key establishment is potentially more suitable than pairwise key establishment as sensors do not waste energy every time they wish to communicate with another device by establishing a new shared secret key.

## 4.2 Key Management in Ad-Hoc Networks

Secure key management is considered a critical and complex issue in the research arena of mobile ad hoc networks. Due to the fact that ad hoc networks are dynamic in topology and functions like routing, mobility management etc. are performed by the nodes in a self-organized manner, a secure communication infrastructure needs to be established for a key management protocol. During the years a number of protocols for managing keys in ad hoc network have been proposed, which can be classified in *key distribution*, *key agreement* and *key pre-distribution protocols*.

**Key Distribution.** In this type of protocols, an authority creates or otherwise obtains secret values and securely distributes them to other nodes. The preliminary type of key management protocol using key distribution is Kerberos, where nodes solely rely on a trusted third party called a key distribution center (KDC), who has to be always online. The problem with ad hoc networks is that there is no fixed infrastructure and nodes can leave or join the network at any time. So, existence of an always available node is very difficult (if not impossible). Furthermore these protocols depend on a centralized entity, which can be hazardous in case of single point of failure. One way to solve this problem is to distribute the service to a certain number of nodes, so that it does not depend on a single entity. Some proposals of key management in ad hoc networking using this technique use a fully distributed certification authority [182] or a partially distributed certification authority [135].

**Key Agreement.** Another type of protocols is contributory key-agreement protocols, where each node contributes an input to establish a secret by doing successive pair wise message exchange. Key agreement protocols do not require the existence of a trusted third party and are fully distributed and self-organized. The dynamic nature of ad hoc networks makes it difficult for key agreement protocols. All the nodes must have to be online during

the time of establishing a secret key. Failure of any node during key establishment process may force the process to begin from scratch. Some proposals using this technique are [21, 120].

**Key Predistribution Schemes.** In a key pre-distribution scheme (KPS), some secret key information is distributed to all nodes during the network set up phase. A major problem in this approach is to determine the network structure, since in each KPS the nodes must know their neighborhood before deployment, e.g. pairwise keys can be established between these nodes (and only these nodes). Unfortunately in real deployments it is difficult to determine such kind of information. One methodology to solve the above problem is to consider a key pre-distribution scheme in which each sensor node receives a random subset of keys from a large key pool before deployment. To agree on a key for communication, two nodes find a common key (if any) within their subsets and use it as shared secret key. Now, the existence of a shared key between a particular pair of nodes can not be ensured but is instead probabilistically guaranteed (this probability can be tuned by adjusting the parameters of the scheme).

A number of key pre-distribution schemes to solve the above mentioned problem are proposed, some of those can be found in [67, 94, 165].

### 4.3 Group Key Establishment Protocols in WSN

Most group key establishment protocols are based on generalizations of Diffie-Hellman key exchange protocol [91]. The first attempt for the construction of such protocols was made by Ingemarsson, Tang and Wong [96] that arrange the participants in a logical ring via a synchronous start up phase. The protocol completes in  $n - 1$  rounds, where  $n$  is the number of the participants.

Burmester and Desmedt presented in [60] a more efficient scheme which requires only three rounds. However, the protocol's disadvantage is that (i) every participant must perform  $n + 1$  exponentiations and (ii) communication is based on concurrent broadcasts that lead to high number of collisions, a situation very common in wireless sensor networks that affects performance [70]. Moreover, the authors do not provide a proof of security (in the stronger sense of semantic security). Recently, Katz and Yung [124] proposed a more general framework that provides a formal proof of security for this protocol. In Hypercube protocol [30] the participants in the network are arranged in a logical hypercube. This topology decreases the number of transmitted data and exponentiation operations, but still the protocol is very demanding for use in sensor networks.

One of the most efficient protocols in the literature for group key management is the third protocol GDH.3 of Steiner, Tsudik and Waidner presented in [170]. This protocol requires serial execution of computations that makes it inefficient for highly dynamic networks with large number of nodes. More precisely, this protocol may not be a good choice for an dynamically evolving ad hoc environment since the last node in the protocols

computation would have to know the whole structure of the network. A very efficient protocol is also presented in [129]. In this recent work, a logical key tree structure is used to improve the scalability of the key agreement protocol. Any device can calculate the group key if it knows all the keys in its co path. This requirement makes the protocol quite expensive in storage memory that is critical for sensor networks. For these reasons, we believe that the simplicity and the limited memory requirements of GDH.3 protocol make it more suitable and applicable in sensor networks.

Moreover, the recent papers of Bresson et al. [50–52] were the first to present a formal model of security for group authenticated key exchange and the first to give rigorous proofs of security for particular protocols.

A performance analysis of Group Key Establishment Protocols is presented in [19, 20] which clearly shows the superiority of GDH.3 protocol in the number of transmitted data and exponentiation operations required. They show that the number of messages and exponentiations is linear to the number of the participants, while for all other protocols are of order  $n \log n$  or  $n^2$ .

**The GDH.3 Protocol.** GDH.3 protocol was presented in [170] together with GDH.1 and GDH.2 protocols. It is particularly suitable for environments which require the minimum computational effort from each group member. The protocol evolves in four stages and here we will present its elliptic curve analog. Suppose that every member in the group agreed on the use of the same elliptic curve parameters. The number of participants is  $n$  and we will denote by  $M_i$  the  $i$ -th participant.

1. In the first stage every group member  $M_i$  generates a random secret value  $k_i$ . The  $M_1$  participant selects a point  $P$  and sends to  $M_2$  the point  $Q_1 = k_1P$ . Then  $M_2$  sends to  $M_3$  the point  $Q_2 = k_1k_2P$  and so on until the protocol reaches member  $M_{n-1}$ . Notice here that the protocol must pass only one time from every participant.
2. Group member  $M_{n-1}$  computes the point  $Q_{n-1} = k_1k_2 \dots k_{n-1}P$  and sends it to all  $M_i$ , with  $i \in [1, n]$ .
3. In the third stage every group member  $M_i$ ,  $i \in [1, n - 1]$  computes a point  $G_i = k_i^{-1}Q_{n-1}$  and sends it to the last group member  $M_n$ .
4.  $M_n$  calculates the values  $k_nG_i$  and send them to the corresponding members  $M_i$ .

After these stages, every group member  $M_i$  can calculate the group key  $Q_n = k_1k_2 \dots k_nP$  by multiplying the value  $k_nG_i$  with its secret number  $k_i$ . Despite its efficiency, the disadvantage of GDH.3 protocol is that it does not offer symmetric operation, because all the participants in the protocol do not perform the same number of operations. If the number of the participants is large, then the computational effort in member  $M_n$  can be devastating for its energy. For this reason, we propose a new protocol which offers symmetric operation and requires that all devices perform the same number of operations.

## 4.4 Cryptographic Properties

There is a strong relationship between group membership events and group key management protocols, since these protocols need to provide key adjustment techniques stemming from membership changes. In a dynamic group the sensor group's view changes frequently since some members may want to join or leave the group so key establishment protocols have to handle membership events in order to guarantee the following cryptographic properties.

**Membership operations** include the following events:

- A Join Event occurs when a single member wants to join the existing group. The group key is updated to include the new member and the all participants are informed about the new key.
- A Leave Event occurs when a member wishes to leave the group, or is forced to leave it. The group key must be properly modified so that the departing participant can no longer use the old group key in order to encrypt/decrypt the group's communications.
- A Group Merge Event occurs when multiple potential members want to join an existing group. The keys of the two groups are merged so that all participates can communicate with each other using a common shared key.
- A Group Partition Event occurs when multiple members leave the group with or without forming their own subgroup. A new key must be established for each partitioned subgroup to guarantee secrecy.

Certainly, the easiest way to handle each group membership event is to re-execute the group key establishment protocol. However, in wireless sensor networks, the computation and communication overhead imposed every time the protocol is re-executed may severely exhaust the battery resources of the devices. Therefore, it is critical for each protocol to find incremental solutions that build upon the existing key that minimize modifications (in order to handle each membership event) and thus minimize the total energy spent (due to the extra computations and message exchanges). Group key management mainly includes activities for the establishment and the maintenance of a group key. Secure group communication requires scalable and efficient group membership with appropriate access control measures to protect data and to cope with potential compromises.

A secret key for data encryption must be distributed with a secure and efficient way to all members of the group. Another important requirement of group key management protocols is key freshness. A key is fresh if it can be guaranteed to be new. Moreover, the shared group key must be known only to the members of the group. Four important cryptographic properties must be encountered in group key agreement [129, 132]. Assume that a group key is changed  $m$  times and the sequence of successive keys is  $\mathcal{K}=\{K_0, \dots, K_m\}$ .

1. *Computational group key secrecy*: It guarantees that it is computational infeasible for any passive adversary to discover any group key  $K_i \in \mathcal{K}$  for all  $i$ .
2. *Decisional group key secrecy*: It ensures that there is no information leakage other than public blinded key information.
3. *Key independence*: It guarantees that a passive adversary who knows a proper subset of group keys can not discover any other of the remaining keys. Key independence can be decomposed into *forward secrecy* and *backward secrecy*. Forward secrecy guarantees that a passive adversary who knows a contiguous subset of old group keys cannot discover any subsequent group key. Backward secrecy guarantees that a passive adversary who knows a contiguous subset of group keys cannot discover preceding group key.

## 4.5 Secure Data Aggregation

Sensor networks provide a powerful solution to many monitoring problems. Nodes in the network may cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion, pollution [162]. However, due to limited resources and short battery life, operations on sensor nodes must be very carefully designed. Moreover, sensors' deployment is not always under direct control and sensors are often under the risk of physical attacks that can affect their security.

One of sensors' main tasks is to monitor data and provide it to a querier. Since data transmission is the very energy-consuming operation for sensor nodes, one should minimize the amount of transmitted data. For this purpose, aggregation techniques have been introduced in order to maximize battery lifetime. However, data aggregation requires that every node in the network is trusted. To address this problem, solutions have been proposed, as in [174] and [68], to provide integrity for data aggregation in sensor networks. Recently, it has been proposed a solution in [160], that make use of cryptographic hash functions and commitments to provide nodes the possibility of verifying that their measurements have been correctly computed in the aggregation process. However, in all these works nodes are able to gain information from data received during the aggregation, thus violating the privacy of data.

Several works instead have focused on ensuring privacy protection while performing aggregation operations. In [119] a solution was proposed to aggregate MACs in order to detect non-authorized inputs; while in [179], the problem of aggregating encrypted data was partially explored. A scheme for aggregation of encrypted data in sensor networks was proposed in [64]. Such a scheme is based on additively homomorphic probabilistic encryption schemes.

## 4.6 Our Algorithmic Solutions.

### 4.6.1 Key Management

In [71, 72], we propose two new distributed group key management protocols suitable for such energy constrained networks. The first protocol resembles GDH.3 in its first stage but instead of requiring direct communication among the participants of the network, it relies on short-range hop-by-hop propagation. To do this we employ a larger number of public key encryption and decryption operations per sensor node. The extra computations are simple for the devices to implement and are evenly distributed across the participants of the network, leading to fewer number of message exchanges. Moreover, our group key management protocol handles membership events like join or leave in order to provide forward and backward secrecy.

Secondly, in contrast to all previously mentioned methods, we present another group key establishment protocol that totally avoids constructing and maintaining a distributed structure that reflects the topology of the network, does not require any strong communication primitive and relies only on simple short-range hop-by-hop message exchanges. Our protocol is particularly suitable for dense ad-hoc networks where the topology is subject to frequent and unpredictable changes. We coordinate the devices in a distributed manner using minimum communication overhead through the use of a mobile agent (software, mobile code) that traverses the network. The protocol imposes minimum communication overhead as the mobile agent is of small size (it fits in a single TinyOS packet). The mobile agent moves *randomly* through the devices of the network and *no specific traversal strategy* (e.g. strictly sequential, in both directions) or global information (e.g. such as the structure of the network) is required. This software agent (mobile code) is responsible for constructing a common key among the participants of the group (by taking into account the contribution of all members of the group) and for the delivery of the established key to all participants.

We provide our basic design choices and explain how our agent-based protocol handles membership events in order to provide forward and backward secrecy. We analyze the performance of the protocol and also conduct a detailed comparative experimental study using a simulation environment. The analysis indicates the correctness and time-efficiency properties while the experimental study demonstrates the feasibility of implementing it in real sensor network devices. Overall our protocol manages to evenly distribute the energy dissipation among the sensor devices, leading to better energy balance. Our study highlights the advantages and disadvantages of our approach given the available technology and the corresponding efficiency (energy, time) criteria.

In our implementation we used the elliptic curve version of Diffie-Hellman problem [91]. The reason is that elliptic curve cryptosystems use much smaller keys than conventional, discrete logarithm based cryptosystems (an 160-bit key in an elliptic curve cryptosystem provides equivalent security with a 1024-bit key in a conventional cryptosystem). This fact makes elliptic curves the only reasonable choice for sensor networks, where the resources

are very limited. Moreover, recent research has shown that public key cryptography based on elliptic curves is feasible to be used in sensor networks [104, 114, 138].

We also conducted a comparative performance evaluation of our protocols with the GDH.3 protocol [170] for various network topologies using both experiments and simulation. The experimental study demonstrates the feasibility of implementing our protocols in real sensor network devices while the simulation study highlights the advantages and disadvantages of each approach given the available technology and the corresponding efficiency (energy, time) criteria. Overall our protocols manage to evenly distribute the energy dissipation among the sensor devices, leading to better energy balance.

#### 4.6.2 Key Pre-distribution

One problem that frequently arises is the establishment of a secure connection between two network nodes. There are many key establishment protocols that are based on Trusted Third Parties or public key cryptography which are in use today. However, in the case of networks with frequently changing topology and size composed of nodes of limited computation power, such as the ad-hoc and sensor networks, such an approach is difficult to apply. One way of attacking this problem for such networks is to have the two nodes share some piece of information that will, subsequently, enable them to transform this information into a shared communication key. Having each pair of network nodes share some piece of information is, often, achieved through appropriate *key pre-distribution* schemes. These schemes work by equipping each network node with a set of candidate key values, some of which shared with other network nodes possessing other keys sets. Later, when two nodes meet, they can employ a suitable key establishment protocol in order to locate shared values and used them for the creation of the communication key.

In our work we give a formal definition of collusion resistant key predistribution schemes and then propose such a scheme based on probabilistically created set systems. The resulting key sets are shown to have a number of desirable properties that ensure the confidentiality of communication sessions against collusion attacks by other network nodes. Here we present simple key set constructions that lead to key sets suitable for use in key pre-distribution schemes in Mobile Ad-Hoc Networks (MANETs). Our constructions are based on simple conditions that are sufficient to give rise to collections of sets that have two security properties: no intersection of key sets is contained in the union of a disjoint set of key sets and the stronger property no intersection of key sets shares a key with the union of a disjoint set of key sets. The former condition ensures that no key sets can collectively contain the entire key set of another set of nodes while the latter condition ensures that when a group of nodes need to communicate, other nodes cannot collectively locate a key which is contained in all the key sets of the nodes. These conditions are paramount to the safety of the cryptosystem used in that in the event that some nodes become compromised and collude towards the discovery of key values of other nodes. In addition, we present some set system constructions based on special polynomial classes

(the Barrington, Beigel, and Rudich polynomials) as well as general polynomials. Our aim is to link deterministic key pre-distribution with the theory of set family construction methods showing that the mathematical properties possessed by the sets of such families lead to key sets with the aforementioned desirable properties.

### 4.6.3 Secure Data Aggregation

In [85], we describe an efficient solution to provide both integrity and privacy protection while performing aggregation operations (such as aggregated sum) in sensor networks. To meet the first goal (data integrity), we refer to the scheme proposed in [68]. Such a scheme is proved to be optimally secure in the data aggregation, i.e., an adversary cannot alter the aggregation result by tampering with the aggregation process. In other words, a malicious node cannot affect the computation other than directly injecting measured data. We cope with the second requirement (privacy protection) by resorting a homomorphic encryption scheme presented in [64].

Data integrity is guaranteed by the computation of a cryptographic commitment structure (based on a cryptographic hash function), which allows sensor nodes to verify that their contributions have been correctly computed in the aggregation. An upper bound on allowable data values, say  $r$ , is introduced to restrict the range of accepted values, e.g., the sensed temperature cannot be higher than  $50^{\circ}\text{C}$ . Furthermore, this value is used to calculate a complementary aggregate, say *complement*. When computing the aggregate *sum*, the querier enforces the constraint that  $sum + complement = nr$ , where  $n$  is the number of nodes in the network. Finally, each nodes maintains a progressive value, *count*, to indicate the number of vertices in the subtree rooted at it. This value is also carried along the aggregation.

The commitment is built by computing a cryptographic hash function over the concatenation of the count, the aggregated value, the aggregated complement, and all the commitments received from the children. We assume that the hash function used to build the commitment is collision-resistant, so that the adversary is computationally prevented to modify any data after the final commitment values have reached the root. A commitment tree is built following the aggregation tree. However, rather than mirroring the aggregation tree, in [68] it is proposed a way to balance the commitment tree, by enforcing the constraint of performing an aggregation only if it results in commitment tree that is complete and binary. Therefore, the result structure is a commitment forest consisting of complete binary commitment trees, whose height is at most  $\log n$ . This technique allows to reduce the congestion to  $O(\Delta \log^2 n)$ , where  $\Delta$  is the maximum degree of any node in the aggregation tree.

Moreover, we want to ensure the privacy protection of aggregated data. In other words, an eavesdropper should not gain any additional information from the aggregation of data. Moreover, internal nodes should be prevented to see values they receive from their children. As previously discussed, we are interested in an efficient end-to-end privacy. To this aim,

we use an additively homomorphic encryption scheme. Due to resource constraints, public-key schemes are considered infeasible for sensor networks and we refer to the symmetric scheme proposed in [64]. This scheme is proved to be perfectly secure.

We use a probabilistic encryption scheme, relying over a keystream  $k$ , generated by using a stream cipher keyed with the node's secret key (say  $s_i$ ) and a unique message id. This  $s_i$  is pre-computed and shared between the base station and the node  $i$ , while the message unique id is included in the aggregation query. Therefore, the key  $k_i$ , to be used for encryption, is derived at run-time. In our implementation, we have used *RC5* to obtain the key  $k_i$ , shared between the node  $i$  and the base station. Actually, *RC5* was chosen since it was freely available within the *TinySec* library for *TinyOS* [122].

When the aggregation is completed, each node can verify that its value has been correctly inserted in the aggregation. The verification is performed as follows. First of all, the commitment values are disseminated to all the nodes. Then, nodes need to know the label of all the siblings of each node on the path towards the root of the commitment tree. These values are easily disseminated too, since commitment trees are binary and complete (for more details and for the proof of correctness of such mechanism, see [68]). Once nodes have the required inputs, they may re-compute labels and compare them against disseminated labels. If any node  $i$  fails in the verification, an alarm is raised, otherwise it sends to the querier the authentication code  $MAC_{K_i}(OK\|Na)$ , where  $OK$  is a unique message identifier,  $Na$  is a nonce which the base station disseminates when starting the query, and  $K_i$  the key shared with the querier. Authentication codes are sent over the aggregation tree and XOR-ed along the paths. Therefore, the base station receives the XOR of all the authentication codes. Since it knows all the used keys, it re-computes and XOR-es the authentication codes. If the result matches the received value, the base station is finally guaranteed of the integrity of the whole aggregation operation.

Aggregation performed within our scheme is ensured to be secure (in terms of integrity check) and also secret (in terms of privacy protection). In fact, an internal node performing aggregation is neither able to tamper nor to recover values of other nodes.

In order to evaluate the feasibility of our scheme, we set up a simulation environment where we run some experiments. We considered the most diffused platform for sensor nodes: *mica/mica2* motes. Hence, we implemented our scheme within the *TinyOs* [4] environment. To run our simulation, we used the *Tossim* simulator [3], which is a discrete-events and highly reliable simulator for *TinyOs* applications. *Tossim* is able to execute the same identical code which is to be run on real nodes. Moreover, it allows external applications to connect through *TCP* sockets and monitor and act on the simulation. For more details about *Tossim*, we refer to [131].

We ran our experiments over a network with 100 nodes, with random topology. An aggregation sum was simulated by using our scheme. We first measured running times and power consumption required for building the aggregation tree for each node. The average running time was 184 ms while the average power consumption was around 233 mJ. Differences among nodes depend on network topology: leaf nodes perform a smaller

quantity of operations and transmit less messages; while, internal nodes may have a different number of children implying a greater number of operations and messages exchanged. The second experiment was aimed to evaluate performances of the aggregation. The third test evaluated the verification of aggregation correctness. This operation is the longest one, since different values have to be propagated to allow nodes to verify that their value was correctly computed. Such test shows an average running time of 259 ms and a battery loss of 292 mJ. Notice that, though being a little heavier, this operation is performed only once, at the end of the computation.

We also considered the cost of the encryption. To this aim, we compared our scheme with the equivalent version without encryption management, i.e., all values are transmitted as plaintext. We witnessed that the derivation of the keystream  $k$ , the run-time encryption (consisting of a simple modular addition), and the final decryption performed by the querier have a small impact which is not practically relevant both on the running time and on the power consumption.

The above results show that our scheme is efficient enough to be used in real world sensor applications that need to securely and privately aggregate data in sensor networks.

## 5 Commitment

Commitment schemes are arguably among the most important and useful primitives in Cryptography and Security. Intuitively a commitment scheme can be seen as the digital equivalent of a sealed envelope. If a party  $A$  wants to commit to some message  $m$  she just puts it into the sealed envelope, so that whenever  $A$  wants to reveal the message, she opens the envelope. Clearly, such a mechanism can be useful only if it meets some basic requirements. First of all the digital envelope should *hide* the message: no party other than  $A$  should be able to learn  $m$  from the commitment (this is often referred in the literature as the hiding property). Second, the digital envelope should be *binding*, meaning with this that  $A$  can not change her mind about  $m$ , and by checking the opening of the commitment one can verify that the obtained value is actually the one  $A$  had in mind originally (this is often referred to as the binding property). These two properties make commitments very useful in a wide range of cryptographic applications such as zero-knowledge protocols, multi-party computation, digital auctions and electronic commerce.

A *commitment scheme* is a primitive to generate and open commitments. More precisely a commitment scheme is a two-phase protocol between two probabilistic polynomial time algorithms **sender** and **receiver**. In a first stage (called the *commitment* phase) **sender** commits to a message  $m$  using some appropriate function  $\text{Com}$  which takes as input  $m$  and some auxiliary value  $r$  and produces as output a value  $c$ . The value  $c$  is sent to **receiver** as a commitment on  $m$ . In the second stage (called the *decommitment* phase) **sender** “convinces” **receiver** that  $c$  is actually a valid commitment on  $m$  (if **receiver** is not convinced, it just outputs some special string). A commitment scheme with this form is called non-interactive since both stages require only one message from **sender** to **receiver**. The requirements that we make on a commitment scheme are the following ones. First, if both **sender** and **receiver** behave honestly, then at the end of the decommitment phase **receiver** is convinced that **sender** had committed to bit  $m$  with probability 1. This is often referred to as the *correctness* requirement. Second a cheating **receiver** can not guess  $m$  with respect to any other legal message  $m'$  with probability significantly better than  $1/2$ . This is the so-called *hiding* property. Finally, a cheating **sender** should be able to open a commitment (i.e., to decommit) with both  $m$  and  $m' \neq m$  only with very small (i.e., negligible) probability (this is the *binding* property). Each of the last two properties (i.e., hiding and binding) can be satisfied unconditionally or relatively to a computational assumption. In our context (i.e., where only two parties are involved) this immediately implies that one can not hope to build a commitment scheme where both the hiding and the binding properties hold unconditionally. Unconditionally binding commitment schemes have been constructed under the sole assumption that one-way functions exist [143] and in that construction an initial message of the receiver is required. It is known how to construct non-interactive unconditionally binding commitment schemes by using any one-to-one one-way function and in [24], the authors show how to construct non-interactive commitments based on one-way functions and derandomization techniques.

Constant-round unconditionally hiding commitment schemes have been constructed under the assumption that collections of claw-free functions [111] or collision resistant hash functions [118] exist. Recently in [117] it is shown how to construct unconditionally hiding commitment schemes from any regular one-way function, but unfortunately, these schemes are not constant round.

Since commitment schemes are very useful primitives they are often used as building blocks to construct larger protocols. In this sense it is often the case that the two basic requirements described above turn out to be insufficient. For this reason commitment schemes with additional properties have been proposed. Here we highlight on some of these constructions, other, more directly related to our results, will be discussed in the next section.

A *trapdoor commitment scheme* (sometimes also called *chameleon commitment*), is a commitment scheme with associated a pair of public and private keys (the latter also called the *trapdoor*). Knowledge of the trapdoor allows the sender to open the commitment in more than one way (this is often referred to as the *equivocality* property). On the other hand, without knowledge of the trapdoor, equivocality remains computationally infeasible. When the commitments computed by means of a trapdoor are distributed exactly as real commitments then the trapdoor commitment scheme is unconditionally hiding. Trapdoor commitments have been shown to exist under the assumption that one-way functions exist [98].

An *extractable* commitment (also known as commitment scheme with *extractability*) is a commitment scheme where we allow the existence of a secret key whose knowledge permits to *extract* the message stored in the commitment. At the same time, without knowledge of the secret key, the message remains (computationally) hidden.

Finally a *universally composable commitment* is a commitment scheme with the very useful property that – informally – even if one concurrently composes it with any other protocol, the security of the commitment scheme is preserved. Universal composability is a very strong notion, which, for the case of commitment schemes, seems to require concurrent non-malleability and extractability.

The first construction of a universally composable commitment scheme has been presented in [62] and it has been later improved in [63] and in [83].

## 5.1 Our algorithmic solution: Hybrid trapdoor commitments

In [66], we introduce the notion of a hybrid trapdoor commitment scheme. Informally a hybrid trapdoor commitment scheme is a general commitment primitive with commitment parameters generation algorithms **HGen** and **HTGen**. If the commitment parameters are obtained as the output of **HGen** then the resulting scheme is an unconditionally binding commitment scheme, while if the parameters are generated by **HTGen** the produced scheme is actually a trapdoor commitment scheme. Moreover, as for mixed commitments, no polynomially bounded adversary, taking as input only the (public) commitment pa-

rameters, should be able to tell the difference between keys generated from **HGen** and keys produced by **HTGen**.

Notice that the notion of hybrid trapdoor commitment may look very similar to that of mixed commitment. There is a crucial difference however. Depending on the way the parameters are generated a mixed commitment can be either an extractable commitment or a trapdoor commitment. In our case, on the other hand, we require only that the commitment is either unconditionally binding or a trapdoor commitment scheme.

Mixed commitments have been introduced to construct universally composable commitments and indeed Damgård and Nielsen proved that it is possible to construct a universally composable commitment from a mixed commitment where the number of  $E$ -keys over the total number of keys is negligible and that the number of  $X$ -keys over the total number of keys is almost 1. Interestingly, a recent result by Damgård and Groth [83] shows that universally composable commitments imply key exchange and, when implemented in the shared random string model, they imply oblivious transfer. Therefore it seems unlikely that universally composable commitments can be implemented from one-way functions only. In [66], on the other hand, we show that hybrid trapdoor commitments can be constructed from any one-way function.

To improve on efficiency we then turn our attention to specific number-theoretic constructions and we propose three different implementations. The first one is very efficient and relies on the Decisional Diffie-Hellman assumption. The remaining two are based on Paillier's [154] Decisional Composite Residuosity Assumption (DCRA for brevity). Interestingly the DCRA-based solutions enjoy the extractability property and turns out to be mixed commitment schemes. In particular our first DCRA-based solution is very efficient and it is actually slightly more efficient than the two implementations proposed in the literature. Our construction of hybrid trapdoor commitments that are based on the Diffie-Hellman assumption are much more efficient than the known constructions of mixed commitments.

**Stronger extensions.** We study some stronger extensions of hybrid trapdoor commitments. In particular we show how to build hybrid simulation-sound trapdoor commitments and hybrid multi-trapdoor commitments from the sole assumption that one-way functions exist. Moreover, we present some practical implementations based on number-theoretic assumptions.

In this paper we show that multi-trapdoor commitment schemes are actually equivalent to digital signatures which are secure with respect to *generic* chosen message attack. Informally in a generic chosen message attack the adversary can obtain signatures only on a list of messages chosen *before* the public key of the signer is published. This is clearly a weaker notion with respect to the standard one where the adversary is allowed to choose the messages adaptively. Since SSTCs are actually equivalent to standard secure signatures, from a practical point of view, our result further clarifies why the known (practical) implementations of multi-trapdoor commitments are more efficient than the

corresponding implementations of SSTC.

**Techniques.** The construction of a hybrid trapdoor commitment scheme, consist in producing a scheme that is a trapdoor commitment scheme for some commitment parameters while it is an unconditionally binding commitment scheme for other commitment parameters. We stress that the two distributions of the commitment parameters are indistinguishable. Once we achieve this result, we combine the basic hybrid trapdoor commitment scheme with a tag-based simulation-sound trapdoor (resp., multi-trapdoor) commitment scheme. This is based on a parallel execution of the two commitment schemes, thus obtaining a hybrid tag-based simulation-sound trapdoor (resp., multi-trapdoor) commitment scheme. We finally stress that the results of [83] actually achieve hybrid trapdoor commitments based on one-way functions only, even though this is not explicitly formalized and claimed in their work.

**Applications.** The main contribution of [66] is to show how to use the different variants of hybrid trapdoor commitments to achieve the following results.

1. Using hybrid trapdoor commitments we show how to construct 3-round concurrent zero-knowledge proof systems, in the shared random string model, for all  $\mathcal{NP}$  languages. We give a construction based on the existence of any one-way function and an efficient construction that is based on the DDH assumption. These results improve the computational soundness achieved in a previous result by Damgård [81] in the sense that ours are actually zero-knowledge proofs rather than zero-knowledge arguments.
2. Using either hybrid SSTC or hybrid multi-trapdoor commitments we show how to construct an unbounded simulation-sound zero-knowledge proof system in the common reference string model. This improves the recent results of [105, 137] where similar results were presented for unbounded simulation-sound zero-knowledge arguments (rather than proofs).

## 5.2 Our algorithmic solution: Non-Malleable Commitments

In some application scenarios, one wants to be able to guarantee that an adversary  $A$ , playing as a receiver in an execution in which a honest committer commits to message  $m$ , is not able to commit to a related value  $\tilde{m}$  to a honest receiver in another execution in which  $A$  plays as a committer. It is easy to observe that the hiding property does not guarantee this extra property. This type of adversary is called a *man-in-the-middle* adversary (as the adversary plays in between two honest players). Commitment schemes secure with respect to these attacks are called *non-malleable* commitments.

Two notions of non-malleable commitments have been considered in the literature. A commitment scheme that is *non-malleable with respect to commitment* guarantees that

no polynomial-time man-in-the-middle adversary  $A$  can *commit* to a message  $\tilde{m}$  that is related to the message  $m$  committed by the honest committer. Instead, a commitment scheme that is *non-malleable with respect to decommitment* (also known as non-malleable with respect to opening) guarantees that after the commitment phase, no polynomial-time man-in-the-middle adversary  $A$ , observing the decommitment to  $m$  of the honest committer, can *decommit* its commitment to a message  $\tilde{m}$  that is related to  $m$ .

The need to design non-malleable cryptographic primitives has been first pointed out in the seminal paper by Dolev, Dwork and Naor [93] who also gave constructions for non-malleable encryption, non-malleable zero-knowledge proofs and non-malleable commitments. The constructions for non-malleable zero-knowledge and non-malleable commitments of [93] required  $O(\log k)$  rounds, where  $k$  is the security parameter. Subsequently, assuming the existence of some trusted parameters, non-malleable commitments were constructed in [80, 83, 90, 101]. The commitment schemes in [80, 101] are non-malleable with respect to decommitment.

The first constant-round non-malleable commitment scheme in the standard model (i.e., without setup assumptions) has been given by Barak [23] under the assumption of the existence of trapdoor permutations and hash functions that are collision resistant against sub-exponential-time adversaries. Pass and Rosen [157] reduced the assumption to the existence of hash functions that are collision resistant against polynomial-time adversaries and gave two different schemes: one that is non-malleable with respect to commitment and one that is non-malleable with respect to decommitment.

More recently, Pass and Rosen [156] have considered *concurrent* man-in-the-middle attacks (cMiM attacks). In such an attack, the adversary is not restricted to be active in only one execution as receiver and in only one execution as committer. Instead the adversary can open any polynomial number of sessions. Instead the adversary can be active in any polynomial number of executions as a receiver and as a committer. A commitment scheme that is secure against cMiM attacks is called *concurrent non-malleable*. As before, we can have two notions of concurrent non-malleable commitment schemes: concurrent non-malleable commitment schemes with respect to commitment and with respect to decommitment. Pass and Rosen [156] showed that the constant-round scheme that is non-malleable with respect to commitment of [157] is actually *concurrent* non-malleable with respect to commitment. This implies that security is guaranteed if the commitments, but not the decommitments, are concurrently executed. They leave as an open problem the construction of commitment schemes that are non-malleable with respect to decommitment. We stress that the concurrent non-malleability of the construction of [156] relies on the assumption that commitments and decommitments do not overlap in time. We retain this assumption in our constructions. We stress that this assumption still allows one to use such commitment schemes in some of their main applications as auctions and voting (where first all parties send their hidden bids/votes, and only in a second stage they are decommitted).

We also remark that the recent work of Barak, Prabhakaran, and Sahai [25] could be

used to construct concurrent non-malleable commitments. Unfortunately the [25] protocol requires a poly-logarithmic round complexity that is worse than the round complexity of the protocol of [156]. The recent work of Barak, Prabhakaran, and Sahai [25] could be used for non-malleable commitments but it would require a poly-logarithmic round complexity.

**Our results.** The main result of [152] consists in the construction of a constant-round commitment scheme that is concurrent non-malleable with respect to *both* commitments and decommitments. This implies that security is preserved when polynomially many commitment phases are concurrently executed and when polynomial many decommitment phases are concurrently executed. This solves an open problem of [156]. We follow [156] in that concurrent non-malleability is guaranteed only if commitments and decommitments do not overlap in time. Notice that a construction of a constant-round concurrent non-malleable commitment in which commitments and decommitments can be interleaved would also give constant-round concurrent (non-malleable) zero knowledge, solving one of the major open problems in the area. The [25] construction could be used with interleaving commitments and decommitments, but as we stressed above the round complexity is not constant.

Our construction builds and extends multiple techniques. In particular, our construction uses the perfect NMZK argument of knowledge of [156–158] but in a critically different manner in which it is used in [156, 157]. Indeed, whereas in [156, 157] the statistical NMZK argument of knowledge is simply combined with a (potentially malleable) commitment scheme and a signature scheme, to achieve security in a concurrent setting we also employ a technique by Feige [99] and a more sophisticated rewind technique. Furthermore, the simulator used by [156, 157] works in a straight-line fashion including non-black-box techniques. Our result, instead, combines the straight-line simulation with a new rewinding simulation that still avoids the well known problems of using rewinds in concurrent settings [95]. Our approach also includes and extends some of the techniques developed for building concurrent NMZK in the bare public-key model [153].

We also show that it is possible to have non-malleable commitments with respect to commitments that are not statistically binding by giving a constant-round construction. This is in contrast with [157] that states that this notion only makes sense with respect to statistically binding commitments.

### 5.2.1 Applications to auctions

The sealed-bid auction is one of the most important type of auctions. Here we have  $n$  parties each one of which writes his bid in a envelope, seals it and sends it to the auctioneer. Once all bids have been received, or once the deadline has passed, the auctioneer opens the envelopes and the highest bidder wins the auction. It has been often suggested that commitments might be used for a digital implementation of a sealed-bid auction. Specifically, each party commits to its bid and once the deadline has passed, commitments are

open. It is then important that commitments are concurrently non-malleable. Indeed, first of all, one would like to guaranteed that the bids are independent from each other: the malleability property. At the same time, one must guarantee that this independence property also holds for the case in which the bidders are not synchronized but rather each of them starts his commitment independently from the other.

## 6 Secret sharing

In threshold secret sharing, a secret is shared amongst  $n$  participants and can only be reconstructed by more than  $t$  of them together. Such schemes have found numerous applications such as key escrow, distributed file storage and distributed computation. However, in threshold schemes, all participants play an equal role and cannot be distinguished according to trust or authority, whereas in many real-life situations, hierarchies are required. Consider an example from the military. Let the secret be the “nuclear button” of a country and suppose that it can only be accessed by two ministers, or a minister and a general, but not by two generals. In this case, a 2-out-of- $n$  threshold scheme is clearly not suitable, since any two generals could pool their shares together to bypass access control. Secret sharing schemes that take into account hierarchies were the first non-threshold schemes considered in the literature.

We investigate the *multiplicativity* of hierarchical schemes. Multiplicativity allows participants, holding shares of two secrets  $s$  and  $s'$ , to privately compute shares of the product  $ss'$  without revealing the original secrets. *Strong* multiplicativity further guarantees that in the presence of an adversary, honest participants can still compute such shares. A simple solution for multiplication of secrets exists for the Shamir threshold scheme. In general, however, it is not known how to efficiently construct a strongly multiplicative linear secret sharing scheme (LSSS) with the desired non-threshold access structure. Thus, we tackle a more specific problem and prove strong multiplicativity for a class of access structures.

Strongly multiplicative schemes turn out to be useful even outside the context of multiplying secrets, since they are resistant to errors in shares. Although in any LSSS with a  $\mathcal{Q}_3$  access structure, the secret is uniquely determined even if the shares submitted by corrupted participants contain errors, it is not known how to locate such errors efficiently. An efficient secret reconstruction algorithm is only known for strongly multiplicative LSSS. This implicit “built-in” verifiability makes strongly multiplicative schemes an attractive building block for secure distributed computation.

In [123], we have proposed two different solutions for obtaining strong multiplication in the hierarchical threshold setting. Some of our constructions achieve robustness against the strongest possible adversary. These schemes are not ideal but have a reasonable information rate for hierarchies with few levels. We also propose ideal constructions which are strongly multiplicative under somewhat stronger, but still feasible assumptions. In particular, we have proposed a modification that improves the multiplication properties of the scheme. Our results are not tight and it is possible that better bounds can be obtained by more careful analysis. However, we have proven the modified scheme to be optimal with respect to the most crucial property—the number of required top-level participants.

The modified ideal scheme has a randomized identity allocation strategy with failure probability  $p = \Theta(1/q)$ , which is a safe bet for large field sizes  $q$ . Still, for the original conjunctive and disjunctive constructions, the author also proposed a deterministic allocation

strategy, which has zero failure probability if the field is sufficiently large. The strategy allocates identities in a monotone fashion, so participants from higher levels get “smaller” field elements. It would be interesting to verify if the deterministic strategy also applies for the new scheme.

## 7 Distributed Oblivious Transfer

Introduced by Rabin in [161], and subsequently defined in different forms [47, 97], the *oblivious transfer* (OT, for short) has found many applications in cryptographic studies and protocol design. Basically, such a protocol enables one party to transfer knowledge to another in an “oblivious” way. Rabin’s definition, for example, enables a Sender to transmit a message to a Receiver in such a way that the Receiver with probability  $\frac{1}{2}$  gets the message while, with the same probability, she does not, and the Sender does not know which event has occurred. Rabin showed how this transfer can be used in order to exchange secrets, and subsequently several other researchers have shown some useful applications of this concept. The protocol proposed by Rabin was later strengthened in [100].

The second OT definition was given in [97]. In this form, the Sender has two secrets and the Receiver is interested in one of them. After the execution of the protocol, the Receiver gets the secret she wishes to recover, obtaining at the same time no information on the other, while the Sender does not know which secret the Receiver has recovered. The author of [97] showed a first application to signing contracts.

The last and more general form of OT was introduced in [47], under the name of *all-or-nothing Disclosure of Secrets*, even if the same concept was born in an artificial intelligence context [180], under the name of *multiplexing*. Here the Sender has  $n$  secrets and the Receiver is interested in one of them. After the execution of the protocol, the Receiver gets the secret she wishes to recover, obtaining at the same time no information on the others, while the Sender does not know which secret the Receiver has recovered.

All these forms were shown to be equivalent [48, 49, 79], and Kilian in [127] showed that the OT is a complete primitive, in the sense that it can be used as a building block for any secure function evaluation (multi-party computation).

A variety of slightly different definitions and implementations can be found in the literature as well as papers addressing issues such as the relation of the OT with other cryptographic primitives, the assumptions required to implement such a concept, reductions among “more complex” forms of OT to “simpler ones” and applicative environments (e.g., [32, 49, 78, 79, 84, 86, 87, 92, 109, 112, 146], just to name few examples).

In the literature there are many papers that address problems related to 1-out-of- $n$  distributed oblivious transfer. In [29], for example, the authors show how to distribute a function between several Servers, in such a way that a user can compute the function by interacting with the Servers; the Servers cannot find out which value of the function the user computes, but the user can compute the function in *more than* one point. Another very close area is represented by PIR (Private Information Retrieval) schemes, introduced in [75]. A PIR scheme enables a user to retrieve an item of information from a public accessible database in such a way that the database manager cannot figure out from the query which item the user is interested in. However, the user can get information about more than one item. On the other hand, in SPIR (Symmetric Private Information

Retrieval) schemes [108], the user can get information about *one and only one* item, i.e. even the privacy of the database is considered. In PIR and SPIR schemes, the emphasis is placed on the *communication complexity* of the interaction of user and Servers. Notice that a SPIR Scheme can be seen as a *communication-efficient* 1-out-of- $n$  oblivious transfer scheme and the protocols given in [108] represent the first 1-round distributed implementation of 1-out-of- $n$  oblivious transfer. However, the main differences between the model we are going to consider and (information theoretic) SPIR schemes are that in SPIR schemes the Receiver communicates with  $k$  out of  $k$  Servers in order to retrieve an item while in our setting the Receiver can choose  $k$  out of  $m$  Servers, where  $k \leq m$ . Moreover, in SPIR schemes, the security of the Sender against *coalitions* of Receiver and Servers is not of concern. Other PIR papers of interest, for the distributed OT scenario we consider, are [31, 89, 107].

Rivest’s model in [164], where a trusted initializer participates *only* during the set up phase of the system (see also [42]), provides a very close setting to the one described in [145]. A paper which deals with distributed oblivious transfer implementations, close to the setting introduced in [145] (but not unconditionally secure) is [173]. Finally, unconditionally secure distributed oblivious transfer schemes for general access structures have also been studied in [139].

In our constructions we use secret sharing schemes. Secret sharing schemes were introduced in 1979 by Blakley [35] and Shamir [166], and have been extensively studied during the last years. The reader can find an introduction in [172] and references to the literature in [171].

## 7.1 Applications

Distributed Oblivious Transfer Protocols have several interesting applications and connections with other cryptographic protocols. Let us quickly describe some of them.

**Privacy Preserving Auctions and Mechanism Design [146].** The notion of DOT was introduced in [145] to improve the protocol of [146]. More precisely, in that protocol, there are three parties: an auctioneer, many bidders, and an agency supporting the auction. The auctioneer advertises the auction and its rules. The bidders submit their bids in “encrypted form” to the auctioneer, and the auctioneer, with the help of the agency, can compute the winner of the auction in such a manner that the privacy of the bidders (i.e., non-essential information about their own bids) is preserved. The weak point of the protocol is that if *the auctioneer and the agency collude*, then the privacy of the bids is lost. In order to strengthen the protocol, the agency can be split in two parts: a central agency, that operates only in a set up phase, and  $m$  Servers, with which the auctioneer communicates in order to compute the auction. In this case the auctioneer needs to collude with  $k$  out of the  $m$  Servers in order to violate the privacy of the bidders. The impossibility result for one-round  $(k, m)$  protocols private against a coalition of  $k - 1$  Servers and the Receiver in this setting means that the highest degree of privacy sought for in [146] with

this approach cannot be achieved. On the positive side, the two-round protocols can be applied to this framework but the *communication pattern* changes and some more details must be taken into account.

**Symmetric Private Information Retrieval.** Distributed Oblivious Transfer protocols have connections with symmetric private information retrieval schemes [108]. A PIR scheme [75] enables a user to retrieve an item of information from a public accessible database in such a way that the database manager cannot figure out from the query which item the user is interested in. However, the user can get information about more than one item. On the other hand, in SPIR (Symmetric Private Information Retrieval) schemes [108], the user can get information about *one and only one* item, i.e. even the privacy of the database is considered. In PIR and SPIR schemes, the emphasis is placed on the *communication complexity* of the interaction of user and Servers. Therefore, a SPIR Scheme can be seen as a *communication-efficient* 1-out-of- $n$  oblivious transfer scheme. The main differences between the model we have considered and (information theoretic) SPIR schemes are that in SPIR schemes the Receiver communicates with  $k$  out of  $k$  Servers in order to retrieve an item while in our setting the Receiver can choose  $k$  out of  $m$  Servers, where  $k \leq m$ . This property is useful since it guarantees a sort of *Robustness* for the SPIR scheme, in the sense that even if some Server crashes, the item can still be retrieved by the user by means of the other available ones. Hence, *a communication-efficient threshold DOT scheme realizes a robust SPIR scheme*. Another important difference is that in information theoretic PIR and SPIR schemes the database is *replicated* among the Servers. Hence, every Server knows the content. In our model only a  $k$ -subset of Servers can reconstruct the database.

Another interesting relation of the DOT model we have studied is with information theoretic PIR schemes with preprocessing [31]. The set up phase performed by the dealer can be seen as the preprocessing stage performed by the database owner in [31]. The combinatorial constructions we have shown are communication-inefficient but they require trivial computation for the Servers, once the scheme has been set up.

Just to emphasize the connection, notice that, using the DOT constructions we can set up a robust unconditionally secure symmetric private retrieval scheme. The database  $\mathcal{D}$  is simply distributed by the owner among  $m$  Servers, according to the  $(k, m)$ -DOT scheme for  $n$  secrets.

## 7.2 Our Algorithmic Solutions

In [41] we have studied unconditionally secure distributed oblivious transfer protocols. Let us define the model we are going to consider. We assume that the Sender holds  $n$  secrets and the Receiver is interested in one of them. Hence, we are concerned with a 1-out-of- $n$  distributed oblivious transfer.

In the distributed setting, the Sender  $\mathcal{S}$  does not directly interact with the Receiver  $\mathcal{R}$  in order to carry out the oblivious transfer. Rather, he *delegates*  $m$  Servers to accomplish

this task for him.

SET-UP PHASE. Let  $m$  and  $k$  be two integers such that  $1 < k \leq m$ . Let  $S_1, \dots, S_m$  be  $m$  Servers holding programs  $P_1, \dots, P_m$ , respectively. The Sender  $\mathcal{S}$  generates  $m$  data  $D_1, \dots, D_m$ , and, for  $i = 1, \dots, m$  sends, *in a secure way*, the data  $D_i$  to Server  $S_i$ .

OBLIVIOUS TRANSFER PHASE. The Receiver  $\mathcal{R}$  holds a program  $R$  and a sequence of random bits  $D_R$  which enables her to interact with a subset  $\{S_{i_1}, \dots, S_{i_k}\}$  of the Servers at her choice. Using the knowledge acquired by exchanging messages with the Servers,  $\mathcal{R}$  recovers the secret in which she is interested, but receives no information on the other secrets. At the same time, no subset of  $k - 1$  Servers, gains any information about the secret she has recovered<sup>1</sup>. More precisely, a distributed  $(k, m)$ -DOT- $\binom{n}{1}$  must guarantee:

1. **Correctness.** If the Receiver gets information from  $k$  out of the  $m$  Servers, she can compute the secret.
2. **Receiver's Privacy.** No coalition of less than  $k$  Servers gains information about which secret the Receiver has recovered.
3. **Sender's Privacy w.r.t.  $k - 1$  Servers and the Receiver.** A coalition of the Receiver with  $k - 1$  dishonest Servers does not get any information about the  $n$  secrets.
4. **Sender's Privacy w.r.t. a "Greedy" Receiver.** Given the transcript of the interaction with  $k$  Servers, the Receiver should gain information about at most a single secret, and no information about the others. This property should be satisfied even if the Receiver, once has computed a secret, colludes with  $k - 1$  dishonest Servers.

A sequence of programs (assigned to the Sender, the Servers and the Receivers) is said to be private for  $(k, m)$ -DOT- $\binom{n}{1}$  if all the above conditions are satisfied.

A sequence of programs is said to be a *strong*  $(k, m)$ -DOT- $\binom{n}{1}$  if it is correct, private and satisfies the following:

1. A dishonest  $\mathcal{R}$ , interacting with a subset  $Y$  of  $k$  Servers, can recover a secret. Then, even if she colludes with a subset  $X$  of  $k - 1$  dishonest Servers, by putting together the information they possess and the transcript of the previous interaction, the coalition does not get any information about other secrets.

Notice that, in [145], properties 3. and 4. are only guaranteed with respect to a threshold  $t$  and a threshold  $\ell$ , respectively, which should be as close to  $k$  as possible.

We study *unconditionally secure distributed oblivious transfer protocols*, introduced in [145] in order to strengthen the security of protocols designed for electronic auctions [146].

Let  $W = W_1 \times \dots \times W_n$  be the set of all possible sequences of  $n$  secrets, and let  $T = \{1, \dots, n\}$  be a set of  $n$  indices. We will represent the data  $W, T, D_1, \dots, D_m$ , and

---

<sup>1</sup>Along the same line of [145], we assume the existence of an external mechanism which guarantees that the Receiver can contact no more than  $k$  Servers. This issue is independent of the distributed oblivious transfer scheme. The reader is referred to [145] for some techniques to solve the problem.

$D_R$ , by the random variables  $\mathbf{W}, \mathbf{T}, \mathbf{D}_1, \dots, \mathbf{D}_m$ , and  $\mathbf{D}_R$ . A basic tool used for proving our results is Information theory and, specifically, the *entropy* function, denoted by  $H(\cdot)$ . The reader is referred to [76] for details.

As for the server memory storage, in the above model, we have shown that in any correct and private  $(k, m)$ -DOT- $\binom{n}{1}$  scheme, for any subset  $X \subseteq \{1, \dots, m\}$ , where  $1 \leq |X| \leq k$ , it holds that each server has to store at least  $H(\mathbf{W})$  bits.

Furthermore, in order to properly execute a Setup phase, the Sender needs at least  $kH(\mathbf{W})$  random bits, since if  $|X| = k$ , then  $H(\mathbf{D}_1, \dots, \mathbf{D}_m) \geq H(\mathbf{D}_X) \geq kH(\mathbf{W})$ . Such lower bound is tight since we provide a DOT protocol using exactly the amount of randomness described.

When we consider the communication complexity of correct and secure  $(k, m)$ -DOT- $\binom{n}{1}$  schemes, we can show that:

- **Interaction Receiver-Server.** The Receiver and a Server need to exchange at least  $H(\mathbf{W}_i)$  bits.
- **Interaction Receiver-Servers.** The Receiver and Servers  $S_X$ , where  $|X| = k$ , need to exchange at least  $k \cdot H(\mathbf{W}_i)$  bits.

We have considered one-round DOT schemes, i.e., schemes in which the interaction between the Receiver and Server  $S_j$  is given by a query  $Q_j$ , sent by the Receiver, and an answer  $A_j$ , sent by  $S_j$ . Hence, for any  $X \subseteq \{1, \dots, m\}$ , the transcript  $C_X = (Q_X, A_X)$ .

In this case, we have shown that in any one-round protocol for  $(k, m)$ -DOT- $\binom{n}{1}$ , for any subset of  $k - 1$  indices  $X \subset \{1, \dots, m\}$ , for any sequence of possible queries  $Q_X$  and corresponding answers  $A_X$ , and for any  $j \notin X$ , an adversary, given only  $D_j$  and  $(Q_X, A_X)$ , can compute all the secrets. In other words, there does not exist any one-round *strong*  $(k, m)$ -DOT- $\binom{n}{1}$  scheme.

On the other hand, if we consider *implication-free* secrets, i.e., for any  $i, j$ , where  $i \neq j$ , the entropy  $H(\mathbf{W}_i | \mathbf{W}_j) > 0$ , or, in other words, secrets might be related but implications are not present, we can show that in any correct and private one-round  $(k, m)$ -DOT- $\binom{n}{1}$  scheme, it holds that:

$$H(\mathbf{D}_R) \geq (k - 1)H(\mathbf{T}).$$

Two one-round protocols for  $(k, m)$ -DOT- $\binom{2}{1}$  have been proposed<sup>2</sup> in [145]. The first one uses *sparse* bivariate polynomials. The second one uses *fully* bivariate polynomials. Both constructions of  $(k, m)$ -DOT- $\binom{2}{1}$  use, as a building block, a sub-protocol from which a dishonest Receiver can infer at most a linear combination of the secrets held by the Sender. The  $(k, m)$ -DOT- $\binom{2}{1}$  protocol is then obtained by composing in a certain way multiple instances of the sub-protocol.

<sup>2</sup>Notice that, a  $(k, m)$ -DOT- $\binom{2}{1}$  can be used as a black box to set up “more complex” oblivious transfer protocols in the same distributed model (see [49, 84, 92] for unconditionally secure reductions). In this case, any improvement in the design of the available  $(k, m)$ -DOT- $\binom{2}{1}$ , implies directly an improvement of the performance of the more complex protocols.

In [41] we describe one-round  $(k, m)$ -DOT- $\binom{n}{1}$  oblivious transfer protocols, which generalize and strengthen the one-round  $(k, m)$ -DOT- $\binom{2}{1}$  protocols proposed in [145]. We assume that  $1 < k \leq m$ , and implement our protocols over the finite field  $F_p$ , where  $p > \max\{m, n\}$  is prime. We further give some combinatorial constructions for distributed oblivious transfer. Some of these constructions require trivial computations once the scheme has been set up by the Sender. The one-round protocols meet the lower bound on the number of random bits the Receiver must use to set up the queries, described above. However, they are not so efficient in terms of Server memory storage and communication complexity.

It is possible to gain in terms of privacy and efficiency of computations if we allow *one more round* of interaction between the Receiver and the Servers.

## 8 Web-Service Security

Web Services technology [18] is often presented as an efficient solution to the increasing request for flexible connection and dynamic cooperation of distributed systems over the Internet. More in general, such technology offers a new set of standards and techniques which has having a very significant impact on IT organizations and is collecting a large amount of interest concretized in common research projects and money investments. Indeed, interoperability, cross platform communication, and language independence are only a part of the appealing characteristics of Web Services. The standardization and the flexibility introduced by Web services in the development of new applications translate into increased productivity and gained efficiency. Development costs are also reduced since the use of standard interfaces, enabling the integration among applications and the reuse of existing solutions, may shorten the total development cycle.

Actually there are many examples in which Web Service technology has been employed, typically in enterprise application integration, development of B2B integration initiatives and Service Oriented Architecture projects.

Web Services technology provides software developers with a wide range of tools and models to produce innovative distributed applications. Interoperability, cross platform communication, and language independence are only a part of the appealing characteristics of Web Services.

Furthermore, the challenges of fast growing and fast evolving business processes lead to a higher level of interactions and to the need of applications integration across organizational boundaries. Services are no longer designed as isolated processes, but to be invoked by other services and to invoke themselves other services. This paradigm is often referred to as *Service-Oriented Computing* (SOC) [155]. Two different approaches can be considered: *service orchestrations*, i.e., there is one particular service that directs the logical order of all other services, and *service choreographies*, i.e., individual services work together in a loosely coupled network [134].

Interactions should be driven by explicit process models. Therefore, the need of languages to model business process arises in particular for processes implemented by Web Services. Recently, many languages have been proposed, such as BPML [61], XLANG [14], WSCI [13], WS-BPEL [121], WS-CDL [12]. Among these, the Business Process Execution Language for Web Services (WS-BPEL) has emerged as the *de-facto* standard for “enabling users to describe business process activities as Web services and define how they can be connected to accomplish specific tasks” [121]. However, whereas security and access control policies for normal Web Services are well studied, Web Services composition still lacks a standard tool to validate the orchestration of processes, in terms of providing integrity and authenticity of the computation.

## 8.1 Secure metering schemes

As soon as companies provide business services and release new products and tools Web services enabled, a problem which in our opinion will become more and more challenging, is the need of methodologies to accurately measure the number of accesses to a given service. Metering techniques have been developed in the context of measuring accesses to Web pages, where it is important to have a statistics on the exposure of advertising [102, 144]. Recently, several directions for designing efficient and secure metering schemes have been proposed. Many proposals are based on various cryptographic techniques, as secure function evaluations, threshold cryptography, and secret sharing. For instance, see [27, 128, 150].

In the context of Web services usage, such metering techniques can be helpful in the development of new business models, where an *audit agency* is in charge of registering *clients* accessing the services, and of paying the *servers* claiming for payment, after that a valid *proof* of the number of serviced requests is presented. In metering schemes, a *proof* is a value that the server can compute only if a fixed number of clients have visited it or a client has visited it a certain number of times. Such a value is sent to the audit agency at fixed time intervals.

The widespread use of Web Service for providing business level solutions put a great emphasis on security issues of the developed applications. The goal is to provide software developers with standard security techniques, enabling the creation of business models in which service providers get paid for the offered resources. In this sense, it is important to combine authentication techniques with mechanisms which allow to count the number of accesses for a given service in a secure way. Up to a few years ago, the most diffused approach for handling authentication credentials was to use custom SOAP headers to transmit user credentials. In last years, many supplemental standards for guaranteeing security features in Web Services have been proposed and collected under the WS-Security specification [142]. This specification (whose last version has been released at the beginning of the 2006) is the result of the work by the OASIS Technical Committee [148] and proposes a standard set of SOAP extensions that can be used when building secure Web services to provide *confidentiality* (messages should be read only by sender and receiver), *integrity and authentication* (the receiver should be guaranteed that the message is the one sent by the sender and has not been altered), *non repudiation* (the sender should not be able to deny the sending of the message, and *compatibility* (messages should be processed according to the same roles by any node on the path message). Such mechanisms provide a building block that can be used in conjunction with other Web service extensions and higher-level applications to accommodate a wide variety of security models and security technologies. Some other related standard proposals are also available, such as SAML [1], XKMS [2], and so on. Each proposal refers to a particular aspect of securing the information flow and protecting the data exchange between the client and the service provider. Although WS-Security (as well as the other secure related specification) is relatively new, only recently,

it has been widely adopted and (almost) completely supported by multiple toolkits.

### 8.1.1 Our algorithmic solution

In [22], we consider the problem of implementing a secure, authenticated, and efficient *metering system*. In [38] a Web Services-based framework is described that implements the metering scheme presented in [39] for Authenticated Web Services. This implementation is characterized by static invocations of the Web Services using custom SOAP headers to provide authentication. We describe a novel implementation of the framework given in [38], based on WS-Security. The SOAP header contains the authentication token which can be used when accessing a restricted web service, where the authentication token is embedded in the message using the mechanisms provided by WS-Security. We compare our implementation with the one presented in [38] and analyze its performance. This analysis shows that WS-Security is mature enough to provide a flexible and dynamic layer to underlie complex and interactive applications which require security management, without the need of developing ad-hoc solutions for each provided feature. Moreover, the available implementations of the standard allow to solve standardization and compatibility issues, achieving a completely standard and dynamic solution. In particular, we show how WS-Security can be used to overcome the addition of customized information inside the SOAP header to provide authentication in metering schemes.

Furthermore, we present a thorough performance analysis that display that the implementation based on WSS performs better than the one presented in [38], despite the latter provides a dedicated solution.

## 8.2 Validating orchestration of web services

A workflow describes the automation of a business process. During the process documents, information, or roles are exchanged among actors in order to complete a task as specified by a well-defined set of rules. In other words, a workflow is composed by a set of tasks, related to each other through different types of relationships [106].

A workflow management system allows to define, create, and manage the execution of a workflow through a software executing on one or more workflow engines. A workflow manager interpreters the formal definition of a process, in order to interact with several actors by managing states and tasks coordination.

Tasks, actors, and processes within the workflow can be designed as desired. We focus on the Web Service technology and on WS-BPEL, the Business Process Execution Language for Web Services. For a thorough overview of Web Services, we refer to [113]. WS-BPEL is an XML-based language for describing the behavior of business process based on Web Services [121]. It provides *activities*, which can be either basic or structured. A basic activity can communicate with the partners by messages (**invoke**, **receive**, **reply**), manipulate data (**assign**), wait for some time (**wait**), do nothing (**empty**), signal faults (**throw**), or end the entire process (**terminate**). A structured activity defines a causal

order on the basic activities and can be nested in another structured activity itself. The structured activities include sequential execution (**sequence**), parallel execution (**flow**), data-dependent branching (**switch**), timeout-or message-dependent branching (**pick**), and repeated execution (**while**). The most important structured activity is a **scope**. It links an activity to a transaction management and provides fault, compensation, and event handling. A **process** is the outmost scope of the described business process. For more details about BPEL, we refer to [181], [103], and [126].

Nowadays, many implementations are available, such as Bexee [6], Oracle BPEL Process Manager [9], BPEL Maestro [10], or iGrafx [8]. Among these, the ActiveBPEL implementation [5] appears to be one of the most interesting solutions, as being open source and freely available. ActiveBPEL is deployed as a servlet into Apache's Jakarta Tomcat container. It has extensive documentation and has been released by Active EndPoints into the public domain under a Lesser Gnu Public License. It provides a full implementation of the BPEL 1.1 specification. Moreover, Active EndPoints released two useful and powerful tools: ActiveBPEL Designer and ActiveBPEL Enterprise. The former is a comprehensive visual tool for creating, testing, and deploying composite applications based on BPEL standard, and it is a plug-in to the Eclipse development environment [7]. The latter is a complete BPEL engine, which provides many features to administrate BPEL processes.

### 8.2.1 Our algorithmic solution

In [37], we refer to the standard tool for providing integrity and authentication in network interactions, i.e., cryptographic signatures. However, two different issues should be considered when addressing this tool to provide security within Web Services interactions. First, the number of signatures would grow linearly to the number of users and processes, arising issues related to the size of certificate chains and bandwidth overhead. Second, there could be applications requiring a specific order to perform processes.

To this aim, we address a recent cryptographic tool, *aggregate signatures*, first presented by Boneh *et al.* [44]. Aggregate signatures allow to aggregate multiple signatures on distinct messages into a single short signature. Furthermore, we consider a variant of this primitive, in which the set of signers is ordered. This scheme is referred to as *sequential aggregate signatures* and it has been presented by Lysyankaya *et al.*[136].

In [37], we present a framework which uses these two schemes to validate Web Services orchestration, by requiring each partner to sign the result of its computation. In particular, since Web Services interactions can either be performed in parallel or sequentially, we need both schemes: when the computation is carried out in parallel, we use the "standard" aggregate signature scheme presented in [44] and its sequential variant presented in [136] to validate sequential workflow computation.

In our scenario, the orchestrator defines the workflow and describes Web Services composition through BPEL. We want to ensure that:

1. The workflow has been correctly followed by the defined partners.

2. The workflow has been correctly followed in the defined order.
3. The partners cannot repudiate their computation.
4. The partners can verify that their computation has been correctly inserted into the workflow.

To this aim, we refer to the Aggregate Signature schemes as AS and to Sequential Aggregate Signatures as SAS. We hereby propose a novel aggregate signature scheme as the combination of the AS scheme in [44] and the SAS scheme in [136]. As stated above, we need both schemes since composite Web Services execution can be performed either in a parallel or in a sequential way. Therefore, we use SAS to sign messages sequentially produced by the partners. Instead, whenever a parallel execution is performed, we use AS to sign and combine produced messages.

When an orchestration includes both parallel and sequential executions, we need to “map” SAS to AS, since the two signature schemes work on different fields.

At the end of the computation, we expect the Orchestrator to perform signature verification to check its correctness. However, since AS and SAS allow every user to perform aggregate signature verification, every entity involved in the composition may verify that its computation has been correctly inserted in the workflow.

To implement our scheme, it is necessary to modify the ActiveBPEL engine to support signature operations. We address the WS-Security standard to represent information in a standard way. In particular, we refer to the possibility of exchanging security data within the `BinarySecurityToken` field in the message header. In fact, WS-Security provides to developers the chance of defining new security tokens and store them as binary data into the `BinarySecurityToken` field. In particular, it is possible to force the value type of the token to respect an XML schema [11].

In order to add this security layer, we need to modify ActiveBPEL to allow intermediate operations to be performed between services’ invocations. In fact, to the best of our knowledge there is no BPEL engine supporting security features in the service composition. Therefore, we need to modify the process deployment operation by introducing a `sign` service to be invoked.

## References

- [1] SAML. <http://www.oasis-open.org/committees/security/>.
- [2] XML Key Management Specification (XKMS). <http://www.w3.org/TR/xkms/>.
- [3] Simulating TinyOs Networks. <http://www.cs.berkeley.edu/~pal/research/tossim.html>, 2003.
- [4] The Tinyos Community. <http://www.tinyos.net/>, 2007.
- [5] ActiveEndpoints ActiveBPEL Open Source Engine, BPEL Standard. <http://www.activebpel.org/>, Last visit, April 2007.
- [6] Bexee: BPEL EXecution Engine. <http://bexee.sourceforge.net/>, Last visit, April 2007.
- [7] Eclipse SDK. <http://www.eclipse.org>, Last visit, April 2007.
- [8] iGrafx BPEL. <http://www.igrafx.com/products/bpel/>, Last visit, April 2007.
- [9] Oracle BPEL Process Manager. <http://www.oracle.com/technology/bpel/index.html>, Last visit, April 2007.
- [10] Parasoft BPEL Maestro. <http://www.parasoft.com/BPELMaestro>, Last visit, April 2007.
- [11] W3C XML Schema. <http://www.w3.org/XML/Schema>, Last visit, April 2007.
- [12] Web Service Choreography Description Language (WSCDL) 1.0. <http://www.w3.org/TR/ws-cdl-10/>, Last visit, April 2007.
- [13] Web Service Choreography Interface (WSCI) 1.0. <http://www.w3.org/TR/wsci/>, Last visit, April 2007.
- [14] XLANG: XML-based extension of WSDL. <http://www.ebxml.org/xlang.htm>, Last visit, April 2007.
- [15] Great Internet Mersenne Prime Search (GIMPS). <http://www.mersenne.org>, Last Visit July 2007.
- [16] Search for extraterrestrial intelligence (SETI@home). <http://setiathome.berkeley.edu>, Last Visit July 2007.
- [17] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Journal of Computer Networks*, 38:393–422, 2002.
- [18] Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijay Machiraju. *Web Services: Concepts, Architecture and Applications*. Springer Verlag, 2004.

- [19] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik. On the performance of group key agreement protocols. *ACM Transactions on Information and Systems Security*, 7(3):457–488, 2004.
- [20] E. Anton and O. Duarte. Performance analysis of group key establishment protocols in ad hoc networks. Technical report, Universidade Federal do Rio de Janeiro, Brazil, 2006. Technical Report GTA-03-06.
- [21] N. Asokan and P. Ginzboorg. Key agreement in ad hoc networks. *Computer Communications*, 23:1627–1637, 2000.
- [22] V. Auletta, C. Blundo, S. Cimato, E. De Cristofaro, and G. Raimato. Authenticated web services: A ws-security based implementation. In *New Technologies, Mobility and Security: Proceedings of NTMS'2007 Conference - Lecture Notes in Computer Science*, 2007.
- [23] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *43rd Annual Symposium on Foundations of Computer Science*, pages 345–355. IEEE Computer Society Press, November 2002.
- [24] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 299–315. Springer-Verlag, August 2003.
- [25] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *47th Annual Symposium on Foundations of Computer Science*, Berkeley, California, USA, October 22–24, 2006. IEEE Computer Society Press.
- [26] Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT*, pages 480–494, 1997.
- [27] S. G. Barwick, W. Jackson, and K. Martin. A general approach to robust web metering. *Designs, Codes, and Cryptography*, 36(1):5–27, 2005.
- [28] D. Bayer, S. Haber, and W.S. Stornetta. Improving the efficiency and reliability of digital time-stamping. In *Sequences II: Methods in Communication, Security, and Computer Science*, pages 329–334, 1993.
- [29] Donald Beaver, Joan Feigenbaum, Joe Kilian, and Phillip Rogaway. Locally random reductions: Improvements and applications. *J. Cryptology*, 10(1):17–36, 1997.
- [30] K. Becker and U. Wille. Communication complexity of group key distribution. In *5th ACM Conference on Computer and Communications Security (CCS 1998)*, pages 1–6. ACM Press, 1998.

- [31] Amos Beimel, Yuval Ishai, and Tal Malkin. Reducing the servers' computation in private information retrieval: Pir with preprocessing. *J. Cryptology*, 17(2):125–151, 2004.
- [32] Mihir Bellare and Silvio Micali. Non-interactive oblivious transfer and applications. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 547–557. Springer, 1989.
- [33] Josh Benaloh and Michael de Mare. Efficient broadcast time-stamping. Technical Report 1, Clarkson University, Department of Mathematics and Computer Science, August 1991.
- [34] Josh Cohen Benaloh and Michael de Mare. One-way accumulators: A decentralized alternative to digital signatures (extended abstract). In *EUROCRYPT*, pages 274–285, 1993.
- [35] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, 48:313–317, 1979.
- [36] Manuel Blum. Coin flipping by telephone. In *Proc. IEEE Spring COMPCOM*, pages 133–137, 1982.
- [37] C. Blundo, Emiliano De Cristofaro, C. Galdi, and G. Persiano. Validating orchestration of web services with bpm and aggregate signatures. In *Second Italian Workshop on Privacy and Security (PRISE 2007)*, 2007.
- [38] Carlo Blundo and Stelvio Cimato. A framework for authenticated web services. In *Proceedings of European Conference on Web Services (ECOWS 04)*, *Lecture Notes in Computer Science*, volume 3250, pages 61–71. Springer-Verlag, Berlin, 2004.
- [39] Carlo Blundo and Stelvio Cimato. A software infrastructure for authenticated web metering. *IEEE Computer*, 37(4):28–33, 2004.
- [40] Carlo Blundo, Emiliano De Cristofaro, Aniello Del Sorbo, Clemente Galdi, and Giuseppe Persiano. Distributed outsourcing of computation on private data. Technical report, Università di Salerno, 2007.
- [41] Carlo Blundo, Paolo D'Arco, Alfredo De Santis, and Douglas R. Stinson. On unconditionally secure distributed oblivious transfer. *J. Cryptology*, 20(3):323–373, 2007.
- [42] Carlo Blundo, Barbara Masucci, Douglas R. Stinson, and Ruizhong Wei. Constructions and bounds for unconditionally secure non-interactive commitment schemes. *Des. Codes Cryptography*, 26(1-3):97–110, 2002.

- [43] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2004)*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.
- [44] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Proceedings of Advances in Cryptology - Eurocrypt 2003, Lecture Notes in Computer Science*, volume 2656, pages 416–432. Springer-Verlag, Berlin, 2003.
- [45] Olaf Bonorden, Joachim Gehweiler, and Friedhelm Meyer auf der Heide. A Web Computing Environment for Parallel Algorithms in Java. In *Proceedings of International Conference on Parallel Processing and Applied Mathematics (PPAM)*, pages 801–808, 2005.
- [46] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.
- [47] Gilles Brassard, Claude Crépeau, and Jean-Marc Robert. All-or-nothing disclosure of secrets. In Odlyzko [149], pages 234–238.
- [48] Gilles Brassard, Claude Crépeau, and Jean-Marc Robert. Information theoretic reductions among disclosure problems. In *FOCS*, pages 168–173. IEEE, 1986.
- [49] Gilles Brassard, Claude Crépeau, and Miklos Santha. Oblivious transfers and intersecting codes. *IEEE Transactions on Information Theory*, 42(6):1769–1780, 1996.
- [50] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Provably authenticated group Diffie-Hellman key exchange – the dynamic case. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 290–309. Springer-Verlag, December 2001.
- [51] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Group Diffie-Hellman key exchange secure against dictionary attacks. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 497–514. Springer-Verlag, December 2002.
- [52] Emmanuel Bresson, Olivier Chevassut, David Pointcheval, and Jean-Jacques Quisquater. Provably authenticated group Diffie-Hellman key exchange. In *ACM CCS 01: 8th Conference on Computer and Communications Security*, pages 255–264. ACM Press, November 2001.
- [53] Ahto Buldas and Aivo Jürgenson. Does secure time-stamping imply collision-free hash functions? In *1st International Conference on Provable Security (ProvSec) 2007*, pages 138–150, 2007. LNCS 4784.

- [54] Ahto Buldas, Peeter Laud, Helger Lipmaa, and Jan Willemson. Time-stamping with binary linking schemes. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO’98*, volume 1462 of *Lecture Notes in Computer Science*, pages 486–501. Springer-Verlag, August 1998.
- [55] Ahto Buldas, Peeter Laud, Märt Saarepera, and Jan Willemson. Universally composable time-stamping schemes with audit. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC 05: 8th Information Security*, volume 3650 of *Lecture Notes in Computer Science*, pages 359–373. Springer-Verlag, Berlin, Germany, September 20–23, 2005.
- [56] Ahto Buldas, Peeter Laud, Märt Saarepera, and Jan Willemson. Universally composable time-stamping schemes with audit. Cryptology ePrint Archive, Report 2005/198, 2005. <http://eprint.iacr.org/>.
- [57] Ahto Buldas and Sven Laur. Do broken hash functions affect the security of time-stamping schemes? In Jianying Zhou, Moti Yung, and Feng Bao, editors, *ACNS 05: 3rd International Conference on Applied Cryptography and Network Security*, *Lecture Notes in Computer Science*, pages 50–65, Singapore, June 6–9, 2006. Springer-Verlag.
- [58] Ahto Buldas and Sven Laur. Knowledge-binding commitments with applications in time-stamping. In *International Conference on Theory and Practice of Public-Key Cryptography (PKC 2007)*, pages 150–165, 2007. LNCS 4450.
- [59] Ahto Buldas and Märt Saarepera. On provably secure time-stamping schemes. In Pil Joong Lee, editor, *Advances in Cryptology – ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 500–514. Springer-Verlag, December 2004.
- [60] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptology (EUROCRYPT 1994)*, pages 275–286. Springer-Verlag, 1994. *Lecture Notes in Computer Science*, LNCS 950.
- [61] Business Process Modeling Language. <http://www.bpmi.org>. Last visit, April 2007.
- [62] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer-Verlag, August 2001.
- [63] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th Annual ACM Symposium on Theory of Computing*, pages 494–503. ACM Press, May 2002.

- [64] Claude Castelluccia, Einar Mykletun, and Gene Tsudik. Efficient Aggregation of encrypted data in Wireless Sensor Networks. In *MOBIQUITOUS '05: Proceedings of the The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, pages 109–117, Washington, DC, USA, 2005. IEEE Computer Society.
- [65] Dario Catalano, Yevgeniy Dodis, and Ivan Visconti. Mercurial commitments: Minimal assumptions and efficient constructions. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, Lecture Notes in Computer Science, pages 120–144. Springer-Verlag, 2006.
- [66] Dario Catalano and Ivan Visconti. Hybrid commitments and their applications to zero-knowledge proof systems. *Theor. Comput. Sci.*, 374(1-3):229–260, 2007.
- [67] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy (2003)*, page 197, 2003.
- [68] Haowen Chan, Adrian Perrig, and Dawn Song. Secure Hierarchical in-network Aggregation in Sensor Networks. In *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS 2006)*, pages 278–287, 2006.
- [69] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In Cramer [77], pages 422–439.
- [70] I. Chatzigiannakis, A. Kinalis, and S. Nikolettseas. Wireless sensor networks protocols for efficient collision avoidance in multi-path data propagation. In *ACM Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN 2004)*, pages 8–16, 2004.
- [71] I. Chatzigiannakis, E. Konstantinou, V. Liagkou, and P. Spirakis. Design, analysis and performance evaluation of group key establishment in wireless sensor networks. In Springer-Verlag, editor, *ICALP 2006 Workshop, Electronic Notes in Theoretical Computer Science*, volume 2nd Workshop on Cryptography for Ad hoc Networks, 2006.
- [72] I. Chatzigiannakis, E. Konstantinou, V. Liagkou, and P. Spirakis. Agent-based distributed group key establishment in wireless sensor networks. In *TSPUC 2007 Workshop, IEEE Proceedings*, volume 3rd IEEE International Workshop on Trust, Security and Privacy for Ubiquitous Computing, 2007.
- [73] Arijit Chaudhuri and Rahul Mukerjee. *Randomized Response: Theory and Techniques*, volume 95 of *Statistics: Textbooks and Monographs*. Marcel Dekker, Inc., 1988. ISBN: 0824777859.

- [74] B. Chor, N. Gilboa, and M. Naor. Private information retrieval by keywords. In *TR-CS0917, Department of Computer Science, Technion*, 1997.
- [75] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.
- [76] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.
- [77] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.
- [78] Claude Crépeau. A zero-knowledge poker protocol that achieves confidentiality of the players’ strategy or how to achieve an electronic poker face. In Odlyzko [149], pages 239–247.
- [79] Claude Crépeau. Equivalence between two flavours of oblivious transfers. In Carl Pomerance, editor, *CRYPTO*, volume 293 of *Lecture Notes in Computer Science*, pages 350–354. Springer, 1987.
- [80] Giovanni Di Crescenzo, Yuval Ishai, and Rafi Ostrovsky. Non-Interactive and Non-Malleable Commitment. In *30th Annual ACM Symposium on Theory of Computing*, pages 141–150. ACM Press, May 1998.
- [81] Ivan Damgård. Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 418–430. Springer-Verlag, May 2000.
- [82] Ivan Damgård, editor. *Commitment schemes and zero knowledge protocols*, volume 1561 of *Lecture Notes in Computer Science*. Springer, 1999.
- [83] Ivan Damgård and Jens Groth. Non interactive and reusable non-malleable commitments. In *35th Annual ACM Symposium on Theory of Computing*, pages 426–437. ACM Press, June 2003.
- [84] Paolo D’Arco and Douglas R. Stinson. Generalized zig-zag functions and oblivious transfer reductions. In Serge Vaudenay and Amr M. Youssef, editors, *Selected Areas in Cryptography*, volume 2259 of *Lecture Notes in Computer Science*, pages 87–102. Springer, 2001.
- [85] E. De Cristofaro. A secure and privacy-protecting aggregation scheme for sensor networks. In *Proceedings of the 2007 IEEE International Workshop "From Theory to Practice in Wireless Sensor Networks"*. IEEE Press, 2007.

- [86] Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. Zero-knowledge arguments and public-key cryptography. *Inf. Comput.*, 121(1):23–40, 1995.
- [87] Alfredo De Santis and Giuseppe Persiano. Public-randomness in public key cryptography. In *EUROCRYPT*, pages 46–62, 1990.
- [88] Aniello Del Sorbo, Clemente Galdi, and Giuseppe Persiano. Distributed certified information access for mobile devices. In Damien Sauveron, Constantinos Markantonakis, Angelos Bilas, and Jean-Jacques Quisquater, editors, *WISTP*, volume 4462 of *Lecture Notes in Computer Science*, pages 67–79. Springer, 2007.
- [89] Giovanni Di Crescenzo, Yuval Ishai, and Rafail Ostrovsky. Universal service-providers for database private information retrieval (extended abstract). In *PODC*, pages 91–100, 1998.
- [90] Giovanni Di Crescenzo, Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Efficient and non-interactive non-malleable commitment. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 40–59. Springer-Verlag, May 2001.
- [91] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.
- [92] Yevgeniy Dodis and Silvio Micali. Lower bounds for oblivious transfer reductions. In *EUROCRYPT*, pages 42–55, 1999.
- [93] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *23rd Annual ACM Symposium on Theory of Computing*, pages 542–552. ACM Press, May 1991.
- [94] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *10th ACM conference on Computer and communications security (CCS 03)*, pages 42–51, 2003.
- [95] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. In *30th Annual ACM Symposium on Theory of Computing*, pages 409–418. ACM Press, May 1998.
- [96] T. ElGamal. A conference key distribution system. *IEEE Transactions on Information Theory*, 28:714–720, 1982.
- [97] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.
- [98] Uri Feige and Adi Shamir. Zero-Knowledge Proofs of Knowledge in Two Rounds. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO’89*, volume 435 of *Lecture Notes in Computer Science*, pages 526–544. Springer-Verlag, August 1990.

- [99] Uriel Feige. *Alternative Models for Zero Knowledge Interactive Proofs*. Weizmann Institute of Science, 1990.
- [100] Michael J. Fischer, Silvio Micali, and Charles Rackoff. A secure protocol for the oblivious transfer (extended abstract). *J. Cryptology*, 9(3):191–195, 1996.
- [101] Marc Fischlin and Roger Fischlin. Efficient non-malleable commitment schemes. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 413–431. Springer-Verlag, August 2000.
- [102] Matthew K. Franklin and Dahlia Malkhi. Auditable metering with lightweight security. *Journal of Computer Security*, 6(4):237–256, 1998.
- [103] Xiang Fu, Tevfik Bultan, and Jianwen Su. Analysis of interacting bpm web services. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 621–630, New York, NY, USA, 2004. ACM Press.
- [104] G. Gaubatz, J. Kaps, and B. Sunar. Public key cryptography in sensor networks – revisited. In *1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004)*, pages 2–18. Springer-Verlag, Lecture Notes in Computer Science, 2004. LNCS 3313.
- [105] Rosario Gennaro. Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 220–236. Springer-Verlag, August 2004.
- [106] Dimitrios Georgakopoulos, Mark Hornick, and Amit Sheth. An overview of workflow management: from process modeling to workflow automation infrastructure. *Distrib. Parallel Databases*, 3(2):119–153, 1995.
- [107] Yael Gertner, Shafi Goldwasser, and Tal Malkin. A random server model for private information retrieval or how to achieve information theoretic pir avoiding database replication. In Michael Luby, José D. P. Rolim, and Maria J. Serna, editors, *RANDOM*, volume 1518 of *Lecture Notes in Computer Science*, pages 200–217. Springer, 1998.
- [108] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. *J. Comput. Syst. Sci.*, 60(3):592–629, 2000.
- [109] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *FOCS*, pages 325–335, 2000.
- [110] O. Goldreich. *Foundations of Cryptography II: Basic Applications*. Cambridge University Press, 2004.

- [111] Oded Goldreich and Ariel Kahan. How to Construct Constant-Round Zero-Knowledge Proof. *Journal of Cryptology*, 9(3):167–190, 1996.
- [112] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229. ACM, 1987.
- [113] Steve Graham, Simeon Simeonov, Toufic Boubez, Glen Daniels, Doug Davis, Yuichi Nakamura, and Ryo Neyama. *Building Web Services with Java: Making sense of XML, SOAP, WSDL, and UDDI*. 2001.
- [114] N. Gura, A. Pate, A. Wander, H. Eberle, and S. Shantz. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In *Cryptographic Hardware and Embedded Systems (CHES 2004)*, pages 119–132. Springer-Verlag, Lecture Notes in Computer Science, 2004. LNCS 3156.
- [115] Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. *Journal of Cryptology*, 3(2):99–111, 1991.
- [116] Stuart Haber and W. Scott Stornetta. Secure names for bit-strings. In *ACM CCS 97: 4th Conference on Computer and Communications Security*, pages 28–35. ACM Press, April 1997.
- [117] Iftach Haitner, Omer Horvitz, Jonathan Katz, Chiu-Yuen Koo, Ruggero Morselli, and Ronen Shaltiel. Reducing complexity assumptions for statistically-hiding commitment. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, Lecture Notes in Computer Science, pages 58–77. Springer-Verlag, May 2005.
- [118] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO’96*, volume 1109 of *Lecture Notes in Computer Science*, pages 201–215. Springer-Verlag, August 1996.
- [119] Lingxuan Hu and David Evans. Secure aggregation for wireless networks. In *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT’03 Workshops)*, page 384, Washington, DC, USA, 2003. IEEE Computer Society.
- [120] J. P. Hubaux, L. Buttyftn, and S. Capkun. The quest for security in mobile ad hoc networks. In *ACM Symposium on Mobile Ad Hoc Networking and Computing*, pages 146 – 155, 2005.
- [121] IBM, Microsoft, and BEA Systems. Business process execution language for web services. August 2002. <http://www.ibm.com/developerworks/library/ws-bpel>.
- [122] Chris Karlof, Naveen Sastry, and David Wagner. Tinysec: a link layer security architecture for wireless sensor networks. In *SenSys ’04: Proceedings of the 2nd*

- international conference on Embedded networked sensor systems*, pages 162–175, New York, NY, USA, 2004. ACM Press.
- [123] Emilia Käsper, Ventzislav Nikov, and Svetla Nikova. Strongly multiplicative hierarchical threshold secret sharing. In *International Conference on Information Theoretic Security*, Lecture Notes in Computer Science. Springer, May 25-28 2007.
- [124] Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125. Springer, 2003.
- [125] John Kelsey and Tadayoshi Konho. Herding hash functions and the nostradamus attack. Technical Report 281, IACR e-print archive, 2006.
- [126] R. Khalaf, A. Keller, and F. Leymann. Business processes for Web Services: principles and applications. *IBM Syst. J.*, 45(2):425–446, 2006.
- [127] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC*, pages 20–31. ACM, 1988.
- [128] Soon Seok Kim, Sung Kwon Kim, and Hong-Jin Park. New approach for secure and efficient metering in the web advertising. In *Proceedings of International Conference on Computational Science and Its Applications (ICCSA 2004)*, *Lecture Notes of Computer Science*, volume 3043, pages 215–221. Springer-Verlag, Berlin, 2004.
- [129] Y. Kim, A. Perrig, and Gene Tsudik. Tree-based group key agreement. *ACM Transactions on Information and Systems Security*, 7(1):60–96, 2004.
- [130] Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *38th Symposium on Foundations of Computer Science (FOCS 1997)*, pages 364–373. IEEE Computer Society, 1997.
- [131] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. Tossim: Accurate and Scalable Simulation of entire Tinyos Applications. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 126–137. ACM Press, 2003.
- [132] L. Liao. Group key agreement for ad hoc networks. Master’s thesis, Ruhr-University Bochum, Germany, 2005.
- [133] Moses Liskov. Updatable zero-knowledge databases. In Bimal K. Roy, editor, *11th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2005)*, volume 3788 of *Lecture Notes in Computer Science*, pages 174–198. Springer, 2005.

- [134] Niels Lohmann, Peter Massuthe, Christian Stahl, and Daniela Weinberg. Analyzing Interacting BPEL Processes. In *Business Process Management, 4th International Conference, BPM 2006, Vienna, Austria, September 5-7, 2006, Proceedings*, volume 4102 of *Lecture Notes in Computer Science*, pages 17–32. Springer-Verlag, September 2006.
- [135] H. Luo and S. Lu. Ubiquitous and robust authentication services for ad hoc wireless networks. In *7th IEEE Symposium on Computer and Communications (ISCC'02)*, 2002.
- [136] Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential aggregate signatures from trapdoor permutations. In *Proceedings of Advances in Cryptology - Eurocrypt 2004, Lecture Notes in Computer Science*, volume 3027, pages 74–90. Springer-Verlag, Berlin, 2004.
- [137] Philip D. MacKenzie and Ke Yang. On simulation-sound trapdoor commitments. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EURO-CRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 382–400. Springer-Verlag, May 2004.
- [138] D. J. Malan, M. Welsh, and M. D. Smith. A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *2nd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, pages 71–80, 2004.
- [139] Alfred Menezes and Palash Sarkar, editors. *Progress in Cryptology - INDOCRYPT 2002, Third International Conference on Cryptology in India, Hyderabad, India, December 16-18, 2002*, volume 2551 of *Lecture Notes in Computer Science*. Springer, 2002.
- [140] Silvio Micali, Michael O. Rabin, and Joe Kilian. Zero-knowledge sets. In *44th Symposium on Foundations of Computer Science (FOCS 2003)*, pages 80–91. IEEE Computer Society, 2003.
- [141] Sanjeev Kumar Mishra and Palash Sarkar. Symmetrically private information retrieval. In Bimal K. Roy and Eiji Okamoto, editors, *First International Conference in Cryptology in India (Indocrypt 2000)*, volume 1977 of *Lecture Notes in Computer Science*, pages 225–236. Springer, 2000.
- [142] Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, and Ronald Monzillo. Web Services Security: SOAP Message Security 1.1. *OASIS* . <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>, 2006.

- [143] Moni Naor. Bit Commitment Using Pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.
- [144] Moni Naor and Benny Pinkas. Secure and efficient metering. In *Proceedings of Advances in Cryptology - Eurocrypt '98, Lecture Notes in Computer Science*, volume 1403, pages 576–590, 1998.
- [145] Moni Naor and Benny Pinkas. Distributed oblivious transfer. In Tatsuaki Okamoto, editor, *ASIACRYPT*, volume 1976 of *Lecture Notes in Computer Science*, pages 205–219. Springer, 2000.
- [146] Moni Naor, Benny Pinkas, and Reuban Sumner. Privacy preserving auctions and mechanism design. In *ACM Conference on Electronic Commerce*, pages 129–139, 1999.
- [147] Kaisa Nyberg. Fast accumulated hashing. In Dieter Gollmann, editor, *Fast Software Encryption*, volume 1039 of *Lecture Notes in Computer Science*, pages 83–87. Springer, 1996.
- [148] The Oasis standard body. <http://www.oasis-open.org>.
- [149] Andrew M. Odlyzko, editor. *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*. Springer, 1987.
- [150] Wakaha Ogata and Kaoru Kurosawa. Provably secure metering scheme. In *Proceedings of ASIACRYPT 00, Lecture Notes in Computer Science*, volume 1976, pages 388–398. Springer-Verlag, Berlin, 2000.
- [151] Wakaha Ogata and Kaoru Kurosawa. Oblivious keyword search. *Journal of Complexity*, 20(2-3):356–371, 2004.
- [152] Rafail Ostrovsky, Giuseppe Persiano, and Ivan Visconti. Constant-round concurrent non-malleable commitments and decommitments. Technical report, Università di Salerno, Italy, 2007.
- [153] Rafi Ostrovsky, Giuseppe Persiano, and Ivan Visconti. Concurrent Non-Malleable Witness Indistinguishability and its Applications. Cryptology ePrint Archive, Report 2006/256, 2006.
- [154] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, May 1999.

- [155] Mike P. Papazoglou. Agent-oriented technology in support of e-business. *Communications of ACM*, 44(4):71–77, 2001.
- [156] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *46th Annual Symposium on Foundations of Computer Science*, pages 563–572. IEEE Computer Society Press, 2005.
- [157] Rafael Pass and Alon Rosen. New and improved constructions of non-malleable cryptographic protocols. In *37th Annual ACM Symposium on Theory of Computing*, pages 533–542. ACM Press, 2005.
- [158] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments (full version). <http://www.eecs.harvard.edu/~alon/PAPERS/conc-nmc/conc-nmc.ps>, 2006.
- [159] Adrian Perrig, John Stankovic, and David Wagner. Security in Wireless Sensor Networks. *Commun. ACM*, 47(6):53–57, 2004.
- [160] Bartosz Przydatek, Dawn Song, and Adrian Perrig. SIA: Secure Information Aggregation in Sensor Networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 255–265, New York, NY, USA, 2003. ACM Press.
- [161] M. Rabin. How to exchange secrets by oblivious transfer. Technical Report Technical Memo TR-81, 1981.
- [162] Cauligi S. Raghavendra, Krishna M. Sivalingam, and Taieb F. Znati. *Wireless Sensor Networks*. Springer, 2003.
- [163] Vincent Rijmen and Elisabeth Oswald. Update on sha-1. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 58–71. Springer, 2005.
- [164] R. Rivest. Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer. manuscript. Available: <http://theory.lcs.mit.edu/~rivest/publications.html>.
- [165] L. Schenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *ACM Conference on Computer and Communications Security*, pages 41–47, 2002.
- [166] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.
- [167] Victor Shoup, editor. *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*. Springer, 2005.

- [168] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *EUROCRYPT*, pages 334–345, 1998.
- [169] Dawn Xiaodong Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *IEEE Symposium on Security and Privacy*, pages 44–55, 2000.
- [170] M. Steiner, G. Tsudik, and M. Waidner. Diffie-hellman key distribution extended to group communication. In *3rd ACM Conference on Computer and Communications Security (CCS 1996)*, pages 31–37. ACM Press, 1996.
- [171] D. R. Stinson and R. Wei. Bibliography on secret sharing schemes. <http://www.cacr.math.uwaterloo.ca/~dstinson/ssbib.html>.
- [172] Douglas R. Stinson. An explication of secret sharing schemes. *Des. Codes Cryptography*, 2(4):357–390, 1992.
- [173] Wen-Guey Tzeng. Efficient 1-out-n oblivious transfer schemes. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography*, volume 2274 of *Lecture Notes in Computer Science*, pages 159–171. Springer, 2002.
- [174] David Wagner. Resilient Aggregation in Sensor Networks. In *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 78–87, New York, NY, USA, 2004. ACM Press.
- [175] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the hash functions md4 and ripemd. In Cramer [77], pages 1–18.
- [176] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full sha-1. In Shoup [167], pages 17–36.
- [177] Xiaoyun Wang and Hongbo Yu. How to break md5 and other hash functions. In Cramer [77], pages 19–35.
- [178] Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin. Efficient collision search attacks on sha-0. In Shoup [167], pages 1–16.
- [179] Dirk Westhoff, Joao Girão, and Mithun Acharya. Concealed Data Aggregation for Reverse Multicast Traffic in Sensor Networks: Encryption, Key Distribution, and Routing Adaptation. *IEEE Trans. Mob. Comput.*, 5(10):1417–1431, 2006.
- [180] S. Wiesner. Conjugate coding. *SIGACT News*, 15:78–88, 1983.
- [181] Petia Wohed, Wil M. P. van der Aalst, Marlon Dumas, and Arthur H. M. ter Hofstede. Analysis of Web Services Composition Languages: The Case of BPEL4WS. In *Proceedings of the 22nd International Conference on Conceptual Modeling (ER)*, pages 200–215, 2003.

- [182] L. Zhou and Z. J. Haas. Securing ad hoc networks. In *IEEE Network Magazine*, pages 24–30, 1999.