34

SECURE SELECTIVE EXCLUSION IN AD HOC WIRELESS NETWORK

Roberto Di Pietro, Luigi V. Mancini
Dipartimento di Scienze dell'Informazione, Università di Roma "La Sapienza", Via Salaria 113, 00198 Roma - Italy, {dipietro,mancini}@dsi.uniroma1.it

Sushil Jajodia
Center for Secure Information System, George Mason University, Fairfax - VA 22030-4444 - USA, Jajodia@gmu.edu

Abstract: A wireless sensor network can be seen as a large number (hundreds of thousand) of small (a few cubic millimetres) devices, battery powered, with very limited hardware resources. Such a network has been studied specifically in the ad hoc model, where the sensors autonomously set up a network infrastructure. We propose here an extension to the current wireless ad hoc sensor network (WSN) model (in particular the base station model), by introducing a Supervisor which has very few interactions with the network, it is mobile in itself, it could have more powerful hardware and it is asynchronous with respect to the sensors. Nevertheless, the Supervisor has to interact with the sensor network, for example to invoke the command to exclude from the network a selected sensor. We believe such a model is particularly suitable for, but not limited to, military applications. We then propose a distributed, cooperative, parallel algorithm for this model that assures the following properties: it enforces both the secure exclusion of a selected compromised sensor from the network and the rekeying of the remaining sensors. It has an overall low overhead both in terms of computation and required transmitted messages. It is scalable, since the algorithm requires only limited, local knowledge of the network topology. Finally, it can be adopted, as an independent layer, to enforce secure exclusion in other models.

Key words: secure selective exclusion, wireless sensor network security, distributed algorithm, symmetric key management.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35586-3_46

M. A. Ghonaimy et al. (eds.), *Security in the Information Society*

1. INTRODUCTION

Two basic system models have been proposed for the wireless network paradigm [14]. The *fixed backbone* wireless system model assumes two distinct sets of entities in a wireless network: a large number of mobile nodes and relatively fewer, but more powerful, fixed nodes. The static network is composed of all fixed nodes and the communication paths between them. The communication between a fixed node and a mobile one within its range occurs via wireless medium. The *ad hoc* wireless system model assumes that mobile hosts can form networks without relying on a fixed infrastructure. The structure of *ad hoc* wireless network is highly dynamic. Recent extensions of the *ad hoc* model deal with large collections of very small communication devices (a few cubic millimetre sensors) networked in an *ad hoc* fashion, which should be deployed in the near future both in military and commercial scenarios [7]. For instance, they could be used to collect data from a field in order to reveal the presence of a toxic gas, to facilitate rescue operations in snow in a wide open mountainous area, to fulfil perimeter surveillance duties, or to measure concentration of metal (e.g., nodules of manganese) on the ocean bed.

1.1 Security Challenges

The *ad hoc* model is naturally attack prone, because of both the hardware limitation and the communication media [8]. The hardware limitation poses tight constraints on both communication and computing power. On communication because such an activity increases the battery consumption, on computing power because sensors are limited both in memory (static and volatile) and in CPU frequency.

For example, as a consequence, asymmetric cryptography cannot be used. Moreover, the wireless communication media, natively spreads information around, so that eavesdropping is even easier than in wired networks [3]. Finally, it not is feasible to build the sensors in a tamperproof shelter for all classes of applications since (1) due to the large number of sensors, the total cost could be too high, (2) the implementation of secure micro-controllers to enforce tamper resistancy has impact on the power consumption and electromagnetic interference, and on the size of the sensors.

Even under the tamperproof assumption, selective exclusion of a network component is a desirable feature, both for cryptographic considerations (key can be recovered by cryptanalysis) and for robustness (a sensor failure can be excluded). This is the reason why we have to take into account, as in [10],

that sensors can be compromised (i.e. an attacker managed to get all the information the sensor contains), despite other security mechanisms.

1.2 Contributions and Roadmap

The main contributions of this paper include: (1) extending the model of a WSN, introducing a Supervisor, which can invoke the exclusion of a sensor from the network as well as other functions (see section 3); (2) providing a scalable, cooperative, distributed and parallel algorithm that allows the selective and secure exclusion of a compromised sensor from the network; (3) rekeying through the same algorithm used to enforce selective exclusion.

Section 2 contains the details of our assumptions, while section 3 introduces the extensions to the WSN model. In section 4 we describe in detail the exclusion of a compromised sensor, introducing notation, scenario, objective, model, architecture and mechanisms devised to achieve the objective. Finally, in section 5, concluding remarks are provided.

2. SYSTEM ASSUMPTIONS

Since the number of the sensors can be in the order of hundreds of thousand or even more, it is desirable not to have a single point of centralization, both for efficiency (it will inevitably be a bottleneck) and availability (single point of failure) reasons. Since there is no knowledge of the network topology, the different sensors must cooperate in order to gather any other information not related to their own state. Moreover, such a cooperation must be limited because of the limited resources of the sensors and not interfere with the data traffic. As a whole, these objectives can be pursued through: (1) the cooperation of as few sensors as possible, in order to scale; (2) the application of the locality principle, that is exploiting only sensors close to the enquiring sensor, in order to minimize the latency and the traffic overhead [4].

2.1 Sensor Network Components

In the following we will assume that: (1) all sensors are *built equal*, that is, they have the same internal components, both hardware and software, and are **not** tamper-proof; (2) each sensor has its own ID; (3) a sort of topology has been superimposed on the wireless *ad hoc* sensor network, as explained in the next subsection. In particular, we assume that the hardware of a sensor provides enough resources to allow the storing of the ID of the sensor and

the storing of the information required to participate in the process of authenticating the Supervisor or the sensor that acts on its behalf (called the *hseed* register). Moreover, each sensor can store a one-way function [13] H, used in the authentication process, and the ID of each of the sensors in its communication range. The initial value of *hseed* is $H^n(S_0)$ for some n and S_0. Our assumptions are coherent with those in [10].

2.2 Underlying level of Architecture

We assume that: (1) the routing layer has already been implemented [14] and an appropriate level of security has been provided to secure it [9]; (2) there is a two-level hierarchical network architecture. In particular, at the upper level, we have Clusterheads (CH*s*) while, at lower level, we have cluster members. The set up of this infrastructure can be achieved in different ways, as can be seen in [12], [1], [2].

The role of a CH, to the extent we are interested in, is to act as a key distributor. The role of the members of the cluster is to gather data from the field (e.g.: temperature, humidity, wind speed and direction, presence of toxic gas, lights conditions and so on) encrypt that data and send it to a specific device (a sensor or a Supervisor). We assume the CH is able to manage the list of its member (i.e. to store their ID) and that each sensor put its ID in any communication where it acts as a sender. The role of the Supervisor is to invoke, asynchronously with respect to the WSN, the order to exclude a specific component from the network. Other functionalities will be discussed in section 3; however, we will omit the details due to lack of space.

3. EXTENSIONS TO THE MODEL

Our model proposes a fusion of both *ad hoc* and *fixed backbone* models. The idea is to get the advantages of both, while avoiding the related disadvantages. In particular, in the model, we propose: (1) to leave the set up and the normal functioning of the WSN to the WSN itself; (2) to put the burden of calculating the new key on the Supervisor; (3) the Supervisor can be mobile; (4) the Supervisor can be assumed tamper proof and may have an higher availability of resources (e.g., computational power, energy power, level of security); (5) the Supervisor can invoke asynchronously, with respect to the network, the execution of commands; (6) the Supervisor can start the rekeying, in an event or time triggered fashion, to reduce the overhead.

In particular, the functionalities the Supervisor should perform include the management of: (1) *poll*: a poll occurs when a sensor no longer belongs to the network. The Supervisor can be involved in the leave when a significant portion of the network is unavailable. It should be fed with such information by the CHs (see subsection 2.2). Information about leave is quite important, in order to suggest upper levels to renovate the sensors in a particular area. Such an operation, if carried out, will enable a join (see next functionality); (2) *join*: the WSN is depleted in the number of the sensors (e.g. due to destruction, failure, batteries exhaustion or the *one step domino effect* -see subsection 4.3). With join, a WSN is re-populated. The Supervisor can play a key role in such an operation, for instance by providing to the operating WSN a token through which the new sensors could be successful authenticated;(3) *merge*: with such an operation the Supervisor performs the secure merge of two or more different *WSNs*. For instance, the merge of the already managed WSN with the one acquired through the *split* performed by the Supervisor of another WSN; (4) *split*: that is, to partition the current WSN and delegate the management of a single partition to another Supervisor. The receiving Supervisor will acquire and integrate this partition to the WSN it already manages through a merge; (5) *selective exclusion*: we want a compromised sensor (in the following referred to as *red* sensor), whose ID is known, to be excluded from the network (i.e. not to be able sending or receiving messages). As an example, a Supervisor could be an enginerized workstation carried by a soldier in the battlespace where the WSN has been deployed or a drone flying over that battlespace. In the following, we will focus on the analysis and implementation of the selective exclusion.

4. EXCLUDING A COMPROMISED SENSOR

In this section, we develop an algorithm that enforces secure exclusion. It is based on the use of a single key, GDK, to encrypt both control and data traffic.

4.1 Notation

We use for CH the variable *ClusterMember* to store the list of its members, and the variable *Neig_List* to store the IDs of the sensors in its communication range.
The primitives RECEIVE and SEND are assumed available. The argument of the RECEIVE is a message (msg), while the SEND has two arguments:

the sensor to which the message has to be delivered, and a message. Their invocation takes the form: RECEIVE (msg) and SEND (x, msg) respectively.

A message is composed of three fields, namely:

1. *type*: is a mandatory field. It codes one of the following information: (1) the message received is a request for the list of the neighbours of the receiving sensor (Neig_Req); (2) the message sent contains in the *content* field the list of the neighbours of the sending sensor (Neig_List); (3) all the sensors to which the authority token has been passed have updated their status (Reply); (4) the *content* field of this message contains the new key, as well as the authority token and the ID of the red sensor (Rekeying),
2. *content*: is an optional field. When present, it codes the following information: (1) the list of the neighbours (i.e. the list of the neighbours in the communication range of the cluster), referred to as *Neig_IDs*; (2) the new key, together with the authority token and the ID of the red sensor; referred to as E_{GDK}(GDK' || *hseed* || red),
3. *ID*: it is a mandatory field. As stated in 3.1, it is the Identifier of the sensor.

The symbol || means the concatenation of the portions before and after the symbol.

In terms of encryption, writing $E_{GDK}(msg)$ states that the message has been encrypted with the algorithm E, using the key $_{GDK}$. Writing m= $E^{-1}{}_{GDK}$(msg) signifies that the message has been decrypted employing the algorithm E with key $_{GDK}$. Each single field of the decrypted message *msg* has been assigned to the corresponding field of the variable *m* of type message.

A single field in a variable *m* of type message can be accessed by writing: m.type or m.ID. It is possible to access a single value of the *content* field, for instance the *hseed* field, by writing: m.content.hseed.

4.2 Scenario and Objective

We assume that the Supervisor of the WSN becomes aware that a specific sensor of the network, identified by its ID, is untrusted. Then the Supervisor invokes the command to exclude the malicious sensor from the network. In the following, we will analyze the measures we have to take in order to securely exclude the malicious sensor from the network.

In the depicted scenario, if in a cluster only one sensor is malicious, the new GDK cannot be communicated to the members of the cluster to which the malicious sensor belongs. In fact, having only the GDK, it can occur that: (1) when the CH receives the new key (GDK'), the malicious sensor could be in the communication range of the sender CH, and therefore could

eavesdrop the message that carries the new key. The compromised sensor could decrypt the message recovering the new GDK'; (2) if the compromised sensor is not in the direct communication range of the CH when it receives the new key, it could eavesdrop the new key when the CH communicates the new key to the sensors in its cluster (to which the compromised sensor belongs).

Therefore, we have to exclude from the communication of the new key GDK' all the members of the cluster, as well as the CH. Moreover, to pursue the objective of avoiding the lack of the new key to the malicious sensor, we have to extend the scope of the exclusion beyond the single cluster. To do that, we need a model, illustrated in the next subsection.

4.3 Model

In the following, we will refer to a cluster to which a malicious sensor belongs as *red*. Let us focus on the *neighbours* of the red cluster. Cluster A is defined to be a neighbour of cluster B, if at least one of the components of cluster A is in the communication range of at least a member of cluster B. The *neighbourhood* of a cluster is given by the collection of each neighbour. The yellow label is assigned to the neighbours of the red cluster, while the green label is assigned to the clusters that do not have a red cluster in their neighbourhood.

If the new key is communicated to a yellow cluster, there is the risk to disclose the new key to the malicious sensor in the red cluster. This could happen if the communication range of the malicious sensor overlaps with the communication range of at least a sensor in the yellow cluster.

To prevent this lack, conservative considerations have to be applied, and therefore we have to exclude from the rekeying also the neighbours of the red cluster. We call the side effect of the exclusion of the yellow cluster to avoid the leakage of the new key as: *one step domino effect*. We will not address how to recover or to limit the one step domino effect here, because it is beyond our scope. As previously stated, we focus on reaching a secure exclusion of the red sensor.

4.4 Architecture

Following the guidelines, we have to depict an architecture that could enable the property of being distributed and cooperative.

1. No global or even local knowledge of the network topology is available when the Supervisor wants to invoke a command. This implies the Supervisor has to set up a discovery phase.

2. Another security concern is related to authentication, that is, how the Supervisor can be authenticated with respect to a CH. Let's refer to the credential it must show as *authority token.*
3. To enforce a distributed solution, the authority token must be passed among the CHs.
4. Moreover, we can take advantage of the independence of the *CHs* to perform rekeying in a parallel distributed fashion, assuming the authority token can be replicated and passed to more than one CH at time.
5. To assure the Supervisor that the rekeying has succeeded, the Supervisor has to be acknowledged before it could invoke a new order. Such an architectural choice avoids layering of different orders, leading the network to an inconsistent state (i.e. different portions of the network working with different keys).

4.5 Mechanisms

Here we introduce a distributed algorithm that, without *prior* knowledge of the topology of the WSN achieves the secure exclusion of the compromised sensor from the WSN. The new key is not leaked to the compromised sensor since the algorithm assures the exclusion of both the yellow and the red clusters from rekeying. The Supervisor has only to start the process. Once an initial green CH has been rekeyed by the Supervisor, this CH holds the authority token of the Supervisor itself. Therefore the green CH can propagate the rekeying, authenticating itself towards other *CHs* acting on behalf of the Supervisor. In this way, the rekeying takes place in a distributed, cooperative and even parallel fashion.

4.5.1 Algorithm

What is worth noting is that the supervisor need to be authenticated by the CH, and this is achieved by a one time password [15] like mechanism. In detail, based on the assumption in subsection 3.1, when a sender claims to be the Supervisor (or to act on its behalf), it will send $H^{n-1}(S_0)$. The receiver will apply $H(H^{n-1}(S_0))$ and compare the result with *hseed.* If the two values match, the sender is authenticated. The receiver will store $H^{n-1}(S_0)$ in *hseed* for subsequent authentication.

Once the CH holds the authority token communicated by the Supervisor (i.e. $H^{n-1}(S_0)$), it can do act as a Supervisor. However, the related scope is limited only to the current phase. Once the secure selective exclusion has taken place, each CH loses its Supervisor role, since it does not hold the new authority token required to be authenticated as Supervisor (it does not hold

$H^{n-2}(S_0)$. To assure the Supervisor be acknowledged the rekeying took place, we resort to the diffusing computation solution [6].

The pseudo code of the Supervisor

```
- discovery phase -
%the Supervisor collects for a certain period of time, say Δ, the ID in its
%communication range; be List the data structure that hosts such a collection.
IF (Red-sensor IN (List)) THEN -the supervisor is in a red cluster: move to another
zone-
   ELSE BEGIN
      reply=0;
      FOREACH x IN List BEGIN
         % get the List of the neighbors of x
         Neig_IDs= getNeighbors(x);
         IF Red-sensor ISIN (Neig_IDs)  THEN -it is a yellow cluster: mark it
      yellow and do nothing -
               ELSE BEGIN - x is a green CH-
                  % Diffusing the rekeying
                  SEND(x, (E_GDK(Rekeying,(GDK' || hseed || red), ID));
                  reply = reply + 1;
               END
      END
   END
IF  reply == 0 THEN move; - our neighborhood was composed of only red and
yellow clusters -
   ELSE BEGIN
      - acknowledgment phase -
      WHILE NOT(reply==0) DO  BEGIN
         RECEIVE (msg);
         m=E^-1_GDK(msg);
         IF m.type==Reply  THEN reply=reply - 1;
            ELSE - an anomaly occurred -
      END
      hseed= H^(n-2)(S_0).  - authority token updating-
   END
```

The pseudo code of a cluster member

```
RECEIVE (msg)
- code added -
m=E^-1_GDK(msg);
IF m.type==Rekeying THEN
   IF H(m.content.hseed)==hseed THEN BEGIN -successful authentication-
      hseed=m.content.hseed; GDK=m.content.GDK'
   END
- normal execution -
```

The pseudo code of a CH

```
RECEIVE(msg);
m=E^-1_GDK(msg); - code added -
```

```
IF m.type == Neig_Req THEN SEND (m.ID,E_GDK(Neig_List||Neig_Ids|| ID));
IF m.type == Rekeying THEN BEGIN
   IF H(m.content.hseed)==hseed THEN BEGIN -successful authentication-
      authority=m.ID;  % remember the sender to which send the reply
      % exclude further authentication with the same old authority token
      hseed=m.content.hseed;  red=m.content.red; GDK'=m.content.GDK'
      % send to the member of the cluster the new key}
      FOREACH x IN ClusterMember SEND(x, (E_GDK(rekeying,(GDK'|| hseed||
       red),ID);
      -continue propagating the key-
      reply=0;
      FOREACH x IN ListNeig_IDs BEGIN
         Neig_IDs= getNeighbors(x) ;
         IF Red-sensor ISIN (Neig_IDs)  THEN - x is a yellow cluster-
            ELSE BEGIN -the CH is a green one-
               SEND(x, (E_GDK(Rekeying,(GDK'|| hseed|| red),ID));
               reply = reply + 1
            END  % else
      END %FOREACH
      IF reply == 0 THEN SEND(m.ID, E_GDK(Reply,ID)); - note: our neighborhood
       was composed of only red and yellow clusters-
      GDK=GDK'
      END % if -successful authentication-
   ELSE - authentication failed -  SEND(m.ID, E_GDK(Reply,ID))
END %if m.type...
IF m.type == Reply THEN BEGIN
   reply = reply - 1;
   IF  reply == 0 THEN  SEND(authority, E_GDK'(Reply,ID))
END
 -normal execution-
```

Notice that rekeying is just a choice to exclude the malicious sensor from the WSN. Another mechanism could have been to teach the yellow cluster not to route the messages from/to the red sensor. This could have been done as a simple extension of the algorithm, intervening in the code, where a cluster is marked as yellow. As an alternative, a decentralized approach could have been taken to start rekeying, that is embedding in each CH the appropriate authority token. But such a choice has a risk: if a CH becomes compromised, it could disrupt the network at will. We could resort to some voting or sharing algorithm in order to rend the solution much robust [11], but this would increase complexity and time in excluding a compromised sensor; moreover, we lose the advantage of having a point external to the sensors, that can take advantage of extra information, resources and control, leading to a quick effective, controlled reaction in case a sensor is compromised.

4.6 Discussion

Under the hypothesis that a compromised node continue misbehaving with its own identity, once discovered as shown, it can be excluded from the WSN. But another thing that a malicious node could do, is to present itself with another ID, possibly with the one of a sensor the attacker has deliberately destroyed. At this point, there is no way to differentiate the forged sensor from a genuine one. The main drawback with the GDK is that once the key is recovered, it does not exist another layer of security, i.e. when the *secret* is discovered no recovery is feasible.

What we would like to do is to introduce a multilevel security, that is, minimize the scope to which the sensor can interact with.

Moreover, under the assumption of lack of tamper proof components and in consideration of crypto analytical attacks, it emerges the necessity of robust techniques that could eventually incur in degradation of the WSN due to the exclusion of the red sensor, but let the rest of the WSN continue operating securely. Such techniques are currently under investigations and will be discussed in other papers.

5. CONCLUDING REMARKS

In the present work we extended the model of a WSN, devising a set of new functionalities that could be useful in a WSN. In particular, we dealt with the management of the secure selective exclusion of a sensor from the WSN, and we proposed a distributed, cooperative, parallel algorithm that, with little or no knowledge of the topology, achieve the goal of excluding a compromised sensor from the WSN. The proposed algorithm, other than being used as an independent layer to tackle the exclusion of a compromised sensor from a WSN, can also be particularly useful to deal with rekeying.

The solution proposed, due to its scalability and efficiency, can be seen as a building block to rely on resilient wireless *ad hoc* sensor network, even in presence of malicious nodes. It appears that if it is possible for the Supervisor to know the complete topology of the network, cleverer decisions could have been taken; for instance the network topology could be reconfigured, in order to minimize the domino effect or to preserve some property (e.g. to maintain the connectivity of the network).

A further goal is to find a solution that allow the Supervisor to know such a topology, even in presence of malicious sensors. For instance, if the physical position of a single sensor network inside a cluster were available, some action could be taken in order to recover sensor network in the red cluster not in the direct communication range of the malicious one.

Another interesting problem to investigate is to feed the corrupted sensor with misinformation, without having the corrupted cluster be aware of this (counter intelligence), in order to make the attacker take its decision on the basis of inconsistent information. Moreover, a challenging field of investigation is how to deal with an arbitrary behaviour of a compromised CH and, more generally, to what extents limit the countermeasures to the fourth principle of Denning[5].

REFERENCES

[1] A.D. Amis et al., "Max-min D-cluster formation in wireless *ad hoc* networks", INFOCOM, 2000

[2] S. Basagni, "Distributed clustering for *ad hoc* networks", Proceedings of the 1999 International Symposium on Parallel architecture, Algorithms, and Networks (I-SPAN'99), IEEE computer Society, 1999, pages 310-315

[3] N. Borisov, Ian Goldberg and D. Wagner, "Intercepting mobile communications: the insecurity of 802.11", the seventh annual international Conference on Mobile computing and networking, Rome, 2001, pages 180-189

[4] J. Broch et al., "A Performance Comparison of multi-hop wireless *ad hoc* network routing protocol", The fourth ACM/IEEE International conference on Mobile computing and networking, 1998, pages 85-97

[5] P.J. Denning "Fault Tolerant Operating Systems", ACM Computing survey, December 1976

[6] E.W. Dijkstra and C.S. Sholten, "Termination detection for diffusing computations", Inf. Process. Lett. Vol. 11, No. 1, 1980

[7] J.M. Kahn, R.H. Katz and K.J. Pister, "Mobile Networking for Smart Dust", ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom99), Seattle, 1999

[8] Vesa Kärpijoky, "Security in *ad hoc* Networks", Helsinki University of Technology, 2001, Internet draft

[9] S. Marti, K.L.M. Baker, "Mitigating Routing Misbehaviour in Mobile *Ad Hoc* Networks", Proceedings of the sixth annual international conference on Mobile computing and networking, 2000, pages 255-265

[10] A. Perrig et al., "SPINS: Security Protocols for Sensor Networks", the seventh annual international Conference on Mobile computing and networking, Rome, 2001, pages 189-199

[11] J. Pieprzyk et al., "Multiparty key agreement protocols", Computers and Digital Techniques, IEEE Proceedings, Vol. 147, issue 4, July 2000, pages 229-296

[12] L. Ramachandran et al., "Clustering Algorithms for Wireless *Ad Hoc* Networks", Proceedings of the fourth international workshop on Discrete algorithm and methods for mobile computing and communication, 2000, pages 54-63

[13] R.L. Rivest, "The MD5 message-digest algorithm", Internet Request for Comments, April 1992, RFC 1321

[14] E.M. Royer, C-K Toh, "A Review of Current Routing Protocols for *Ad Hoc* Mobile Wireless Network", IEEE Personal Communication Magazine, April 1999, pages 46-55.

[15] N.M. Haller, "The S/KEY one-time password system", In ISOC, 1994