A Probabilistic Key Agreement Scheme for Sensor Networks without Key Predistribution*

Vasiliki Liagkou[1,3], Effie Makri[4], Paul Spirakis[1,3], and Yannis Stamatiou[2,3]

[1] University of Patras, Dept. of Computer Engineering, 26500, Rio, Patras, Greece
[2] Mathematics Department, 451 10, Ioannina, Greece
istamat@uoi.gr
[3] Research and Academic Computer Technology Institute, N. Kazantzaki, University of Patras, 26500, Rio, Patras, Greece
[4] University of the Aegean, Department of Mathematics, 83000, Karlovassi, Samos, Greece

Abstract. The dynamic establishment of shared information (e.g. secret key) between two entities is particularly important in networks with no pre-determined structure such as wireless sensor networks (and in general wireless mobile ad-hoc networks). In such networks, nodes establish and terminate communication sessions dynamically with other nodes which may have never been encountered before, in order to somehow exchange information which will enable them to subsequently communicate in a secure manner. In this paper we give and theoretically analyze a series of protocols that enables two entities that have never encountered each other before to establish a shared piece of information for use as a key in setting up a secure communication session with the aid of a shared key encryption algorithm. These protocols do not require previous pre-distribution of candidate keys or some other piece of information of specialized form except a small seed value, from which the two entities can produce arbitrarily long strings with many similarities.

Keywords: Key agreement, key predistribution, mobile ad-hoc networks.

1 Introduction

Wireless Sensor Networks (WSNs) have some constraints, with regard to battery life, processing, memory and commnication ([1]) capacity, and as such are deemed unsuitable for public crypto-based systems. Thus, symmetric key cryptosystems are more appropriate for these types of networks, but lead to problems with key distribution. These problems are mitigated with key *pre-distribution schemes*, in which candidate keys are distributed to members of the network before the start communication. Many innovative and intuitive key pre-distribution

* This work was partially supported by the European Union project ABC4Trust (Attribute-based Credentials for Trust) funded within the context of the 7th Research Framework Program (FP7).

H. Jahankhani et al. (Eds.): ICGS3/e-Democracy 2011, LNICST 99, pp. 72–79, 2012.

schemes for WSNs have been proposed for solving the problem of key distribution in sensor networks. On the two ends of the spectrum are key pre-distribution schemes that use a single master key as the encryption key distributed amongst all the nodes, and all pairwise keys, where a unique key exists for every pair of sensors. The former provides the most efficient usage of memory and scales well, but an attack on one node compromises the whole network, whereas the latter provides excellent resilience but does not scale well. In addition, schemes exist which are in essence probabilistic, relying on the fact that any two neighbouring nodes have some probability p of successfully completing key establishment.

Some such schemes (presented in [2–8]) pre-suppose that the sensor nodes have been loaded with some pre-existing information (i.e. the key, or sets of keys) prior to network deployment, except for Liu and Cheng ([9]). They propose a self-configured scheme whereby no prior knowledge is loaded onto the sensor nodes, but shared keys are computed amongst the neighbours.

In this paper, we propose a key agreement scheme whereby network nodes are not pre-loaded with candidate keys, but generate pairs of symmetric keys from two, initially, random bits strings. The initial research conducted ([10]) proposed a protocol that involved the examination of random positions of subsets of size k, and the elimination of a random position if the two bit strings were found to disagree on more than half the examined positions. In that paper, however, the nodes cannot secretly compute the number of differing positions, a problem that is resolved in the present paper using secret circuit computations. In addition, the present protocols do not eliminate differing bits but flips them, depending on the number of bit difference in the examined subset of k bits. This leads to a different stochastic process that called for a different theoretical analysis.

2 The Bit-Similarity Problem

Two entities, say 0 and 1, initially possess an N-bit string, X_N^0 and X_N^1 respectively. The entities' goal is to cooperatively transform their strings so as to increase the percentage of positions at which their strings contain the same bits, which we denote by $X(i)$, with i being the time step of the protocol they execute. Then $X(0)$ is the initial percentage of the positions at which the two strings are the same. Below we provide a randomized protocol in which the two entities examine randomly chosen subsets of their strings in order to see whether they differ in at least half of the places. If they do, one of the entities (in turn) randomly flips a subset of these positions. This process continues up to a certain, predetermined number of steps. The intuition behind this protocol is that when two random substrings of two strings differ in at least half of their positions, then flipping some bits at random in one of the substrings is more likely to increase the percentage of similarities between the strings than to decrease it. In the description of the protocol $X_N^c[S]$ denotes a substring of string X_N^c defined by the position set S. Protocol for user $U_c, c = 0, 1$ Protocol parameters known to both communicating parties: (i) k, l, the subset sizes, (ii) T, the number of protocol execution steps, (iii) the index (bit position) set N, (iv) The circuit

C with which the two entities jointly compute whether there are at least $\lceil k/2 \rceil$ similarities between randomly chosen subsets of their strings.

$i \leftarrow 1$ /* The step counter. */
while $i \leq T$ /* T is a predetermined time step limit.*/
  begin /* while */
    $S \leftarrow \mathrm{JOINT\_RAND}(k, \{1, \ldots, N\})$/* Shared random set of k positions. See text. */
    same_pos $\leftarrow C(X_N^c[S], X_N^{(c+1 \mod 2)}[S])$ /* A secret computation of number of positions with same contents). */
    if (same_pos $\geq \lceil \frac{k}{2} \rceil$ **and** odd$(i + c)$) **then** /* Users 0 and 1 alternate. */
      begin
        $SF \leftarrow \mathrm{RAND}(l, S)$ /* Random set of l positions from within S to be flipped by the user whose turn it is to flip. */
        flip the bits of $X_N^c[SF]$
      end
    SYNCHRONIZE /* Users 0 and 1 wait to reach this point simultaneously */
    $i \leftarrow i + 1$
  end/* while */

3 Secret Two-Party Function Computation

During the execution of the protocol, it is necessary for the two communicating parties to see whether they agree on at least half of the positions they have chosen to compare (line 13-14 of the protocol). Thus, the two parties need to perform a computation: compute the number of positions on which the corresponding bits in the two chosen subsets of k bits are the same. This is an instance of an important, general problem in cryptography: *Secure Computation.* More formally, let A and B be two parties with inputs of n_A and n_B bits respectively. The objective is to jointly compute a function $f : \{0,1\}^{n_A} \times \{0,1\}^{n_B} \rightarrow \{0,1\}$ on their inputs. The issue, here, is that A and B cannot, simply, exchange their inputs and compute the function since they will learn each other's inputs, something that is not desirable in a secure computation setting. More importantly, even it A and B are willing to share their inputs, they would not allow an eavesdropper to acquire these inputs too. This leads to the problem of *secure function computation.* In our context, we consider the following two Boolean functions:

$$f_r : \{0,1\}^k \times \{0,1\}^k \rightarrow \{0,1\} \text{ with } w_A, w_B \in \{0,1\}^k$$

$$\text{and } 0 \leq r \leq k \ \ : f(w_A, w_B) = \begin{cases} 1 & \text{if } X(w_A, w_B) \geq r; \\ 0 & \text{otherwise.} \end{cases}$$

We are interested in $r = \lceil \frac{k}{2} \rceil$.

$$f_X : \{0,1\}^k \times \{0,1\}^k \rightarrow \{0,1\}^{\lceil \log_2(k) \rceil} \text{ with } w_A, w_B \in \{0,1\}^k :$$
$$f_X(w_A, w_B) = x, \text{ with } x = X(w_A, w_B) \text{ written in binary.}$$

The function f_X is, strictly, an ordered tuple $(f_X^0, f_X^1, \ldots, f_X^{\lceil \log_2(k) \rceil - 1})$ of $\lceil \log_2(k) \rceil$ 1-bit Boolean functions, where the function f_X^i computes the ith most significant bit of $x = X(w_A, w_B)$ (with $i = 0$ we take the most significant bit and with $i = \lceil \log_2(k) \rceil - 1$ we take the least significant bit). Using techniques from oblivious function computation (see [11] for a survey on these techniques), we can prove that the computation of f_r and f_X can be done with randomized protocols using $O(|C_{f_r}|)$ and $O(|C_{f_X}|)$ communication steps respectively, with C_{f_r} and C_{f_X} being the Boolean circuits that are employed for the computation of f and f_X respectively. Since both f_r and f_X are easily seen to be polynomial time computable Boolean functions, we can construct for their computations circuits of size polynomial in their input sizes, i.e. circuits C_{f_r} and C_{f_X} such that $|C_{f_r}| = O(k^{c_1})$ and $|C_{f_X}| = O(k^{c_2})$, with constants $c_1, c_2 \geq 0$. Since k is considered a fixed constant, we conclude that we can compute f_k and f_X in a constant number of rounds. The number of random bits needed by each step of the randomized protocol is in both cases $O(k)$ and, thus, constant. To sum up, the functions f_r and f_X can, both, be evaluated on two k-bit inputs w_A, w_B held by two parties A, B using a constant number of rounds and a constant number of uniformly random bits. In what follows, we will assume that the communicating parties use the function $f_{\lceil \frac{k}{2} \rceil}$. With regard to the required randomness, we assume that each of the two parties has a true randomness source, i.e. a source of uniformly random bits. Such a randomness source can be easily built into modern devices. This randomness source is necessary in order to implement the randomized oblivious computation protocols for the computation of the function $f_{\lceil \frac{k}{2} \rceil}$. In addition, it will be used in order to produce the randomly chosen positions, are required by Step 6 of the protocol. Since each position can range from 1 up to N (the string size), to form a position index we need to draw $\lceil \log_2(N) \rceil$ random bits. Alternatively, if we allow the two parties to share a small (in relation to N) seed, they can produce the random positions in synchronization and, thus, avoid sending them over the communication channel.

4 Theoretical Analysis of the Protocol

In order to track the density of positions where two strings agree, we will make use of Wormald's theorem (see [12]) to model the probabilistic evolution of the protocol described in Section 2 using a deterministic function which stays provably close to the real evolution of the algorithm. The theorem in [12] essentially states is that if we are confronted with a number of (possibly) interrelated random variables (associated with some random process) such that they satisfy a Lipschitz condition and their expected fluctuation at each time step is known, then the value of these variables can be approximated using the solution of a system of differential equations. Furthermore, the system of differential equations results directly from the expressions for the expected fluctuation of the random variables describing the random process. We will first prove a general lemma that gives the probability of increasing the similarity between two strings through flipping, at random, the contents of a certain number of positions.

Lemma 1. *Let w_1, w_2 be two strings of 0s and 1s of length k. Let also j, $0 \leq j \leq k$, be the number of places in which the two strings differ. Then if l positions of one string are randomly flipped, the probability that s of them are differing positions is the following:* $P_{k,j,l,s} = \frac{\binom{j}{s}\binom{k-j}{l-s}}{\binom{k}{l}}$.

Proof. In the above equation the denominator is the number of all subsets of positions of cardinality l of the k string positions while the numerator is equal to the number of partitions of the l chosen positions such that s of them fall into the j differing positions and the remaining $l - s$ fall into the remaining $k - j$ non-differing positions of the two strings. Thus their ratio gives the desired probability. □

The following lemma, which is easy to prove based on general properties of the binomial coefficients, provides a closed form expression for a sum that will appear later in some probability computations.

Lemma 2. *The following identity holds:* $\sum_{s=0}^{l}(2s-l)\frac{\binom{j}{s}\binom{k-j}{l-s}}{\binom{k}{l}} = \left(\frac{2j}{k} - 1\right) l$.

We will now derive the deterministic differential equation that governs the evolution of the random variable $X(i)$ manipulated by the protocol in Section 2 using Wormald's theorem.

Theorem 1. *The differential equation that results from the application of Wormald's theorem on the quantity $X(i)$ (places of agreement at protocol step i) as it evolves in the agreement protocol is the following:*

$$\mathbf{E}[X(i+1) - X(i)] = \sum_{j=\lceil \frac{k}{2} \rceil}^{k} \sum_{s=0}^{l} [(s-(l-s))P_{k,j,l,s}]P_{n,n-X(i),k,j}.$$

Proof. We will determine the possible values of the difference $X(i+1) - X(i)$ along with the probability of occurrence for each of them. The protocol described in Section 2 flips l positions within the k examined positions, whenever these k positions contain $j \geq \lceil \frac{k}{2} \rceil$ differing positions in the two strings. From the flipped l positions, is s of them $(0 \leq s \leq l)$ are disagreement positions, then the two strings will have gained s agreement positions, losing $l - s$. The net total is $s - (l - s)$. The probability that this total occurs, for a specific value of s and a specific value of j is equal to $P_{k,j,l,s}P_{n,n-X(i),k,j}$. Summing up over all possible values of s, j we obtain (in Theorem 1). □

Corollary 1. *The following holds:*

$$\mathbf{E}[X(i+1) - X(i)] = \sum_{j=\lceil \frac{k}{2} \rceil}^{k} l\left(\frac{2j}{k} - 1\right) \frac{\binom{n-X(i)}{j}\binom{X(i)}{k-j}}{\binom{n}{k}}.$$

Proof. The above Equation follows from Theorem 1 using the Equation of Lemma 1 with $k = n, l = k, s = j, n - j = X(i)$, in conjunction with Lemma 2.

Corollary 2. *Using Wormald's Theorem (in [12]), the evolution of the random variable $X(i)$ whose mean fluctuation is given in Corollary 1 can be approximated by the following differential equation:*

$$\frac{dx(t)}{dt} = \sum_{j=\lceil \frac{k}{2} \rceil}^{k} l \left(\frac{2j}{k} - 1 \right) \binom{k}{j} [1 - x(t)]^j x(t)^{k-j}.$$

Proof. By applying the approximation $\binom{N}{k} = \frac{N^k}{k!} \left(1 + O\left(\frac{1}{N}\right)\right)$ of the binomial coefficients which is valid for $k = O(1)$ on the three binomials which appear on the the right-hand side of Equation in Corollary 1, we obtain the following: $\frac{\binom{n-X(i)}{j}\binom{X(i)}{k-j}}{\binom{n}{k}} \simeq \binom{k}{j} \left(1 - \frac{X(i)}{n}\right)^j \left(\frac{X(i)}{n}\right)^{k-j}$. Using Wormald's theorem, we make the correspondence $x(t) = \frac{X(i)}{n}$ and $\frac{dx(t)}{dt} = \mathbf{E}[X(i+1) - X(i)]$, which results in the required differential equation of Corollary 2. □

5 Efficiency of the Protocol

From Corollary 2 we see that the percentage of similar positions, represented by the function $x(t)$, is a monotone increasing function since its first derivative is always positive. In what follows, we will estimate how fast this percentage increases depending on its initial value $x(0)$ as well as the parameters l and k.

Lemma 3. *The solution $x(t)$ to the differential equation given in Corollary 2 is monotone increasing.*

Proof. From the differential equation, we see that the first derivative of the function $x(t)$, which is equal to the right-hand side of the differential equation, is strictly positive, since $0 < x(t) < 1$. Thus, the function $x(t)$ is monotone increasing. □

Lemma 4. *Let $x(t_1)$ be the value of the function $x(t)$ at time instance t_1 and $x(t_2)$ be the value at time instance t_2, $t_1 < t_2$. Let, also, $c(t_1)$ be the absolute value of the point at which the tangent line to the point $(t_1, x(t_1))$ of $x(t_1)$ cuts the t-axis and $c(t_2)$ the corresponding value for t_2. Let, also, $p(x) = \sum_{j=\lceil \frac{k}{2} \rceil}^{k} l \left(\frac{2j}{k} - 1\right) \binom{k}{j} (1-x)^j x^{k-j}$. Then,$p(x(t_1)) = \frac{x(t_1)}{c(t_1)+t_1}, p(x(t_2)) = \frac{x(t_2)}{c(t_2)+t_2}$.*

Proof. Let ϵ_1 and ϵ_2 be the two tangent lines to the function $x(t)$ at the points $(t_1, x(t_1))$ and $(t_2, x(t_2))$ respectively, as shown in Figure 1. Due to the monotonicity of $x(t)$, the points at which the two lines intersect with the t-axis are negative. Let $c(t_1)$ and $c(t_2)$ be the absolute values of these two points for lines ϵ_1 and ϵ_2 respectively. Then from the two right angle triangles that are formed we have $\tan(\phi_1) = \frac{x(t_1)}{c(t_1)+t_1}$ and $\tan(\phi_2) = \frac{x(t_2)}{c(t_2)+t_2}$. From the definition of the derivative, $\tan(\phi_1) = \frac{dx(t)}{dt}|_{t_1}$ and $\tan(\phi_2) = \frac{dx(t)}{dt}|_{t_2}$. From the Equations of Corollary 2 and Lemma 4, we have $\frac{dx(t)}{dt} = p(x(t))$ and, thus, the statement of the lemma follows. □

Theorem 2. *Let t' be the time instance at which $x(t') = hx(0)$, with $1 \leq h \leq \frac{1}{x(0)}$. Then, $t_2 \leq \frac{x(0)}{p(hx(0))} \cdot (h-1)$.*

Proof. We set $t_1 = 0, t_2 = t'$ and $x(t') = hx(0)$ in Lemma 4 and we obtain, the following: $p(x(0)) = \frac{x(0)}{c(0)}, p(hx(0)) = \frac{hx(0)}{c(t')+t'}$. From these equations we obtain

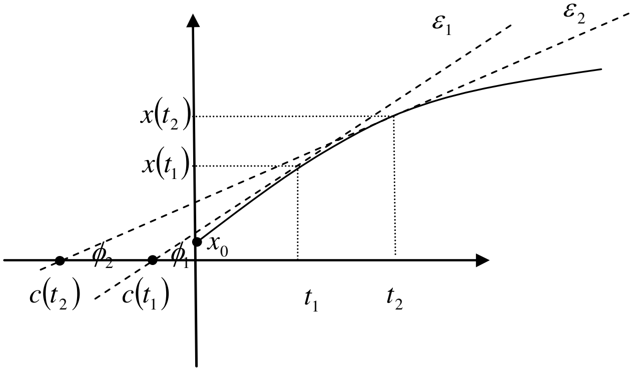

Fig. 1. The two tangent lines for the proof of the Theorem

the following: $\frac{p(x(0))}{p(hx(0))} = \frac{c(t')+t'}{hc(0)}$. Solving for t' we obtain the following: $t' = \frac{hp(x(0))c(0)}{p(hx(0))} - c(t')$.Since $p(x(0))$ is the inclination of the tangent line to $x(t)$ at the point $(0, x(0))$, it holds that $p(x(0)) = \frac{x(0)}{c(0)}$. Thus, t' becomes $\frac{hx(0)}{p(hx(0))} - c(t')$.Since $x(t)$ is monotone increasing, the point at which the tangent to this function cuts the $x(t)$-axis at any point is greater than or equal to $x(0)$ (see Figure 1). Let x_c be this point. Then $p(x(t')) = \frac{x_c}{c(t')}$ or, since $x(t') = hx(0)$, $p(hx(0)) = \frac{x_c}{c(t')}$. Since $x_c \geq x(0)$, $c(t') \geq \frac{x(0)}{p(hx(0))}$. Thus, we obtain $t' \leq \frac{hx(0)}{p(hx(0))} - \frac{x(0)}{p(hx(0))} = \frac{x(0)}{p(hx(0))} \cdot (h-1)$ which is the required. □

Lemma 5. *The following lower bounds hold for the polynomial in the first Equation of Lemma 4: If $x < 1/2$ then $p(x) \geq l[x^k + 1 - 2x]$. If $x \geq 1/2$ then $p(x) \geq \frac{l}{k}(1-x)^k \binom{k}{\lceil \frac{k}{2} \rceil} \lceil \frac{k}{2} \rceil$.*

Proof. In the first Equation of Lemma 4, allowing the sum index to cover all the range from 1 to k reduces the value of the sum since it adds negative terms. Thus $p(x) \geq l \sum_{j=1}^{k} \left(\frac{2j}{k} - 1\right) \binom{k}{j} (1-x)^j x^{k-j}$. Since the sum evaluates to $l[x^k + 1 - 2x]$ the first statement of the lemma follows. If, on the other hand, $x \geq 1/2$, then lower bound given for the first statement of the lemma is not good since it may even become negative. In this case we observe that the term $(1-x)^j x^{k-j}$ is minimized for $j = k$. Setting $j = k$ in the first Equation of Lemma 4, we obtain the second statement of the lemma. □

Corollary 3. *The following bounds hold for the time instance t': (i)If $hx(0) < 1/2$ then $t' \leq \frac{x(0)(h-1)}{l[hx(0))^k + 1 - 2hx(0)]}$.(ii)If $hx(0) \geq 1/2$ then $t' \leq \frac{kx(0)(h-1)}{l[1-hx(0)]^k \binom{k}{\lceil \frac{k}{2} \rceil} \lceil \frac{k}{2} \rceil}$.*

From the first inequality of Corollary 3 we see that the percentage of similarities grows fast, if we start from $x(0)$ aiming at $hx(0)$, with $h \geq 1$ and $hx(0) < 1/2$ is only a very coarse upper bound). If the target is, however, at $hx(0)$ with $x \geq 1/2$ the upper bound is not good as the denominator tends to 0 fast. However, since this denominator is simply the first derivative of $x(t)$ at $hx(0)$, this derivative fast tends to 0 if $hx(0) \geq 1/2$ and, thus, the tangent at this point tends to become parallel to the t-axis. Thus, we have again fast convergence.

6 Conclusions

In this paper we described a series of protocols that can be used in order to increase the percentage of similarities between two strings held by two communicating parties without revealing their values. The propose protocols are, in fact, general and may be used in any situation involving either wireless or conventional networks in which there is no trusted third party or key management authority among the network nodes.

References

1. Carman, D.W., Kruus, P.S., Matt, B.J.: Contstraints and approaches for distributed sensor network security. Technical Report 010, NAI Labs, The Security Research Division Network Associates, Inc. (2000)
2. Eschenauer, L., Gligor, V.: A key-management scheme for distributed sensor networks. In: 9th ACM Conf. Computing and Comm. Security (CCS 2002), pp. 41–47 (2002)
3. Chan, H., Perrig, A., Song, D.: Random key predistribution schemes for sensor networks. In: IEE Symposium of Privacy and Security, pp. 197–213 (2003)
4. Chan, S.-P., Poovendran, R., Sun., M.-T.: A key management scheme in distributed sensor networks using attack probabilities. In: IEEE Global Telecommunications Conference, GLOBECOM (2005)
5. Chan, H., Perrig, A.: Pike: Peer intermediaries for key establishment in sensor networks. In: Proceedings of the IEEE Infocom (2005)
6. Du, W., Deng, J., Han, Y.S., Varshney, P.: A pairwise key pre-distribution scheme for wireless sensor networks. In: 10th ACM Conference on Computer and Communications Security, pp. 42–51. ACM Press (2003)
7. Du, W., Deng, J., Han, Y.S., Chen, S., Varshney, P.: A key management scheme for wireless sensor networks using deployment knowledge. In: IEEE Infocom, pp. 586–597 (2004)
8. Blom, R.: An Optimal Class of Symmetric Key Generation Systems. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) EUROCRYPT 1984. LNCS, vol. 209, pp. 335–338. Springer, Heidelberg (1985)
9. Liu, F., Cheng, X.: A self-configured key establishment scheme for large-scale sensor networks. In: 3rd IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), pp. 447–456 (2006)
10. Makri, E., Stamatiou, Y.: Distributively increasing the percentage of similarities between strings with application to key agreement. In: 5th International Conference on AD-HOC Networks and Wireless, pp. 211–223 (2006)
11. Cramer, R.: Introduction to Secure Computation. In: Damgård, I.B. (ed.) EEF School 1998. LNCS, vol. 1561, pp. 16–42. Springer, Heidelberg (1999)
12. Wormald, N.: The differential equation method for random graph processes and greedy algorithms. Annals of Applied Probability 5, 1217–1235 (1995)