An Anonymous Efficient Private Set Intersection Protocol for Wireless Sensor Networks

George Moldovan, Anda Ignat

Department of Design, Computer Science and Media
RheinMain University of Applied Sciences
Unter den Eichen 5
65195 Wiesbaden
george.moldovan@hs-rm.de
anda.ignat@hs-rm.de

Abstract: We present an efficient protocol which, under certain assumptions, provides a suitable level of security and anonymity in the ideal cipher model when computing the intersection of two private data-sets containing lists of elements from a large domain. The assumptions are that each node is pre-loaded with a set of pseudonyms, signed by the network's trusted authority; that the cardinality of each data-set is globally known. Our protocol first establishes a secure, trusted connection between two partners, then uses lightweight, symmetrical key operations for encoding and privately comparing the elements of two sets. Given a cryptographically secure symmetric encryption scheme, our protocol is safe for both semi-honest and malicious adversaries. The primary target platform for this protocol are Wireless Sensor Networks (WSNs), specifically those used in Ambient Assisted Living (AAL) scenarios, which almost entirely consist of a heterogeneous mix of devices, providers and manufacturers.

1 Introduction

Self-organizing WSNs consist of a multitude of small-scale devices with sensing properties and with wireless communication capabilities. The size and cost of such nodes makes them easy to be manufactured and distributed, but an inherent implication is the fact that they are resource restricted devices: low energy, low CPU processing power, small memory and storage space. WSNs are used to collect, process and distribute data from an environment. This makes them usable in projects related to military or surveillance tasks, vehicular networks communication, body and environment monitoring with AAL applications.

The AAL technologies are meant to help create and maintain a safe environment (both medical and socially) that addresses the needs of elderly or impaired people. The scenario is even more relevant considering that the current trend in Europe sees a rise in the percentage of elderly people [Ste05]. A cost-effective, unintrusive care, that would allow them to retain their autonomy is an appropriate, welcomed solution.

The use of WSNs in the AAL context implies that important security and privacy re-

quirements, that address both legal and personal needs, have to be fulfilled for them to be commercially and legally accepted (i.e. patience confidentiality needs maintaining). The public's acceptance, as they are social projects, is also a key aspect.

The WSN security is a broad research domain with specific challenges: privacy protocols and algorithms have to suit their hardware restrictions, but this has to be done without sacrificing the security level and without hindering other applications from performing their tasks, as the security should be a discrete, intrusive layer. Even more, in case of AAL projects, there are some additional, notable requirements:

- For certain AAL use case scenarios, the trade-off between performance and security needs to not surpass certain metrics, like the response time needed for a system to respond to the input of a user. Such interactions between the users and the system should be felt as instantaneous, i.e. users shouldn't have to wait in the seconds or minutes range if they want to remotely trigger a light-switch.
- Concerning groups of sensors and devices, an AAL scenario is implicitly heterogeneous, as several users and their devices might join or leave the network: visits from friends and technical or medical personnel.
- The anonymity of the user, of his processes and requests is another important aspect - a TV, light switch or energy measurement system should be oblivious of the identity of its users, but at the same time it should retain the ability to validate their authorization of using its services.

Problem Statement and Our Contribution Suppose node A is a data aggregator or end device (i.e. a display) that queries node B. Because of the multi-hop, ad hoc nature of the network, one could consider two relevant scenarios: (i) node A cannot reach B directly, and (ii) node A has to make sure (i.e. by checking a list of permissions) that it only accepts data readings from a node affiliated with the network. In the scenario (i), node B would have to initiate a route through trusted nodes only, while in the latter (ii) A would have to check whether B's permissions match its own access list. In both scenarios, this is ideally done without any of the involved parties revealing any additional information except a list of common group memberships in the data routing scenario, or the list of common permissions in the access control scenario. This is known as the *Private Set Intersection* [FNP04] problem .

The goal of this paper is to offer an efficient private set intersection algorithm, usable on resource restrained sensor nodes. We make the following assumptions: that there exists a trusted authority (TA), that the number of elements in each node's data-set is relatively small, and that there is a global domain from which the elements were extracted and which is large enough. It is beyond the scope of this paper to provide suggestions regarding changes to the sets on the nodes in the WSN after deployment, like the revocation of an element.

Returning to the previous AAL scenario, let node A and B each have k elements from a common domain whose size is n. These elements could stand for groups inside the

network or different access permissions. It is required to securely and privately compute the intersection of the two groups belonging to *A* and *B*. Given $A = \{a_1, ..., a_k\}$ and $B = \{b_1, ..., b_k\}$, compute $A \cap B$ without disclosing any other information about the identity of the other elements.

We propose a resource-efficient solution which preserves the anonymity of the involved nodes, the anonymity of the elements that are not part of the their sets intersection, and which uses lightweight cryptographic operations for its methods. Briefly described, the core of the protocol uses the value of each element in a set, it's signature generated by the overall TA, and a secure, shared session key. Both parties will encrypt each of their set elements by computing, from the element's corresponding signature and the shared session key, a one-time symmetric key.

The role of the signature associated to each element, and it's actual form, is to prevent the creation or unauthorized association of elements by a node. A detailed explanation is not in the scope of this paper and will be explained in a future publication

An important note is that, since there is not real need for decrypting the generated, encrypted values, the use of a message authentication code like HMAC [19802] or AES-CMAC [TIETF06] would yield an even faster protocol, with fewer resource requirements.

Paper Organization The rest of the paper is organized as follows. Section 2 briefly describes the related work and the general functionality of alternative solutions. Section 3 describes the cryptographic blocks used in the current form of the protocols. Section 4 contains the detailed description of our protocol, together with its prerequisites and explicit assumptions. Section 3.3 lists the behaviour of different possible malicious nodes; Section 5 offers an overview of future plans and changes to the protocol, security wise.

2 Related Work

Freedman, Nissim and Pinkas (FNP) proposed in [FNP04] a *Private Disjointness Test*, achieved by using a scheme that preserves the group homomorphism of addition, as well as of multiplication with a constant.

An additive homomorphic, public key cryposystem (Paillier's cryptosystem [Pai99]), allows a party to oblivious compute $E_{pk}(x) \cdot E_{pk}(y) = E_{pk}(x+y)$ or to compute $(E_{pk}(y))^x = E_{pk}(x \cdot y)$ being given only $E_{pk}(x)$ and $E_{pk}(y)$, where *x*, *y* are some plain-text messages and E_{pk} is the additive homomorphic function.

The basic structure of the FNP protocol consists of defining a polynomial *P* whose roots are the elements of private set $X = \{x_1, \ldots, x_n\}$ of size k_c:

$P(y) = (x_1 - y) \cdot (x_2 - y) \cdot \ldots \cdot (x_{k_c} - y) = \sum_{u=0}^{k_c} \alpha_u y^u$.

The α_u coefficients are then encrypted using a Paillier's cryptosystem and sent by the verifier *A* to the prover *B*, whose role is to evaluate the polynomial for each of his private elements. Each result is randomized by multiplying it with a non-zero constant *r*. An

optional final step is to also add the element to the randomized polynomial result. The resulting cipher-text will thus have the form $E_{pk}(r \cdot P(y))$ or $E_{pk}(r \cdot P(y) + y)$ for each element *y* of *B*. Once the verifier *A* receives the encryptions back from the *B*, he only needs to check if any of the decrypted values are zero, since any common element would be a root of *P*. If the latter encryption was used, *A* will need to check if the decryption matches an element from his own set.

Agrawal, Evfimievski and Srikant (AgES) proposed a different approach in [AES03], for information sharing accross private database. For it to work, a commutative encryption scheme is needed [Gam85], [MVO96]. Informally, a commutative encryption schemes uses to a pair of cryptographic function E_1 and E_2 with the property that $E_A(E_B(m) = E_B(E_A(m))$. If just $E_A(E_B(m))$ is known, neither *A* or *B* will be able to retrieve *m*.

AgES works as follows. Both partners *A* and *B* apply a hash on their private sets. Each then generates a secret key and encrypts each hashed element. The resulting sets of encryptions $\{E_A(H(x_1^A)), \ldots, E_A(H(x_n^A))\}$ and $\{E_B(H(x_1^B)), \ldots, E_B(H(x_k^B))\}$ are exchanged, where H is a cryptographic hash function, E is the encryption function using *A*'s and *B*'s secret keys, *n* and *k* are the cardinalities of the two sets. Each party then encrypts all the elements it received from the communication partner. In *A*'s case, this means computing $E_A(E_B(H(x_j^B))$, for all $1 \leq j \leq k$. For *B*, $E_B(E_A(H(x_i^A))$, for all $1 \leq i \leq n$. *A* then proceeds to send pairs of the form $P_j = \langle E_B(H(x_j^B), E_A(E_B(H(x_j^B))\rangle$ back to *B*. By intersecting the commutative encrypted set received from *A* with his own, *B* will get a subset of elements for which $E_A(E_B(H(x_i^B)) = E_B(E_A(H(x_j^A))$. x_j^B is then found by using P_j.

Both the FNP and the AgES protocols and their variants imply the use of public-key operations, whose number is directly proportional with the size of the elements in the groups of the involved parties. This is problematic especially for sensor networks.

3 Preliminaries

3.1 Bilinear Pairings

Let $(\mathbb{G}_1, +)$ and $(\mathbb{G}_2, \cdot)$ denote two cyclic groups of order *q* (some large prime), in which the Discrete Logarithm Problem (DLP) is considered to be hard.

An admissible bilinear pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ has the following properties:

1. *Bilinearity*: $\forall g_1, g_2, g_3 \in \mathbb{G}_1,\ \hat{e}(g_1 + g_2, g_3) = \hat{e}(g_1, g_3) \cdot \hat{e}(g_2, g_3)$ and $\hat{e}(g_1, g_2 + g_3) = \hat{e}(g_1, g_2) \cdot \hat{e}(g_1, g_3)$.
2. *Non-degeneracy*: $\exists g_1, g_2 \in \mathbb{G}_1 :\ \hat{e}(g_1, g_2) \neq 1$.
3. *Computability*: $\forall g_1, g_2 \in \mathbb{G}_1,\ \hat{e}(g_1, g_2)$ is efficiently computable.

In practice, the known implementations of these pairings - the Weil ([Jou00]) and the Tate pairings - prove that such constructions exist and involve fairly complex mathematics.

Typically, $\mathbb{G}_1$ is an elliptic-curve group and $\mathbb{G}_2$ is a finite field.

3.2 Identity-based Encryption

Following the definition given by Boneh and Franklin [BF03], an identity-based encryption scheme (IBE) is specified by the following four algorithms:

Setup Given a security parameter $k \in \mathbb{Z}^+$ as input, the algorithm returns the system parameters $params$ and the $master - key$.

Step 1: A randomized algorithm (known as a Bilinear Diffie-Hellman parameter generator), running on input k in polynomial time, generates: a prime q, the description of two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q, and the description of an admissible bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. A random generator $P \in \mathbb{G}_1$ is also chosen.

Step 2: A random $s \in \mathbb{Z}_q^*$ is picked and $P_{pub} = sP$ is set.

Step 3: The cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$ and $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^n$ for some n are chosen.

Then, the message space is defined as $\mathcal{M} = \{0,1\}^n$ and the cipher-text space as $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$. The system parameters $params = \{q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2\}$ are published, and the $master - key$ $s \in \mathbb{Z}_q^*$ is kept secret.

Extract Using $params,\ master - key$ and an arbitrary $ID \in \{0,1\}^*$ (the public key) as input, a private key d_{ID} is returned by:

1. computing $Q_{ID} = H_1(ID) \in \mathbb{G}_1^*$, and
2. setting $d_{ID} = sQ_{ID},\ s$ being the $master - key$.

Encrypt With $params,\ ID$ and $M \in \mathcal{M}$ as input, the cipher-text $C \in \mathcal{C}$ is returned by doing the following:

1. compute $Q_{ID} = H_1(ID) \in \mathbb{G}_1^*$,
2. pick a random $r \in \mathbb{Z}_q^*$, and
3. set $C = \langle rP, M \oplus H_2(g_{ID}^r) \rangle$ where $g_{ID} = \hat{e}(Q_{ID}, P_{pub}) \in \mathbb{G}_2^*$.

Decrypt For $params,\ C = \langle U, V \rangle \in \mathcal{C}$ a cipher-text encrypted using the public key ID and a private key $d_{ID} \in \mathbb{G}_1^*$, $M = V \oplus H_2(\hat{e}(d_{ID}, U)) \in \mathcal{M}$ is returned.

To note is that the masks used during encryption and decryption are the same since:

$$\hat{e}(d_{ID}, U) = \hat{e}(sQ_{ID}, rP) = \hat{e}(Q_{ID}, P)^{sr} = \hat{e}(Q_{ID}, P_{pub})^r = g_{ID}^r$$

Each of these four algorithms must satisfy the standard consistency constraint: when d_{ID} is a private key generated by the *Extract* algorithm for a given ID as public key, then

$$\forall M \in \mathcal{M} : \ Decrypt(params, C, d_{ID}) = M \ where \ C = Encrypt(params, ID, M)$$

3.3 Adversary Models

The definition of the adversary models, as defined in [Gol96], follows.

The Semi-honest Adversary Model The *semi-honest* model of an adversary assumes that both parties act according to their defined action in a protocol, but they may store all intermediate results and computations, and use this to learn more than what they would know after working in an ideal model. In this case, only the initiator *A* of the protocol receives an input from the responder *B*. Thus, *A*'s anonymity means that the responder *B* is not able to determine if the sender has different inputs. *B*'s anonymity means that *A* will not get more information from its output than that defined by the protocol.

The Malicious Adversary Model A *malicious* model permits an adversary to not follow the protocol's description. In this case, this would mean the responder *B* not replying to *A*'s request, sending random generated elements or aborting the protocol at any point. Security in the malicious adversary model means not revealing more information than in an ideal model.

4 Protocol Description

The set for each entity that takes part in the protocol is assigned to it by a central secure and TA. The initialization of each party consists of the following steps:

4.1 Initialization Stage

Both the network owner (the TA) and each of the nodes have to perform an initial setup procedure. The two entities are described as follows:

Trusted Authority *Bilinear Maps Setup* The TA starts by generating the system parameters $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2 \rangle$ and retaining the private master-key g_{TA} as previously described in Section 3.2. Additionally, a cryptographically safe symmetric encryption scheme E is included into the system parameters.

Element List Setup The TA defines a large enough domain, from which the elements are randomly chosen. The TA also sets the global constant value k_c, which represents the maximal cardinality of each set loaded on a node.

Nodes *Pseudonyms Generation for Each Node* On every node, the TA deploys the public system parameters. It then generates and loads a set of pairs of the form $\{PS_i^{A_i}, SP_i^{A_i}\}$, where $PS_i^{A_i}$ is a collision-resistant pseudonym associated to node A_i, and $SP_i^{A_i} = g_{TA}H_1(PS_i^{A_i})$ is the *secret point* associated to it. [ZLLF06] and [BDS+03] have a more elaborate description of such a key agreement; since we have a similar implementation, the same notation is used.

Element List Loading The TA loads, on every node A_i, a set of pairs of the form $\{x_i^{A_i}, CS_i^{A_i}\}$, where $CS_i^{A_i}$ represents the commitment signature (CS) for the value $x_i^{A_i}$, with the property that for any two nodes, only identical elements will also have a common CS. Formally, for any two nodes A_i and A_j, $CS_i^{A_i} = CS_j^{A_j}$ iff $x_i^{A_i} = x_j^{A_j}$. The commitment signature has two roles: first, to prove that an element was created and distributed by the TA. Second, it is used to prove that a certain set of elements was created and assigned to a specific node by the TA.

Because the cardinality of each set is set to size k_c, the TA will also generate unique random values and corresponding CSs to fill all the sets to the requested size, in those cases when the representation of a node's functionality would require less than k_c different elements.

4.2 Message Exchange

There are two distinct phases (see Figure 1) needed by the protocol: the first stage creates *trust* between the two communicating parties. The second stage handles the private set intersection and elements obfuscation.

Network Affiliation Proof and Shared Key Creation The network affiliation proof and session key agreement both work by using an adapted IBE scheme, like the similar one used in [ZLLF06] and [BDS+03]: let there be two nodes, *A* and *B*. The messages exchange takes place as follows:

1. *A* initiates the protocol by selecting an unused pseudonym of his PS_i^A , together with a random generated number n_A, and sending both to *B*:
   $A \rightarrow B: \ PS_i^A, \ n_A$

2. Upon receiving the message, *B* generates a random number n_B, selects an unused pseudonym PS_j^B, and computes the key $K_{B \rightarrow A} = \hat{e}(H_1(PS_i^A), SP_j^B)$,
   which forms the value
   $V_{B \rightarrow A} = H_2(n_A \,||\, n_B \,||\, 0 \,||\, K_{B \rightarrow A})$.

The message is then:
$A \leftarrow B:\ PS_j^B, n_B, V_{B\to A}$

3. *A* generates his own key $K_{A\to B} = \hat{e}(H_1(PS_j^B), SP_i^A)$,
then checks if
$H_2(n_A \,||\, n_B \,||\, 0 \,||\, K_{A\to B}) \stackrel{?}{=} V_{B\to A}$.

4. The equation holds, according to Section 3.1, only if both nodes, *A* and *B*, had their pseudonym and corresponding secret key issued by the same TA, owner of g_{TA}:

$$\begin{aligned} K_{B\to A} &= \hat{e}(H_1(PS_i^A), SP_j^B) \\ &= \hat{e}(H_1(PS_i^A), g_{TA}H_1(PS_j^B)) \\ &= \hat{e}(g_{TA}H_1(PS_j^A), PS_i^B) \\ &= \hat{e}(SP_j^A), PS_i^B) \\ &= K_{A\to B} \end{aligned}$$

If the verification succeeds, *A* knows that *B* is under the same TA as him, and sends:
$V_{A\to B} = H_2(n_A \,||\, n_B \,||\, 1 \,||\, K_{A\to B})$.
$A \to B:\ V_{A\to B}$

5. *B* performs the same test as in the previous step and, if it succeeds, it will trust *A*.

Private Set Intersection As a result, not only did *A* and *B* authenticate themselves as *trusted* (in a certain degree) nodes, they also established a shared key $K_{A\to B} = K_{B\to A} = K$, that is used for establishing the private set intersection as follows:

1. Using the public symmetric encryption scheme *E*, *B* computes for every element of his set the following output: $E_{CK_j^B}(x_j^B)$, where $CK_j^B = CS_j^B \oplus K$ is the commitment key.
It then sends the resulting set $set(B)^K = \{E_{CK_j^B}(x_j^B),\ \forall x_j^B \in set(B)\}$ to *A*, ordered lexicographically.

2. Upon receiving $set(B)_s^K$, *A* creates its corresponding set $set(A)^K$ by computing: $E_{CK_i^A}(x_i^A)$, where $CK_i^A = CS_i^A \oplus K$ for each of its elements. The intersection of $set(A)^K$ and $set(B)^K$ will then contain only their common elements.

4.3 Security

The security requirements are stated for the private set intersection part of the protocol - regarding the anonymous authentication and key agreement stage, the proof is given by the fact that given the difficulty of solving DLP in $\mathbb{G}_1$, given any pair $\{PS_i^{A_i}, SP_i^{A_i}\}$, it is computationally unfeasible to deduce g_{TA}. Detailed proofs on bilinear maps can be found in [BF03].

Network Affiliation Proof and Shared Key Creation

1. $A \rightarrow B:\ PS_i^A,\ n_A$
2. $B: K_{B\rightarrow A} = \hat{e}(H_1(PS_i^A), SP_j^B)$
   $V_{B\rightarrow A} = H_2(n_A \mid\mid n_B \mid\mid 0 \mid\mid K_{B\rightarrow A})$
   $A \leftarrow B:\ PS_j^B, n_B, V_{B\rightarrow A}$
3. $A: K_{A\rightarrow B} = \hat{e}(H_1(PS_j^B), SP_i^A)$
   $H_2(n_A \mid\mid n_B \mid\mid 0 \mid\mid K_{A\rightarrow B}) \stackrel{?}{=} V_{B\rightarrow A}$
   $V_{A\rightarrow B} = H_2(n_A \mid\mid n_B \mid\mid 1 \mid\mid K_{A\rightarrow B})$
   $A \rightarrow B:\ V_{A\rightarrow B}$
4. $B: V_{A\rightarrow B} \stackrel{?}{=} H_2(n_A \mid\mid n_B \mid\mid 1 \mid\mid K_{B\rightarrow A})$

Private Set Intersection
A and B share the common key $K_{A\rightarrow B} = K_{B\rightarrow A} = K$
E is a symmetric encryption scheme

1. $B:\ set(B)^K = \{\forall x_j^B \in set(B),\ E_{CK_j^B}(x_j^B),\ with\ CK_j^B = CS_j^B \oplus K\}$
   $B \rightarrow A:\ set(B)^K$
2. $A: set(A)^K = \{\forall x_i^A \in set(A),\ E_{CK_i^A}(x_i^A),\ with\ CK_i^A = CS_i^A \oplus K\}$
   $set(A)^K \cap set(B)^K$ will contain A's and B's common elements.

Figure 1: Message Exchange

The Semi-honest Case The protocol satisfies the *correctness* requirement in the semi-honest model, as *A* receives an encryption of an element x_i as long as it is part of $A_s \cap B_s$, where A_s is the set of elements in *A* and B_s is the set of elements in B. For all the rest *A* receives, in the ideal cipher model, what looks like random values .

A's *privacy* requirement is automatically preserved, as it doesn't expose any of its data-set elements to *B* in the current scenario.

B's *privacy*, given an A^* that operates in an ideal model and given *A* that operates in a semi-honest model, means that their view of the protocol is indistinguishable. Informal, the case is represented by a curious adversary in the following example: by issuing a valid request to another node, the attacker receives the encrypted list of its elements, then exhaustively goes through the entire domain from which the values x and CSs were generated, an computes all possible results. If the domains are small enough, the attacker succeeds in finding all of the senders elements. As the assumption is that the TA uses a sufficiently large domain, the proof is direct and holds as long as the chosen *E* is a secure symmetric key scheme.

The Malicious-Adversaries Case The meaningful attack of a malicious node means trying generating random or carefully crafted elements instead of using the ones assigned by the certification authority. More specifically, in the current context, without knowing the signature key used to generate CS for a value x, it means having to find two correct random numbers s, t with the property that for a i, $(x_i, EC_i) = (s, t)$. The same proof as in the *semi-honest case* applies.

5 Conclusions and Future Work

We presented an resource efficient, secure private set interaction protocol, while assuming certain conditions, meant to be used in WSNs, more specifically in the context of AAL environments. The protocol also authenticates trusted nodes and performs all the communication anonymously (sender and receiver anonymity). It is secure against malicious attacks by using an efficient, symmetric encryption schemes where the actual value, as well as they key (commitment signature) are both unknown to a malicious attacker.

The protocol is not suitable for *disjointness tests*, as the cardinality of the intersection set, as well as the elements that form it, are known. We hope to offer a viable, WSN solution to this this problem, too.

A desirable feature is the ability of a party to detect the cases when the partner does not contain a valid set of elements prior to performing the private set intersection routine. By our definition, valid means assigned by the TA. This would prevent nodes from trying to assume a false list of elements or to request a list without having a certified one of their own. We will present our proposed solution in the near future.

References

[19802] Federal Information Processing Standards Publication 198. The Keyed-Hash Message Authentication Code (HMAC), 2002.

[AES03] Rakesh Agrawal, Alexandre Evfimievski, and Ramakrishnan Srikant. Information Sharing Across Private Databases. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 86–97. ACM, 2003.

[BDS+03] Dirk Balfanz, Glenn Durfee, Narendar Shankar, Diana Smetters, Jessica Staddon, and Hao-chi Wong. Secret Handshakes from Pairing-based Key Agreements. In *IEEE Symposium on Security and Privacy*, pages 180–196. IEEE, 2003.

[BF03] Dan Boneh and Matt Franklin. Identity-Based Encryption from the Weil Pairing. In *SIAM J. of Computing*, pages 586–615, 2003. Extended abstract in Crypto'01.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In *Advances in Cryptology*, EUROCRYPT '04, pages 1–19. Springer-Verlag, 2004.

[Gam85] Taher El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *IEEE Transactions on Information Theory*, pages 469–472, 1985.

[Gol96] Oded Goldreich. Adaptively Secure Multi-party Computation. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, pages 639–648. ACM, 1996.

[Jou00] Antoine Joux. A One Round Protocol for Tripartite Diffie-Hellman. In *Proceedings of the 4th International Symposium on Algorithmic Number Theory*, pages 385–394. Springer-Verlag, 2000.

[MVO96] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[Pai99] Pascal Paillier. Public-key Cryptosystems based on Composite Degree Residuosity Classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT '99, pages 223–238. Springer-Verlag, 1999.

[Ste05] H. Steg. Ambient Assisted Living - European Overview Report, September 2005.

[TIETF06] RFC 4493 The Internet Engineering Task Force, IETF. The AES-CMAC Algorithm, 2006.

[ZLLF06] Yanchao Zhang, Wei Liu, Wenjing Lou, and Yuguang Fang. MASK: Anonymous On-Demand Routing in Mobile Ad Hoc Networks. In *Transactions on Wireless Communications*, pages 2376–2385. IEEE, 2006.