Key propagation in wireless sensor networks

Gianluca Dini, Lanfranco Lopriore

Dipartimento di Ingegneria dell'Informazione, Università di Pisa, via G. Caruso 16, 56126 Pisa, Italy
E-mail: {g.dini | l.lopriore}@iet.unipi.it

Abstract — With reference to a network consisting of sensor nodes connected by wireless links, we approach the problem of the distribution of the cryptographic keys. We present a solution based on communication channels connecting sequences of adjacent nodes. All the nodes in a channel share the same key. This result is obtained by propagating the key connecting the first two nodes to all the other nodes in the channel. The key propagation mechanism is also used for key replacement, as is required, for instance, in group communication to support forms of forward and backward secrecy, when a node leaves a group or a new node is added to an existing group.

Keywords — cryptographic key, key propagation, key replacement, sensor network.

1. INTRODUCTION

We shall refer to a network consisting of sensor nodes connected by wireless links. We make no hypothesis on the network topology, which is subject to dynamic modifications due, for instance, to node mobility and changes of the wireless transmission strength of the nodes (if the transmission power increases, new links are generated between contiguous nodes; if the transmission power decreases, existing links tend to disappear [1], [2]). Moreover, new nodes can be added to the network, and existing nodes can leave the network (e.g. as a consequence of a battery exhaustion).

In a network of this type, stringent limitations exist in terms of hardware complexity, computational power and energy consumption [3], [4]. In fact, the design of a wireless sensor network is largely different from that of a traditional wired-line or wireless network. As a consequence of the energy costs of wireless communications, the number of messages transmitted across the network must be kept low, for instance. Similarly, the processor time has an energy cost that must be kept to a minimum. These constraints, and the modern tendency to support new applications taking advantage of an always increasing number of nodes [5], complicate the incorporation of security protocols, as are necessary in most applications because of the lack of physical protection and the unattended positioning [6], [7].

A relevant problem is the distribution to nodes of the cryptographic keys, which are required to support basic services such as data integrity, confidentiality and authenticity [8], [9]. This problem has been widely investigated and several schemes have been proposed for link-level key establishment [10], [11], node-to-node key agreement [12], and group rekey-

ing [13], [14]. The so-called *basic scheme* [10] is a well-known example of a probabilistic key sharing scheme. In the basic scheme, a large pool $\mathcal{P}$ of key values is generated in the key pre-distribution phase; we shall use the term *global keys* to denote these key values. Each node is pre-loaded with a set of p global keys, chosen at random out of $\mathcal{P}$. Two network nodes can communicate if they share at least one global key; the probability that this actually happens is a function of quantity p and the cardinality of $\mathcal{P}$, e.g. the probability is 0.5 if $p = 75$ and $\mathcal{P}$ contains 10,000 key values [10]. For adequate levels of network density (i.e. the number of nodes in a neighborhood), the probability that a node is disconnected from the network is negligible.

Of course, probabilistic key sharing is unable to support sequences of three or more nodes: the probability that all these nodes share the same global key is vanishingly low. In this paper, we shall present an approach to node-to-node key agreement that supports the creation of secure *communication channels* between nodes. A communication channel connects a *start node*, an *end node* and a sequence of *intermediate nodes* in the path between the start node and the end node. All the nodes in a communication channel share the same key. This result is obtained by *propagating* the key connecting the start node and the first subsequent node to all the other nodes in the channel. Communication channels are bidirectional; they can be used to deliver reply messages as well as to transmit a message from/to any intermediate node.

Our key agreement scheme integrates with routing. Creation of a secure communication channel proceeds while the first message is routed along the path from the start node to the end node. We take advantage of a pre-existing link-level key establishment so that link keys are used to protect confidentiality and integrity of the propagating key. However, once a secure channel has been established, no hop-by-hop encryption/decryption is necessary, with evident advantages in terms of performance.

Similarly to the SPINS set of security protocols for sensor networks [12], our key management scheme achieves a form of secure key agreement; however, in SPINS a centralized approach is followed that relies on a base station playing the role of a key distribution center. Although effective, this approach suffers from a limited scalability (the whole key establishment traffic passes through the base station and the nearby sensor nodes) and a single point of failure (when the base station is unavailable, no end-to-end secure channel can be established). We overcome these limitations in a fully decentralized approach whereby the key management traffic is completely sustained by the start node, the end node and the intermedi-

ary nodes.

A similarity exists between our key propagation scheme and directed diffusion key management; in both cases, keys are propagated by localized interactions, i.e. message exchanges between neighbors. However, directed diffusion is a data centric communication paradigm, which facilitates attribute-based naming and in-network processing to reduce network traffic [15]; thus, it is particularly well suited for group key management [14], [16]. In contrast, our key propagation scheme has been conceived to fit a communication model whereby nodes are identified by name, and inter-node communication is layered on an end-to-end delivery service provided within the network. Thus, key propagation is especially well suited to end-to-end key establishment.

The rest of this paper is organized as follows. Section 2 presents our connection model with special reference to the propagation of the cryptographic keys and key replacement. Section 3 discusses the connection model from a number of salient viewpoints including group communication and storage requirements. Section 4 gives concluding remarks.

2. THE CONNECTION MODEL

We model a message as consisting of a control part and a data part. Let us refer to message *m* sent by a start node, say node A, to an end node, say node B, and let N_0, N_1,..., N_r be the intermediate nodes in the path from A to B. The control part is generally read by the intermediate nodes. This may be necessary for message routing, for instance. On the other hand, the data part is only read by the end node, B.

In a possible approach, a cryptographic key is associated with each consecutive pair of nodes and is used to establish a form of secure communication between these nodes taking advantage of a symmetric-key cipher. In this approach, let K_0 be the cryptographic key shared by nodes A and N_0; the control part of message *m* is encrypted in A and is decrypted in N_0 by using K_0. Similarly, let K_1 be the key shared by node N_0 and the subsequent node N_1; the control part of *m* is encrypted in N_0 and is decrypted in N_1 by using key K_1. The process is extended to all the subsequent nodes. The data part is encrypted in the start node A by using a cryptographic key in common with the end node B; this message part is simply forwarded by each intermediate node to the subsequent node until the end node B is reached, where it is finally decrypted.

In contrast, in our approach, we create a communication channel between nodes A and B through nodes N_0, N_1,..., N_r. A message transmitted through the communication channel is transformed to ciphertext only once, by the start node A. All the subsequent channel nodes

N_0, N_1,..., N_r possess the cryptographic key used in the transformation; they will use this key to read the control part of the message and then forward the whole message to the next node, until the end node B is reached. Of course, significant advantages follow in our approach from the point of view of the number of transformations from plaintext to ciphertext.

2.1. Cryptographic keys

We shall refer to a local network consisting of up to 2^d nodes. Two set of keys are stored in each node, the *local keys* and the *global keys*. Each key has a *name* and a *value*. The name K of a local key consists of two components, i.e. $K = <K_{node}, K_{incr}>$. Quantity K_{node} is codified in the d most significant bits of K, and is equal to the name of the node where the key was generated. This node is called the *principal* of the key, and is denoted by $P(K)$. Thus, $K_{node} = P(K)$. Quantity K_{incr}, codified in the least significant bits of K, is called the *incremental name* of K. A simple technique for generation of incremental names is a sequential generation that takes advantage of a *local key counter* in each node. At any given time, the local key counter of a given node contains the incremental name of the next local key that will be generated in that node. The local key counter is initialized to 0 when the system is generated, and is incremented by 1 after every action of local key generation taking place in that node.

Let $\mathcal{K}$ be the set of all possible key values, and let $f : \mathcal{K} \to \mathcal{K}$ be a *key conversion function* (a one-way hash function that is efficient to compute and hard to invert [17]). A *master value* v is associated with each local key K, and is used to determine the value k of this key by applying the key conversion function, i.e. $k = f(v)$. As will be shown shortly, the master value of a given key is used when the value of that key should be replaced, to certify the key replacement messages.

Global keys are utilized according to a probabilistic key sharing scheme. The name of a global key is the order number of that key in the pool of key values generated in the key pre-distribution phase. Global keys are only aimed at the propagation of the local keys.

2.2. Communication channels

Two nodes are *adjacent* if they can communicate via a direct wireless link. Adjacent nodes A and B are *connected* on local key K if both of them hold this key. The connection is denoted by $\{A, B\}_K$. In a situation of this type, a message can be sent by node A to node B encrypted by using K. Connections are bidirectional: if there exists connection $\{A, B\}_K$, then there exists connection $\{B, A\}_K$ and a message can be sent by node B to node A encrypted by using K.

The connection property can be extended to more nodes, as follows: nodes A, N_0, N_1,...,

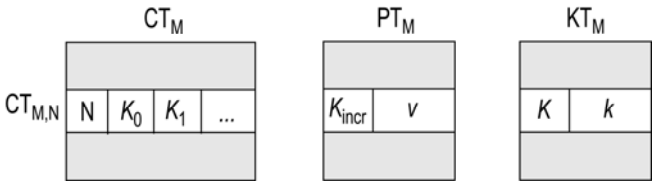

Figure 1. Tables in node M.

N_r and *B* are connected on local key *K* if they all hold this key. In a situation of this type, we have a communication channel on local key *K* that is denoted by $\{A, N_0, N_1,..., N_r, B\}_K$ and extends from node A to node B through the intermediate nodes N_0, N_1,..., N_r. Key *K* is called the *channel key*. In a situation of this type, a message can be sent by A to B encrypted by using *K*. Each intermediate node will be in the position of decrypting the message, as it holds key *K*. Communication channels are bidirectional. A communication channel can be used by any node of the channel to send a message to any other node of the channel, in both directions, by using the channel key to encrypt the message.

2.3. Key propagation

In each given node M, three tables implement a form of distribution of the information concerning keys and connections, as follows (see Figure 1):

- The *connection table* CT_M features one entry for each node N that is connected to node M. The entry $CT_{M,N}$ reserved for N contains the name of that node and a list of local keys K_0, K_1, etc.. These keys are shared by M and N, and as such they give rise to connections $\{M, N\}_{K0}$, $\{M, N\}_{K1}$,... between these nodes. This means that it is possible to send messages between M and N encrypted by using anyone of these keys. As a consequence of the bidirectionality of connections, if $CT_{M,N}$ contains a given key, then $CT_{N,M}$ (i.e. the entry reserved for node M in the connection table CT_N of node N) contains the same key.
- The *principal key table* PT_M features one entry for each key *K* for which K_{node} = M (i.e. M is the principal of *K*). The entry reserved for key *K* contains the incremental name K_{incr} of this key and the master value *v* of this key.
- The *local key table* KT_M features one entry for each local key *K* held by this node, for which $K_{node} \neq$ M (i.e. node M is *not* the principal of *K*). The entry reserved for a given key contains the name *K* and the value *k* of this key.

In the following, we shall hypothesize that the control part of a message is preceded by a plaintext, which includes the name of the key that was used to encrypt the rest of the message. In this hypothesis, let A and B be two adjacent nodes, and let us suppose that node A is aimed at sending a message *m* to node B (e.g. the routing component on A has selected B as the next

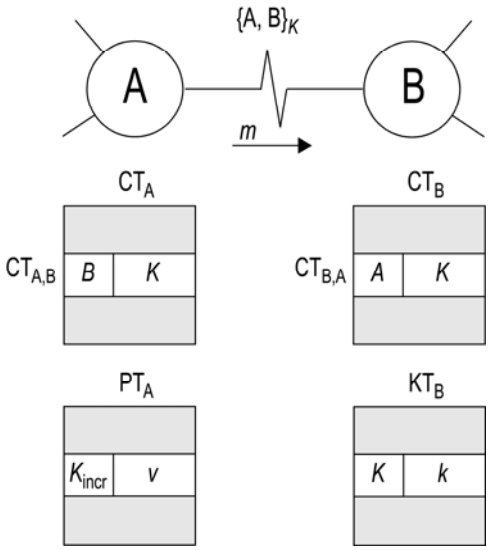

Figure 2. Sending a message from node A to node B, and final configuration of the connection tables and the key tables.

hop). Message delivery proceeds as follows (see Figure 2):

1. If an entry $CT_{A,B}$ is reserved for node B in connection table CT_A, then nodes A and B are connected. Let *K* be a local key contained in $CT_{A,B}$, corresponding to connection $\{A, B\}_K$. Taking advantage of this connection, node A sends message *m* to node B enciphered by using key *K*. Node B is in the position of deciphering *m* by using *K*.
2. If no entry is reserved for node B in CT_A, node A sends node B a message containing the names of its own global keys. On receipt of this message, node B performs a search for a global key shared with A and returns the name *G* of this key to A (*G* will be empty if no global key is shared between A and B).
3. On receipt of global key name *G* from B, node A generates a new local key name $K = \langle K_{node}, K_{incr} \rangle$ where K_{node} = A (i.e. node A is the principal of *K*) and a master value *v* for this key; pair $\langle K_{incr}, v \rangle$ is inserted into principal table PT_A. Then, node A applies the key conversion function *f* to master value *v* to obtain the value $k = f(v)$ of key *K*, and then uses key *G* to send pair $\langle K, k \rangle$ to node B. (If *G* is empty, node A performs a pairing attempt with a different adjacent node and communicates with B through this node. If all pairing attempts fail, A is disconnected from the network.)
4. Node B reserves an entry $CT_{B,A}$ for node A into its own connection table CT_B, it inserts key *K* into this entry, and inserts pair $\langle K, k \rangle$ into its own local key table KT_B. Then, B returns a positive reply message to A.
5. Node A sends message *m* to node B enciphered by using key *K*. Node B is in the position of deciphering *m* by using *K*.

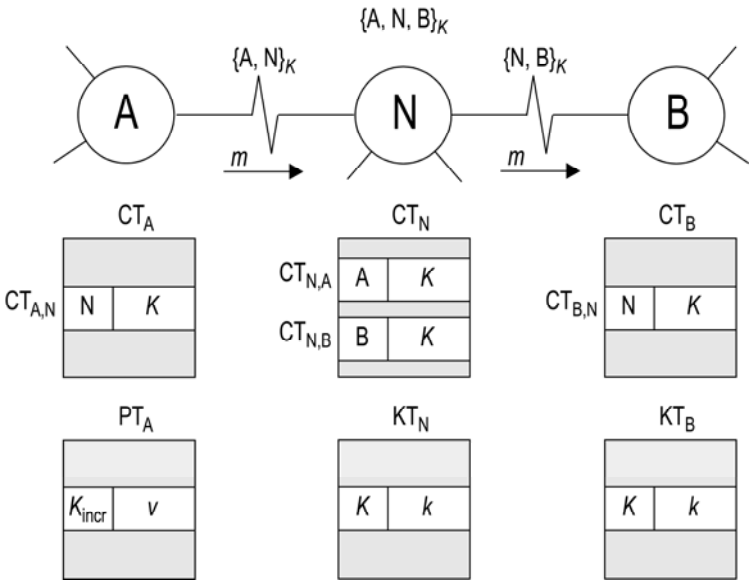

Figure 3. Sending a message from node A to node B through node N.

On termination, key *K* has been propagated to node B and connection $\{A, B\}_K$ has been set up.

Let us now consider the case that node A sends a message *m* to node B through a third node N. We shall suppose that connection $\{A, N\}_K$ has been already set up. The delivery of message *m* proceeds as follows (see Figure 3):

1. Node A enciphers message *m* by using key *K* and sends this message to node N by means of connection $\{A, N\}_K$;
2. If there exists connection $\{N, B\}_K$ between nodes N and B, node N forwards message *m* to B; otherwise
3. If there exists connection $\{N, B\}_{K'}$ where $K' \neq K$, node N uses this connection to send a message to B encrypted by using *K*' and containing pair <*K*, *k*> so as to set up connection $\{N, B\}_K$; otherwise
4. A search is made for a global key *G* shared by nodes N and B[1] (if N and B share no global key, node N performs a pairing attempt with a different adjacent node and communicates with B through this node). Then, node N uses key *G* to send pair <*K*, *k*> to B so as to set up connection $\{N, B\}_K$.
5. Node N uses connection $\{N, B\}_K$ to forward message *m* to B.

On termination, key *K* has been propagated to node B and communication channel $\{A, N, B\}_K$ has been set up.

The actions delineated above can be extended to an arbitrary sequence of nodes A, N_0,

[1] It is important to note that global key *G* does not need to be shared by node A.

N_1,..., N_r, B. On termination, key *K* has been propagated from node A to node B as well as to all intermediate nodes, and communication channel $\{A, N_0, N_1,..., N_r, B\}_K$ has been set up.

We may conclude that the delivery of a message encrypted by using key *K* from node A to node B through an arbitrary number of nodes causes the propagation of *K* to all the intermediary nodes and generates a communication channel involving these nodes. A salient property of a communication channel using key *K* and generated by propagation is that the principal node P(*K*) of *K* is always the first node of the channel. This means that, given the name *K* of the cryptographic key of a given communication channel, it is always possible to identify the first node of this channel by using the K_{node} component of key name *K*. We shall take advantage of this property in key replacement.

2.4. Key replacement

A *local key replacement* is the action of assigning a new value to an existing local key. The new key value will have to be inserted into the local key table of every node holding this key. Of course, a result of this type can be obtained by broadcasting a local key replacement message to all the network nodes. In an alternative approach, we avoid the high costs in terms of network traffic that are connected with message broadcasting by taking advantage of the mechanism for key propagation. In this approach, we only generate the messages necessary to reach those nodes that are part of a communication channel relying on the key being replaced.

Let *K* be the key to be replaced, let *v* be the master value of this key, and let *k* be the corresponding key value. Furthermore, let *v** be the new master value of *K*, and let *k** be the corresponding new key value. We exploit the fact that the principal node of key *K*, say node A, is identified by the K_{node} component of quantity *K* (i.e. K_{node} = A), and all communication channels relying on *K* start from A. The actions involved in key replacement are as follows (see Figure 4):

1. Node A performs a search in its own connection table CT_A to find all the entries that contain key name *K*. For each of these entries, let N be the corresponding node.
2. Node A performs a search in entry $CT_{A,N}$ of connection table CT_A for a local key $K' \neq K$ (if no such local key exists, a global key is used. If A and N share no global key, A performs a pairing attempt with a different adjacent node and communicates with N through this node).
3. Node A assembles a key replacement message *m* that includes key name *K*, the *old* master value *v* of this key and the *new* key value *k**. Message *m* is sent to N.
4. Node N authenticates message *m* as follows. Quantity *v* is extracted from the message

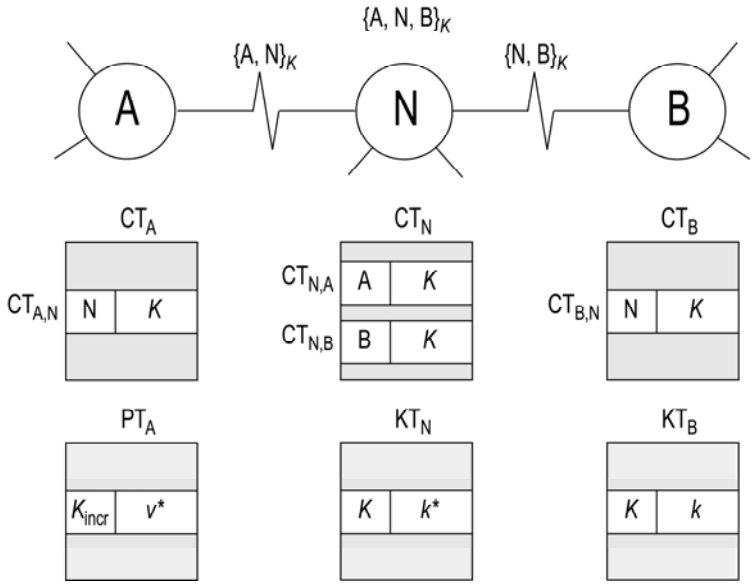

Figure 4. Local key replacement. The figure shows a situation in which the new value k^* of key K has been propagated in channel $\{A, N, B\}_K$ from node A to the intermediate node N, and this node has not yet propagated the new value to node B.

and is used to evaluate quantity $f(v)$; the result is compared with the value k of key K, as contained in local key table KT_N. If no match is found, the authentication fails and m is discarded. If the match is successful, the new key value k^* of key K is extracted from m and is inserted in the entry corresponding to key K in local key table KT_N.

It should be clear that in the intermediate propagation phases, when the new key value has been distributed to only a subset of all the nodes involved in the key replacement, messages may well be sent using the old key. This will be the case, for instance, for a reply message sent in a channel relying on key K in the reverse direction, from the end node to the start node, as long as the propagation of the new key value has not yet reached the end node. Detection of situations of this type occurs as follows. Each given message m includes the name K of the key that was used to encrypt that message both in plaintext and in the ciphertext form obtained by using the same key, K. When the generic node M receives message m, it deciphers the key name contained in ciphertext in m by using the key value contained in its own local key table KT_N. If the result does not match the key name contained in plaintext in m, then either KT_N contains the outdated value of K, or message m was encrypted by using the outdated value of K. In both cases, message m should be discarded.

3. DISCUSSION

A key is compromised if it has been acquired by an illegitimate agent. This can be the result of application of well-known techniques for data stealing [18], [19], for instance. When

a global key is compromised, it should be eliminated from all the network nodes. This result can be obtained by broadcasting a key invalidation message. If more global keys have been compromised, the global key invalidation message will include a list of the names of these keys [20]. On receipt of a message of this type, each node will simply delete the compromised global keys. So doing, these keys will no longer be used to establish connections between nodes.

When a local key has been compromised, it should be replaced. In our configuration of the node tables, replacing a given key means to insert the new key value into the local key table of every node holding this key. Of course, a result of this type can be obtained by broadcasting a local key replacement message to all network nodes. In an alternative approach, we take advantage of the key replacement mechanism, illustrated in Subsection 2.4. The ensuing reduction of the number of messages that are required for key replacement is especially important as far as energy consumption is concerned [21].

3.1. Group communication

In the *group communication model*, sensor nodes that cooperate for the same task are logically placed in the same group [13], [22]. Several groups may well co-exist within the given sensor network. Interactions between the nodes that are members of the same group are frequent, whereas interactions between the members of distinct groups are comparatively rare. Groups can be organized hierarchically; in this case, interactions between different groups only involve the nodes at the highest hierarchy levels. Group organization is logical rather than physical. In the presence of different applications, the composition of the different groups may well overlap according to arbitrary topologies; nodes are grouped on the basis of application-specific requirements, and different applications may take advantage of the same node.

Our key propagation mechanism is well suited to support the group communication model. In each group, a single key, the *group key*, will be shared by all group members. A node is chosen to be the *group controller* for the purpose of key distribution within the group. To this aim, the group controller (e.g. a base station) creates one or more communication channels relying on the group key, so as to propagate the group key to all group members. Each member can communicate with the other nodes in the group by using this key. Multiple overlapping group topologies are supported by the possibility of associating several local keys with the same given node.

Group membership may well change. When a new node is added to an existing group, the

key of the group must be assigned to this node. This result can be simply obtained by sending a message to the new node using this key. The message can be sent by the group controller as well as by any other group member. A common requirement is that when a new node enters a group, the group key is replaced to prevent the new node from using the old group key to read messages exchanged before it joined the group if it has recorded these messages (*backward secrecy*) [13]. A result of this type will be obtained by taking advantage of our key replacement mechanism; the new node will be included in a key replacement message. An important property is that, if a key is changed while a node is off-line, this node will be in the position to recover after coming back on-line. This will happen the first time that a message is sent to that node using the new group key.

When a node leaves a group, the replacement of the group key is mandatory to prevent that node from taking advantage of communication with the group members any longer (*forward secrecy*) [13]. Of course, the new key will be sent to all group members except the leaving node, and the new key cannot be sent using the old key, as the leaving member knows the old key. Once again, our key replacement mechanism is a solution to this problem. The new key will be sent using alternative connections relying on other local keys, or, if no such connection exists, on the global keys. A further application is a *periodic rekeying*, i.e. the action of replacing the value of a local key regardless of changes in group topology [23], [24]. An action of this type may be carried out at regular intervals to safeguard key secrecy, and to maintain resilience to attacks and failures, for instance [25], [26].

3.2. Storage requirements

As anticipated in the introduction, as a consequence of the stringent limitations existing in sensor nodes on memory space, the cost of key storage is a significant parameter. Let us refer to a large-scale network consisting of up to 2^{24} nodes, i.e. the size of the K_{node} component of a key name is 24 bits (see Subsection 2.1). If we reserve 8 bits for the K_{incr} component, then up to 256 keys can be generated in each node. In this hypothesis, the size of a key name is 4 bytes. For 64-bit key values, in the given node M the size of an entry of the principal table PT_M is 9 bytes, and the size of an entry of the local key table KT_M is 12 bytes. These memory costs correspond to a communication channel that departs from or includes M, respectively.

We hypothesized that the name of a global key is equal to its order number in the key pool $\mathcal{P}$ originally generated in the key pre-distribution phase. In this hypothesis, the size of a global key name descends from the cardinality of $\mathcal{P}$. A size of 16 bits will accommodate a

large $\mathcal{P}$. It follows that the memory requirement of a global key is 10 bytes (2 bytes for the key name and 8 bytes for the key value). As seen in the introduction, the total number *p* of global keys in each node is a parameter of the probabilistic key management scheme, and is a function of the desired degree of connectivity. We may well argue that the number of global keys in each node is significantly greater than the number of communication channels that depart from or include that node. We may conclude that the memory requirements for storage of the local keys are a negligible fraction of the total requirements for key storage.

4. CONCLUDING REMARKS

We have considered an important security problem of wireless sensor networks, the distribution of the cryptographic keys. We have presented a solution that supports the creation of communication channels connecting sequences of adjacent nodes. A salient feature of our approach is that all the nodes in a channel share the same key. This result has been obtained by propagating the key connecting the first two nodes to all the other nodes in the channel. We have obtained the following results:

- A message transmitted through the communication channel is transformed to ciphertext only once, by the start node. All the subsequent channel nodes possess the cryptographic key; they will use this key to read the control part of the message and then forward the whole message to the next node. So doing, we obtain a significant reduction of the number of message transformations from plaintext to ciphertext, which is an important factor as far as energy consumption is concerned.
- Communication channels are bidirectional. If a communication channel has been set up in the delivery of a message from node A to node B, then the same channel can be used in the reverse direction to deliver a reply message from node B to node A. A channel can be used for message transmission between any two channel nodes. In this way, we keep the number of actions of key distribution to a minimum.
- Local key replacement takes advantage of the mechanism for key propagation. We avoid the high costs in terms of network traffic that are connected with message broadcasting. Instead, we generate only those messages that are necessary to reach the nodes that are part of the communication channels relying on the key being replaced.
- Key propagation gives effective support to group communication. We can take advantage of the possibility of associating several local keys with the same given node to implement multiple groups with overlapping topologies. The key replacement mechanism is well

suited to support forms of backward and forward secrecy, replacing the group key when a new node enters an existing group or a node leaves a group.

ACKNOWLEDGMENTS

The authors wish to thank the anonymous reviewers for their constructive suggestions and insightful comments.

This work has been partially supported by the EU FP7 Integrated Project PLANET (Grant Agreement no. FP7-257649), and by the TENACE PRIN Project (Grant no. 20103P34XC_008) funded by the Italian Ministry of Education, University and Research.

REFERENCES

[1] L. H. A. Correia, D. F. Macedo, A. L. dos Santos, A. A. F. Loureiro, J. M. S. Nogueira, "Transmission power control techniques for wireless sensor networks," *Computer Networks*, vol. 51, no. 17 (December 2007), pp. 4765–4779.

[2] S. Lin, J. Zhang, G. Zhou, L. Gu, J. A. Stankovic, T. He, "ATPC: adaptive transmission power control for wireless sensor networks," *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, Boulder, CO, USA, November 2006, pp. 223–236.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8 (August 2002), pp. 102–114.

[4] A. Perrig, J. Stankovic, D. Wagner, "Security in wireless sensor networks," *Communications of the ACM*, vol. 47, no. 6 (June 2004), pp. 53–57.

[5] L. Lopriore, "Hardware/compiler memory protection in sensor nodes," *International Journal of Communications, Network and System Sciences*, vol. 1, no. 3 (August 2008), pp. 207–283.

[6] J. Zhang, V. Varadharajan, "Wireless sensor network key management survey and taxonomy," *Journal of Network and Computer Applications*, vol. 33, no. 2 (March 2010), pp. 63–75.

[7] Y. Zhou, Y. Fang, Y. Zhang, "Securing wireless sensor networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 3 (Third Quarter 2008), pp. 6–28.

[8] M. A. Simplício, P. S. L. M. Barreto, C. B. Margi, T. C. M. B. Carvalho, "A survey on key management mechanisms for distributed wireless sensor networks," *Computer Networks*, vol. 54, no. 15 (October 2010), pp. 2591–2612.

[9] K. Ren, *Communication Security in Wireless Sensor Networks*. PhD dissertation, Worcester Polytechnic Institute, 2007. Available at: *http://www.wpi.edu/Pubs/ETD/Available/etd-040607-174308/unrestricted/Kui_Ren.pdf*

[10] L. Eschenauer, V. D. Gligor, "A key-management scheme for distributed sensor networks," *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Washington, DC, USA, November 2002, pp. 41–47.

[11] S. Zhu, S. Setia, S. Jajodia, “LEAP+: efficient security mechanisms for large-scale distributed sensor networks,” *ACM Transactions on Sensor Networks*, vol. 2, no. 4 (November 2006), pp. 500–528.

[12] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, D. E. Culler, “SPINS: security protocols for sensor networks,” *Wireless Networks*, vol. 8, no. 5 (September 2002), pp. 521–534.

[13] G. Dini, I. M. Savino, “LARK: a lightweight authenticated rekeying scheme for clustered wireless sensor networks,” *ACM Transactions on Embedded Computing Systems*, vol. 10, no. 4 (November 2011).

[14] R. Di Pietro, L. Mancini, Y.W. Law, S. Etalle, P. Havinga, “LKHW: a directed diffusion-based secure multicast scheme for wireless sensor networks,” *Proceedings of the International Conference on the Parallel Processing*, Pittsburgh, PA, USA, October 2003, pp. 397–406.

[15] C. Intanagonwiwat, R. Govindan, D. Estrin, “Directed diffusion: a scalable and robust communication paradigm for sensor networks,” *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, Boston, Massachussets, USA, August 2000, pp. 56–67.

[16] M. Shehab, E. Bertino, A. Ghafoor, “Efficient hierarchical key generation and key diffusion for sensor networks,” *Proceedings of the Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, Santa Clara, CA, USA, September 2005, pp. 76–84.

[17] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, October 1996.

[18] N. Tuck, B. Calder, G. Varghese, “Hardware and binary modification support for code pointer protection from buffer overflow,” *Proceedings of the 37th International Symposium on Microarchitecture*, Portland, Oregon, USA, December 2004, pp. 209–220.

[19] Y. Younan, F. Piessens, W. Joosen, “Protecting global and static variables from buffer overflow attacks,” *Proceedings of the Fourth International Conference on Availability, Reliability and Security*, Fukuoka, Japan, March 2009, pp. 798–803.

[20] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, M. Galloway, “A survey of key management schemes in wireless sensor networks,” *Computer Communications*, vol. 30, no. 11–12 (September 2007), pp. 2314–2341.

[21] Y. Liang, W. Peng, “Minimizing energy consumptions in wireless sensor networks via two-modal transmission,” *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1 (January 2010), pp. 12–18.

[22] O. Cheikhrouhou, A. Koubâa, G. Dini, M. Abid, “RiSeG: a ring based secure group communication protocol for resource-constrained wireless sensor networks,” *Personal and Ubiquitous Computing*, vol. 15, no. 8 (December 2011), pp. 783–797.

[23] F. Kausar, S. Hussain, J. H. Park, A. Masood, “Secure group communication with self-healing and rekeying in wireless sensor networks,” *Proceedings of the 3rd International Conference on Mobile Ad-Hoc and Sensor Networks*, Beijing, China, December 2007, pp. 737–748.

[24] T. Pham, P. A. Watters, “The efficiency of periodic rekeying in dynamic group key management,” *Proceedings of the Fourth European Conference on Universal Multiservice Networks*, Toulouse, France, February 2007, pp. 425–432.

[25] M. Eltoweissy, M. Moharrum, R. Mukkamala, “Dynamic key management in sensor networks,” *IEEE Communications Magazine*, vol. 44, no. 4 (April 2006), pp. 122–130.

[26] S. Rafaeli, D. Hutchison, “A survey of key management for secure group communication,” *ACM Computing Surveys*, vol. 35, no. 3 (September 2003), pp. 309–329.