Secure and Reliable Clustering in Wireless Sensor Networks: A Critical Survey

Péter Schaffer[a,1,*], Károly Farkas[b,c], Ádám Horváth[c], Tamás Holczer[b], Levente Buttyán[b]

[a] *Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg*
[b] *Department of Telecommunications, Budapest University of Technology and Economics, Hungary*
[c] *Institute of Informatics and Economics, University of West Hungary, Hungary*

Abstract

In the past few years, research interest has been increased towards wireless sensor networks (WSNs) and their application in both the military and civil domains. To support scalability in WSNs and increase network lifetime, nodes are often grouped into disjoint clusters. However, secure and reliable clustering, which is critical in WSNs deployed in hostile environments, has gained modest attention so far or has been limited only to fault tolerance. In this paper, we review the state-of-the-art of clustering protocols in WSNs with special emphasis on security and reliability issues. First, we define a taxonomy of security and reliability for cluster head election and clustering in WSNs. Then, we describe and analyze the most relevant secure and reliable clustering protocols. Finally, we propose countermeasures against typical attacks and show how they improve the discussed protocols.

Keywords: Secure and reliable clustering, dependable clustering, cluster head election, attacker, sensor networks, taxonomy

1. Introduction

With the advances in sensor development technologies, wireless sensor networks (WSNs) are growing popular for a wide range of applications in the military and civil domains. A network of sensors can be employed among others for environmental monitoring, military or civil surveillance, target tracking, territory/premises protection or disaster forecast. WSNs usually comprise a mass

*Corresponding author.
Email addresses: peter.schaffer@uni.lu (Péter Schaffer), farkask@hit.bme.hu (Károly Farkas), horvath@inyf.nyme.hu (Ádám Horváth), holczer@crysys.hu (Tamás Holczer), buttyan@crysys.hu (Levente Buttyán)

[1] This work was partially done while Péter Schaffer was with Budapest University of Technology and Economics.

of small, energy-constrained devices with limited processing and wireless communication capabilities. These devices integrate sensors and actuators and can operate in an unattended manner even in hostile environments.

To support scalability and increase the lifetime of WSNs, sensor nodes are often grouped into clusters. These clusters are mostly disjoint and consist of a leader, referred to as the cluster head, and member nodes. The cluster heads are usually elected in the network through local or global mechanisms. For instance, nodes with the highest degree in their neighborhood or closest location to given reference points in the network can become cluster head. On the other hand, member association is done via some association procedure, e.g., selecting the cluster head with the shortest communication distance. The cluster head controls the operation of its cluster in the way that it periodically collects sensor readings from the associated member nodes and forwards the gathered information to a (central) sink station often after local processing (e.g., aggregation or filtering). Such cluster head election protocols are, for example, [1, 2, 3, 4, 5, 6].

All the same, security and reliability aspects of clustering and cluster head election have gained modest attention so far. Security and reliability (or alternatively, dependability) in this case means resistance against attacks.[2] Such attacks can have different goals depending on the resources and knowledge available to the adversary. For instance, a primary goal can be to reduce the efficiency of data collection performed by the sensor network or to completely disrupt its operation. This can be achieved either by preventing the termination of the algorithm, or by confusing the nodes with regard to their responsibilities in the network. Alternatively, an adversary may aim to predict the structure of the network, i.e., foretell nodes that will be cluster heads and nodes that will be associated with a given cluster head. This can be done, for example, by overhearing packets sent by the nodes during the election process, or by extracting information from publicly available parameters. The acquired information can be applied to launch targeted attacks against cluster heads having a larger impact than attacks against randomly chosen nodes. Clearly, as sensor networks are often supposed to operate in hostile environments, one cannot neglect the viewpoint of dependability in cluster head election protocol design.

On the one hand, there are many papers in the literature that survey the security solutions applied in wireless sensor networks, e.g. [7, 8, 9, 10, 11, 12, 13, 14]. These papers detail the common security issues in sensor networks, like authentication, intrusion detection, secure routing, secure data aggregation, etc. However, none of these papers address the issue of secure and reliable cluster head election in particular. On the other hand, some survey papers, e.g. [15, 16, 17, 18], tackle the problem of clustering and cluster head election in sensor networks focusing on issues like power–quality trade-off, complexity, cluster stability, cluster head election criteria, and so on. Regrettably, the latter papers do not consider the security aspects of clustering. Our paper is intended to fill this gap.

[2] In this paper, we use *security and reliability* and *dependability* interchangeably.

In this paper, we give a critical overview of the state-of-the-art on secure and reliable clustering protocols and their cluster head election mechanisms in homogeneous wireless sensor networks. First, we define a taxonomy of dependable cluster head election and clustering. We identify the key properties of dependability, such as *termination*, *completeness*, *consistency*, *role non-manipulability*, *association non-manipulability*, *role unpredictability*, *association unpredictability*, *role unidentifiability* and *association unidentifiability*, and give their definitions. We also propose a property investigation sequence that we follow in our investigations. Moreover, we define three adversarial models, such as *passive*, *active* and *compromising*, that cover the possible failure and attack scenarios. Then, we categorize and briefly describe the most relevant dependable clustering protocols and systematically investigate the fulfillment of the defined dependability properties in case of the different adversarial models. The taxonomy, the resulting categorization of protocols, and the analysis methodology are original contributions. The rest of the paper is a survey carried out from a novel, dependability-oriented viewpoint. We will see that the majority of the investigated protocols show weak dependability and assuming compromising adversaries, usually none of the properties except termination and completeness can be guaranteed. Finally, we propose countermeasures against typical attacks and show how the dependability of the discussed protocols can be improved by applying these countermeasures.

The paper is structured as follows. In Section 2, we define the dependability properties and adversarial models. In Section 3, we categorize the most relevant dependable clustering protocols and carry out sample analyses to highlight their characteristic features. We propose countermeasures and show their effect on dependability in Section 4. Finally, we summarize our results in Section 5.

2. Dependability properties and adversarial models

In this section, we detail the properties that a dependable cluster head election protocol should satisfy. Furthermore, we define an intuitive classification of the properties that we use in order to simplify the analysis of the protocols. We also introduce the three adversarial models that are considered throughout the analyses.

2.1. Dependability properties

In order to compare existing cluster head election solutions, we define a set of dependability properties. These properties are, in our opinion, important to satisfy in order for a cluster head election protocol to be considered secure and reliable. The set of properties contains basic notions (e.g., termination) and also advanced notions (e.g., non-manipulability). The definitions of the properties rely on the following notions:

- *Role:* the role of a sensor node is either cluster head or cluster member
- *Cluster head:* a sensor node is called cluster head if it has been elected as cluster head during the election process

- *Cluster member:* a sensor node is called cluster member, if it has *not* been elected as cluster head, but it is associated with a cluster head
- *Association:* the association of a cluster member is the knowledge of which cluster head it belongs to[3]

Based on these notions, we define the following properties that a secure and reliable clustering solution has to fulfill.

- *Termination:* after a finite time, every node considers the protocol as terminated
- *Completeness:* every node for which the protocol terminates knows its role
- *Consistency:* for every node A, if node A is associated with node B as its cluster head, then node B is indeed a cluster head[4]
- *Role non-manipulability:* for every node for which the protocol terminates, the attacker cannot influence the node's role
- *Association non-manipulability:* for every cluster member, the attacker cannot influence the node's association
- *Role unpredictability:* from the viewpoint of the attacker every node has the same probability to become a cluster head a priori to the election process
- *Association unpredictability:* for every node that becomes cluster member in the election process, the attacker cannot predict a priori to the election process which potential cluster head the node will associate to[5,6]
- *Role unidentifiability:* for every node, the attacker cannot identify the node's role even after observing the entire cluster head election process
- *Association unidentifiability:* for every node known by the attacker as cluster member, the attacker cannot identify the cluster head to which the node is associated even after observing the entire cluster head election process[6]

[3]Note that this knowledge might not be represented at the particular cluster member node if the protocol supports anonymous cluster head election (as in [19, 20]).

[4]Note that the case of $A = B$ is not excluded.

[5]Note that the set of potential cluster heads for the node may not contain all nodes in the network and the attacker may have a priori knowledge about that set. This fact does not violate the association unpredictability property as long as the set of potential cluster heads is large enough in accordance with the application.

[6]Note that in the special case of having only one cluster head elected (which all the member nodes associate to) and the corresponding role property being unsatisfied, we consider this property to be unsatisfied.

The termination, completeness and consistency properties formulate basic requirements for dependable cluster head election approaches (and for non-security-oriented solutions, too). The termination property requires the protocol to terminate in a finite time, i.e., dead-locks or infinite waiting times are not allowed. The completeness property can be interpreted only if the termination property is met, and requires every terminated node to have its role. According to the consistency property, if the node's role is that of a cluster member, then the node should be associated with a cluster head that further believes itself as cluster head. As a counterexample, if a cluster member node thinks that it is associated with node B, and node B does not believe itself to be cluster head, then the protocol is not consistent.

The other properties, i.e., non-manipulability, unpredictability and unidentifiability, are special features of dependable cluster head election mechanisms. It makes sense to examine these properties only if the consistency property is met. Without consistency, which is considered to be a basic property, it is meaningless to speak about more advanced requirements, as a protocol that satisfies advanced properties without satisfying the basic ones is useless in practice anyway. Later, we use this intuition to define a property verification sequence for our analysis purposes.

The two non-manipulability properties state that the adversary should not be able to alter the role or the association of the sensor nodes during the election process. This is of paramount importance, because if an adversary could do that, then he could, for example, force a node controlled by him to become cluster head all the time. Hence, he could take over the control of a significant part of the sensor network, i.e., the whole cluster of that cluster head. Alternatively, the adversary could also force its node to always become a cluster member, thus saving energy.

Though the unpredictability and unidentifiability properties seem similar, there exists a temporal difference between them. While unpredictability requires the adversary to be incapable of precisely determining the identity of the cluster head before the election process takes place, unidentifiability relaxes the same, and allows the adversary to observe the election process. Both of these requirements aim at preventing the adversary from identifying the upcoming cluster head, since that node may become the primary target of attacks. Note that in this survey we investigate the unidentifiability of the upcoming cluster head node only *before* and *during* the election, since we focus on the dependability of the election process itself. In other words, we do not address the possible identification of the cluster head after the election process (using, e.g., traffic analysis) in this survey.

The property investigation sequence that we use in our analysis is illustrated in Figure 1. It reflects how fundamental we think a property is to be fulfilled. Thus, the basic properties are located on Investigation Levels 1-3, while the advanced properties are on Investigation Levels 4-5. We move on with the investigation up to Investigation Level 4 only if the properties on the lower investigation levels are fulfilled. The advanced properties on Investigation Level 4 are checked in parallel. A property on Investigation Level 5 is examined

only if its 'peer' property on Investigation Level 4 is fulfilled. For example, the investigation of association non-manipulability is influenced only by the fulfillment of role non-manipulability and independent of the fulfillment of role unpredictability and association unpredictability.

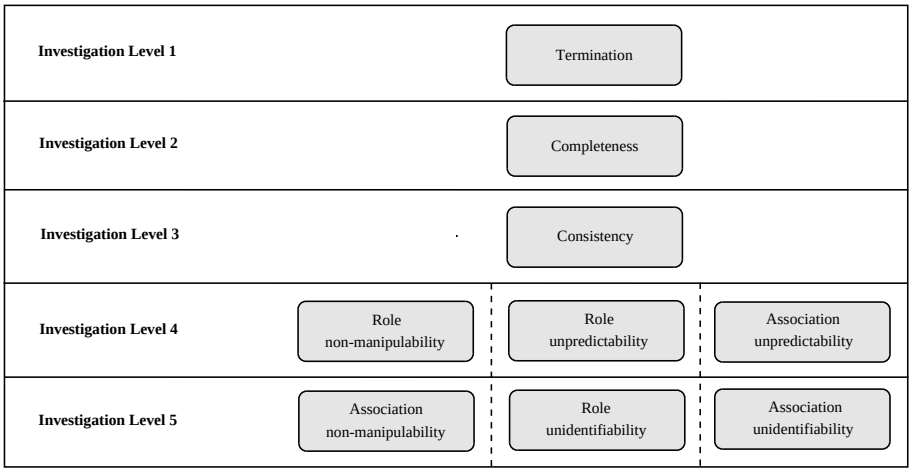


Figure 1: Property investigation sequence

2.2. Adversary models

In the corresponding literature, the interpretation of the concept of dependability is often limited to the notion of fault-tolerance. In this survey, we also investigate the cases when an adversary is present in the network. We group adversaries according to their strength into three typical categories as follows.

- *Passive adversary:* A passive adversary is only able to perform eavesdropping at any point of the network, or can even eavesdrop the whole communication of the network. As sensor nodes use wireless channels for their communication, this assumption seems valid. This adversary is also able to undertake traffic analysis or any kind of statistical analysis based on the eavesdropped information.

- *Active adversary:* In addition to the abilities of the passive adversary, an active adversary can also tamper with the wireless channel. This means, that an active adversary can replay, forge, modify and delete messages beside eavesdropping, even in a full-duplex manner[7]. However, the active adversary is still an outsider adversary, i.e., he does not know the secret keys (if any) of the sensor nodes.

[7] A full-duplex active adversary is able to receive and transmit concurrently. [21, 22]

- *Compromising adversary:* The most powerful adversary we consider is the compromising adversary. He possesses all the abilities of an active adversary. In addition, he can compromise some of the nodes, by which we mean that he can access the secret information (e.g., keys) stored in the compromised nodes, and he can also change the inner state and behavior of the compromised nodes. This means that the compromised nodes can exhibit arbitrary behavior, and thus, this case corresponds to what is usually called the Byzantine[8] fault model.

Note that in case of a compromising adversary, the compromised node cannot be required anymore to fulfill any of the dependability properties defined above. Therefore, in case of node compromise, we investigate the influence of the compromised node on the remaining (i.e., not compromised) part of the network, and confine the evaluation of the properties only to that part.

As it readily follows from the description, there is an inherent dependency of the adversarial models that is illustrated in Figure 2. Here, $A \rightarrow B$ means that the fulfillment of a property assuming adversarial model B requires the fulfillment of the same property assuming adversarial model A.

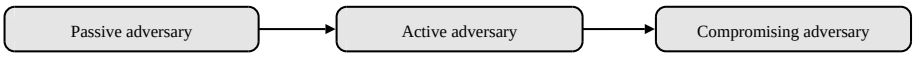


Figure 2: Dependency graph of adversarial models

In order to carry out the analyses, we investigated the fulfillment of the properties under the different adversarial models. The number of necessary investigations (i.e., property–adversarial model pairs) is usually narrowed based on the sequencing among the properties (given in Figure 1) and on the dependency among the adversarial models (see Figure 2). For example, if the consistency property is not met assuming an active adversary, then it is also not met, by definition, assuming a compromising adversary. Moreover, in this case we do not investigate the role non-manipulability property assuming an active or compromising adversary either, and so on. In Section 3, we present a comprehensive analysis of the most relevant dependable clustering solutions from the literature.

3. Analysis of state-of-the-art dependable clustering protocols

In this section, we analyze the state-of-the-art secure and reliable clustering protocols and their cluster head election mechanisms in WSNs. We investigate the fulfillment of the defined dependability properties in case of the different adversarial models.

[8]The Byzantine term, introduced originally in [23], is used to denote participants in the network whose actions may not conform with the relevant protocol specifications.

In our analysis, we considered 29 current, dependability-oriented sensor network clustering papers. During the first round of analysis, we eliminated almost one third of them from further investigations, namely [24, 25, 26, 27, 28, 29, 30, 31, 32]. These works either only discuss the application of clustering for given purposes, describe solely the basic idea of a cluster head election method without specifying the protocol in detail, consider reliability in a different meaning, or assume a heterogeneous sensor network with differences in the nodes' capabilities. Therefore, these papers do not fit into our targeted framework.

Regarding the remaining papers, we systematically investigated the fulfillment of the defined dependability properties assuming the passive, active and compromising adversarial models, respectively. To simplify the investigation we followed our property investigation sequence (see Section 2.1). We found that the protocols in these papers can be grouped into five categories according to the property pattern they exhibit. The five property patterns are illustrated in Table 1.

In what follows, we discuss the five property patterns in detail. We list all the papers belonging to a given category, furthermore, as an example we also analyze the only, or, where possible, two protocols in that category and point out why the different properties are satisfied or violated by the protocol(s).

3.1. Property pattern 5-2-2: Non-manipulable protocols from the viewpoint of a passive attacker

The protocols proposed in [33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44] satisfy the termination, completeness, consistency, role and association non-manipulability properties assuming a passive adversary, but only the termination and completeness properties[9] assuming an active or compromising adversary. We refer to this as *property pattern 5-2-2*, as five properties are met in case of a passive adversary, and two in case of both an active and a compromising adversary (see Table 1a). We discuss two protocols in detail in this category, namely [39] and [42].

3.1.1. A dependable clustering protocol for survivable underwater sensor networks

Description. In [39], a dependable clustering protocol is proposed to provide a survivable cluster hierarchy against cluster head failures in underwater sensor networks (UWSNs). The proposed algorithm is applicable to any wireless sensor networks not just to UWSNs. The protocol selects a primary and a backup cluster head for each cluster member to avoid re-clustering in case of failure. The network consists of homogenous sensor nodes, which are in one of the following three states during the clustering mechanism: *cluster head*, *cluster*

[9]Note that even if the fulfillment of the termination property is not clear from the protocol description we still assume it in case of all the three adversarial models. For further explanation refer to Section 4.1.

Table 1: Patterns of dependability properties (notations: P – passive attacker, A – active attacker, C – compromising attacker, ✓ – satisfied, ✗ – unsatisfied, – – not investigated)

(a) Property pattern 5-2-2

	P	A	C
Termination	✓	✓	✓
Completeness	✓	✓	✓
Consistency	✓	✗	✗
Role non-manipulability	✓	–	–
Association non-manipulability	✓	–	–
Role unpredictability	✗	–	–
Association unpredictability	✗	–	–
Role unidentifiability	–	–	–
Association unidentifiability	–	–	–

(b) Property pattern 6-4-2

	P	A	C
Termination	✓	✓	✓
Completeness	✓	✓	✓
Consistency	✓	✓	✗
Role non-manipulability	✓	✗	–
Association non-manipulability	✓	–	–
Role unpredictability	✗	✗	–
Association unpredictability	✓	✓	–
Role unidentifiability	–	–	–
Association unidentifiability	✗	✗	–

(c) Property pattern 7-2-2

	P	A	C
Termination	✓	✓	✓
Completeness	✓	✓	✓
Consistency	✓	✗	✗
Role non-manipulability	✓	–	–
Association non-manipulability	✓	–	–
Role unpredictability	✓	–	–
Association unpredictability	✓	–	–
Role unidentifiability	✗	–	–
Association unidentifiability	✗	–	–

(d) Property pattern 9-7-2

	P	A	C
Termination	✓	✓	✓
Completeness	✓	✓	✓
Consistency	✓	✓	✗
Role non-manipulability	✓	✗	–
Association non-manipulability	✓	–	–
Role unpredictability	✓	✓	–
Association unpredictability	✓	✓	–
Role unidentifiability	✓	✓	–
Association unidentifiability	✓	✓	–

(e) Property pattern 9-9-6

	P	A	C
Termination	✓	✓	✓
Completeness	✓	✓	✓
Consistency	✓	✓	✓
Role non-manipulability	✓	✓	✓
Association non-manipulability	✓	✓	✗
Role unpredictability	✓	✓	✓
Association unpredictability	✓	✓	✗
Role unidentifiability	✓	✓	✓
Association unidentifiability	✓	✓	–

member, or *cluster head candidate* state. Initially, each node is in the cluster head candidate state.

The protocol is divided into three phases: *initialization*, *clustering* and *finalization* phase. In the initialization phase, each node discovers its one-hop neighbors, and maintains an uncovered neighbor set which contains these neighbors being still in cluster head candidate state. From this set, each node generates all the possible combinations of clusters, which means that if a node has n uncovered neighbors it will generate 2^n potential clusters. Among all potential clusters, a candidate selects a cluster as its *qualified cluster* with minimum average cost, which is computed from a given function based on energy consumption and residual energy of the cluster members.

In the clustering phase, each candidate sends the average cost of its qualified cluster to all candidates within its two-hop range, and collects the average costs sent by the other candidates. Candidates with the minimum average cost within their two-hop range will change their status to cluster head and send an *invite* message to the members of their qualified cluster. Candidates which received the invite message change their status to cluster member. Further, as an acknowledgement to the invite message, they broadcast a *join* message and add the sender of the invite message to their cluster head list as their primary cluster head. If no invite message is received in a given time period, the candidate will stay in the same state and reselects its qualified cluster, because some nodes in the former qualified cluster might have changed their status. After a node becomes a cluster member, it will keep monitoring the invite messages of other nodes, and adds the sender of these messages to its cluster head list. Clustering will be repeated until all candidates become either a cluster head or a cluster member.

To guarantee that every cluster member has at least two elements in its cluster head list, a finalization phase is run. In this phase, all cluster members return to cluster head candidate state, and the clustering protocol is performed one more time. As a result, each cluster member will be covered by at least one more cluster head or will become a cluster head itself. Combining the two cluster head lists excluding the primary cluster head, we get the backup cluster head list, in which the cluster head with the minimum distance to the cluster member will be selected as the backup cluster head.

Analysis. After a given time T, the protocol will terminate, even assuming a compromising attacker. However, an active attacker can replay the average cost message with a very small cost value and delay the termination. If the attacked node did not receive any invite messages, it stays in cluster head candidate state, and reselects its qualified cluster. If a predefined timeout t_1 is reached, the whole clustering mechanism stops and nodes which are still in cluster head candidate state will represent themselves in the network. Therefore, the protocol satisfies the completeness property even in the case of a compromising attacker.

However, the consistency property is satisfied only in case of a passive attacker, similarly to the non-manipulability properties. An active attacker can bring the network into an inconsistent state by fabricating and replaying invite

and join messages. If a candidate A declares itself as a cluster head and sends an invite message to the members of its qualified cluster, the attacker can jam the invited nodes. After this, the attacker sends a fabricated join message to A on behalf of an invited node B. Now, A believes that B is its cluster member, but B is still in cluster head candidate state in fact. Moreover, since other nodes believe B to be a candidate, they can invite B to be their cluster member.

Based on the given cost function, even a passive attacker can predict the roles and the associations of the nodes. By eavesdropping the cost messages in the i^{th} execution of the clustering phase, a passive attacker gets knowledge about the average costs of the possible clusters. Using this knowledge, the attacker can gain information about the outcome of the $(i+1)^{\text{st}}$ execution of the clustering phase. For example, if a node has remarkably higher remaining energy than the other nodes in the network, it has a higher chance to become cluster head having its lower average cost. Knowing this, an attacker can foretell that this high-energy node will consecutively become cluster head as long as its energy level is still high. Moreover, the protocol elects backup cluster heads for each of the nodes among the cluster heads whose *invite* messages have been heard previously. This latter election is based on the physical distance between the nodes, which may be known to the attacker. Therefore, backup cluster heads can be easily predicted even by a passive adversary. Hence, the unpredictability properties cannot be assured.

3.1.2. Resilient Cluster Leader Election Protocol

Description. The resilient cluster leader election protocol [42] is intended to elect a cluster head from a previously defined cluster in a resilient way. The main goal is to prevent an attacker from influencing the election. The authors apply Blundo's key pre-distribution scheme [45] in order to establish pairwise keys between the sensor nodes, and one-way key chains for broadcast authentication. The nodes' ID and key chain commitment are tied together allowing for the latter to be easily verified.

The operation of the protocol consists of three phases. In the initialization phase, each node computes the same ordered list L_{candi} of the cluster members. The list is computed by a shuffle algorithm using the public ID of all nodes and their key chain commitments.

In the anti-spoofing announcement phase, each node reveals a secret key from the one-way hash chains it owns. The key can be a value from the YES chain, or a value from the NO chain. The value from the YES chain means that the node has enough remaining energy to be a cluster head in the upcoming round. The usage of the NO chain means that the node cannot be a cluster head in the following round due to its flat battery. None of these secret values can be predicted by an attacker due to the one-way property of the key chains, but can be verified by all nodes based on the node IDs.

In the third phase, which is the distributed decision making, each node selects its cluster head. The cluster head is the next active node after the current cluster head in the candidate list. A node is active if it has revealed a YES key in the current round. The inactive nodes are neglected. The elected

cluster head confirms its state by a simple confirmation and recovery protocol, where the cluster head discusses its view about other nodes (last revealed secret keys) with every other node. After the confirmation and recovery protocol, the elected cluster head can be sure, that it is the only one cluster head in its partition.

Analysis. The termination property holds for the protocol, as after the confirmation and recovery protocol, the election is terminated. The protocol is complete, as every node can decide about its role on its own in the worst case, i.e., even if an active adversary applies jamming or a compromising adversary is present in the network. Consistency, however, is ensured by the confirmation and recovery protocol only when facing a passive attacker. An active attacker can mount a selective jamming attack through which a cluster member node can be convinced that another node is cluster head while it is not.

The latter attack has two steps: First, the attacker has to jam some broadcast elements of the YES chains of other nodes, in particular, the YES key of the expected cluster head. This will imply a different L_{candi} for the attacked node, say node A. This node will then select a cluster head, say node B, different of the other nodes' cluster head and notify B by sending him all the received YES and NO keys. When node B receives A's messages and realizes that A has missed some keys, it will try to alert A. However, if the latter message is jammed by the attacker, then node A will think that node B is its cluster head, while B considers itself as a regular node.

The non-manipulability properties are ensured, by definition, when assuming a passive attacker. Regarding unpredictability, the list L_{candi} is computed using publicly broadcast values at the initialization phase in each round of cluster head election. By eavesdropping the communication in the i^{th} round, the attacker can establish a good guess about which node is going to be elected cluster head in the $(i+1)^{\text{st}}$ round. Namely, the next node in the list L_{candi} after the cluster head in round i is probably to become cluster head in round $i+1$. Therefore, unpredictability does not hold even in case of a passive adversary, and, consequently, unidentifiability cannot be satisfied either. Finally, since consistency is not met in case of an active or compromising adversary, the properties on Investigation Levels 4 and 5 are not investigated for the protocol according to the property investigation sequence (see Figure 1).

3.2. Property pattern 6-4-2: Protocol providing association unpredictability against an active attacker

In addition to the capabilities of the protocols falling under *property pattern 5-2-2*, the protocol described in [46] also satisfies the consistency property in case of an active adversary, and the association unpredictability property both in case of a passive and active adversary. Therefore, [46] falls under *property pattern 6-4-2* (see Table 1b).

3.2.1. SecLEACH

Description. In [46], the authors propose SecLEACH, an enhancement of the well-known clustering protocol LEACH [1]. The main idea of SecLEACH is to authenticate, with the help of the base station, the cluster head advertisement messages, and to apply Message Authentication Codes (MACs) to all the protocol messages. Additional methods in use are one-way key chains with delayed key disclosure proposed in [47] and random key predistribution proposed in [48].

A priori to the network deployment, three types of keys are uploaded to every sensor node. First of all, a ring of m keys with the corresponding key IDs, randomly drawn from a large key pool, to ensure secure cluster head–cluster member communication. Secondly, every sensor node gets a pairwise key shared with the base station for the secure cluster head–base station communication. Finally, a one-way key chain is generated and the last element of the chain is stored in every sensor node, which will be used for authentication purposes.

In the first step during the protocol operation, each self-elected cluster head H broadcasts an advertisement message that contains its ID, a nonce and a MAC value computed using the key shared with the base station. The base station authenticates the sender of the advertisement message using their shared key. If the sender is authentic, the base station includes the sender into its list $\mathcal{V}$ of eligible cluster head candidates. Some time later, $\mathcal{V}$ is broadcast by the base station protected with a MAC. The MAC is produced using the upcoming key k^j in the base station's key chain. k^j will be broadcast shortly after the previous message, allowing the nodes to check the authenticity of $\mathcal{V}$.

The second step in the protocol is the association of the ordinary nodes to the cluster heads. An ordinary node A chooses the closest cluster head H based on signal strength from the list $\mathcal{V}$ with which it has a common key. Then A sends a join message to H encrypted with their shared key. Finally, cluster heads broadcast the time schedule of their clusters.

Analysis. SecLEACH satisfies the termination property, since each round in the protocol terminates in a predefined time, which depends only on the local timer of the nodes. The cluster head election is based only on local decision, therefore, SecLEACH also satisfies completeness even assuming an active or compromising attacker.

Consistency is also satisfied, but only assuming a passive or an active attacker. While the authenticated protocol messages defend the protocol even from an active attacker, a compromising attacker can bring the network into an inconsistent state. A compromised node can pretend being a cluster head and send its advertisement message with a very large signal strength. The ordinary nodes with which the compromised node has a shared key will join to it. Henceforth, the compromised node can refuse acting as a cluster head, however, the cluster members deem that the network operates properly.

SecLEACH guarantees non-manipulability only in case of a passive attacker. If an attacker jams the advertisement message of a cluster head H, then H will not be on the list $\mathcal{V}$ of the valid cluster heads broadcast by the base station. Even though H will know that the advertisement message was not received by

the base station (consistency is still satisfied), H cannot be cluster head in the given round.

Role unpredictability is not satisfied, even assuming a passive attacker. If an attacker eavesdrops the communication in the i^{th} protocol round, she can be sure that the cluster heads in the i^{th} round will not be cluster heads in the $(i+1)^{\text{st}}$ round again. This means that the probability of becoming cluster head will differ for different nodes. The reason for this is that SecLEACH (similarly to LEACH) applies a load-balancing scheme that forbids cluster head nodes to re-apply for the same role as long as all other nodes have not served as cluster heads as well. Even though role unpredictability remains unsatisfied, association unpredictability is satisfied by SecLEACH even in case of an active adversary. Since the identity of the prospective cluster members cannot be foretold precisely, the attacker has no means to assess their associations, either.

At last, association unidentifiability is not satisfied by SecLEACH. In the last two message exchanges in the protocol, both the cluster members and the cluster head reveal their relationship via public message segments.

3.3. Property pattern 7-2-2: Unpredictable protocols from the viewpoint of a passive attacker

The protocols introduced in [49, 50, 51, 52] satisfy, in addition to *property pattern 6-4-2*, the role unpredictability property in case of a passive adversary, however, fail to satisfy consistency and association unpredictability when facing an active adversary. Therefore, the latter protocols belong to *property pattern 7-2-2* (see Table 1c). In this category, we discuss the protocols proposed in [49] and [50].

3.3.1. SANE based on Merkle's puzzle and homomorphic encryption

Description. The authors of [49] propose three protocols. The first solution in the paper, namely SANE based on predetermined random values falls under *property pattern 5-2-2*, while the other two proposals, i.e., SANE based on a commitment scheme, and SANE based on Merkle's puzzle and homomorphic encryption fall under *property pattern 7-2-2*. We will detail the latter protocol in this section, which applies Merkle's puzzle [53] for setting up pairwise keys between the current cluster head and the cluster members, and an additively homomorphic encryption transformation [54] for encrypting and summing up the pseudorandom contributions of the sensor nodes. These contributions together define the ID of the upcoming cluster head.

The protocol works as follows. At the end of each epoch, the current cluster head A_t applies Merkle's puzzle to establish pairwise keys k_i with each sensor nodes s_i (including itself). After that, each node computes the homomorphically encrypted sum $\sum_{j=1}^{i} E_{k_j}(r_j) = E_{k_i}(r_i) + \sum_{j=1}^{i-1} E_{k_j}(r_j)$ starting at s_1 with the sum equal to $E_{k_1}(r_1)$, where r_i is the random value of node s_i. After contributing to the sum, node s_i adds itself to the list of contributors, and unicasts the sum to node s_{i+1} reliably. If s_{i+1} has failed, s_i unicasts the sum to the next available node. The process ends when all nodes including the current cluster head have

contributed to the sum, and the sum is available to all sensor nodes. After a predefined short time in the decryption phase, the current cluster head floods all the pairwise keys k_j and the corresponding ID's to all the nodes. Each node then individually decrypts the ciphered sum using the pairwise keys that results in $R_i = \sum_j E_{k_j}(r_j) - \sum_j k_j$. In case s_i does not receive a key k_j, it explicitly requests k_j from A_t. If A_t persistently fails to deliver k_j, s_i proceeds with processing the encrypted sum without considering k_j. Finally, a mapping function is applied at each node in order to convert the random aggregate value R_i to the ID of the upcoming cluster head.

Analysis. This proposed SANE solution satisfies the termination property, since each node considers the cluster head election to be finished right after the decryption operation and the subsequent mapping. These latter operations are performed independently at each node, and terminated regardless of message losses.

The completeness and consistency properties are satisfied in case of a passive attacker. However, an active attacker can mislead a node regarding the cluster head it elects. A passive attacker cannot circumvent the establishment of the pairwise keys, neither can he prevent the encrypted sum from being unicast along the sequence of sensor nodes. On the contrary, an active attacker can interfere with the election process and imply different cluster heads for a set of nodes. For example, an active attacker can jam node A in the decryption phase, thus, A will not receive some of the pairwise keys, which will result in an altered outcome of the decryption and the mapping at node A. This means that A will elect a cluster head different from the cluster head elected by the remaining nodes. Although the authors of [49] accept this situation as normal operation, it does not satisfy our consistency property, as the cluster head elected by the jammed node does not consider itself cluster head.

The non-manipulability properties of the algorithm are assured assuming a passive attacker, because such an attacker cannot influence the computation of the ciphered sum and the subsequent operations.

The fulfillment of the unpredictability properties is an important feature of the solution at hand. A passive attacker cannot a priori determine the ID of the upcoming cluster head (role unpredictability), since the sensors contribute to the encrypted sum by a random value (i.e., the process is not deterministic). Moreover, the sum aggregating the contributions is always encrypted, thus, an attacker cannot learn anything about the contributions until the decryption phase when the keys are revealed. Since the role of the nodes is unpredictable, the algorithm also satisfies the association unpredictability property.

The unidentifiability properties, however, are violated by the algorithm even considering a passive attacker. Namely, an attacker who is able to listen to the conversation among the nodes is able to eavesdrop the homomorphically encrypted sum at the stage when all the nodes have contributed already. Later on, he is able to eavesdrop the broadcast decryption keys, decrypt the homomorphic sum, and apply the public mapping function that finally gives the ID of the cluster head. Each node will associate with this given cluster head, thus,

association unidentifiability is violated by the algorithm, too.

3.3.2. Fault-tolerant clustering in ad hoc and sensor networks
Description. In [50], the authors propose two clustering algorithms for sensor networks represented by general graphs or by unit disk graphs, from which we investigate the latter one. It is assumed that the nodes are homogenous and communicate following a synchronous model, where time is divided into rounds. Every message has a restricted size of $\mathcal{O}(\log\ n)$ bits and can contain only a constant number of node identifiers. The algorithm can be divided into two phases. In the first phase, a dominating set of cluster heads is selected, while in the second phase, this set is extended to a k-fold dominating set where every node is covered by at least k cluster heads.

The first part of the algorithm works by repeatedly decreasing the number of *active nodes*, i.e., the potential cluster heads. Initially, all nodes are active. In each round, an active node v considers only those active neighbors that are within distance Θ to v. The range Θ starts with a small value and it is doubled in every round. In the i^{th} round r_i, every active node chooses a random identifier in the range $[1 \ldots n^4]$, the width of which ensures with high probability that the IDs will be unique. After that, every active node selects among its active neighbors (in the given range Θ_i) the one with the highest ID (perhaps itself). If an active node is elected at least by one active node, it remains active. If not, it becomes passive and stops executing the first part of the algorithm. Nodes which remain active after a given number of rounds will become cluster heads.

In the second part of the algorithm, the number of neighboring cluster heads of all nodes is examined. If that number c is less than k, the given node elects randomly $k - c$ neighboring regular nodes to be its cluster heads and informs these nodes about this action. Otherwise, if there have been enough cluster heads elected, the algorithm finishes.

Analysis. The first part of the algorithm consists of a given number of rounds, while the second part is only one additional round. Since every node executes the algorithm independently of the other nodes, the execution depends only on the internal state of the nodes. Therefore, it is guaranteed that the protocol terminates within time T, even in case of a compromising attacker.

The protocol satisfies the completeness property even assuming a compromising attacker. The protocol consists of a given number of rounds, after which every node knows its role; cluster head if it was elected in every round at least by one node, cluster member otherwise.

The consistency property is satisfied only when assuming a passive attacker. An active attacker can modify the ID sent by a node A to a high value and jam the election messages sent to A. Hence, A will not know about the election and changes its status to passive, while its neighbors will think that A is still active. If this happens in the last round, the passive neighbors of A will consider A as their cluster head. Moreover, if an attacker jams the messages of A in the second phase of the protocol when node A informs its $k - c$ neighbors to be its cluster heads, A will consider them as cluster heads, even though those nodes

will not know about this. So, the system can be brought into an inconsistent state by an active attacker.

A passive attacker trivially cannot manipulate roles and associations, i.e., the role- and association non-manipulability properties are satisfied by the algorithm.

The cluster head election process is based on random numbers generated in all rounds in a distributed way. Therefore, a passive attacker cannot predict the status of the nodes before the actual round, only when the nodes disclose their randomly chosen IDs. Moreover, the attacker cannot be sure that passive nodes will be regular nodes after the first phase, since they can be elected in the second phase to ensure the k-fold dominating set property. The second election process is also a random election, and the attacker cannot predict its result before the disclosure of the selected IDs.

Since the disclosure of the random IDs happen to be in plaintext, the algorithm cannot satisfy the role unidentifiability property as a passive adversary can overhear the disclosed IDs and gain information about the cluster heads. After the cluster heads are elected, the remaining nodes will join the cluster heads of their dominating sets. This means that knowing the cluster heads determines the associations at least to some extent, since cluster member nodes will join those cluster heads that are their one hop neighbors (each cluster member will join exactly k cluster heads).

3.4. Property pattern 9-7-2: Unidentifiable protocol from the viewpoint of an active attacker

On top of *property pattern 7-2-2*, the protocol proposed in [55], beyond fulfilling all the properties in case of a passive attacker, also satisfies the consistency, unpredictability and unidentifiability properties in case of an active adversary. Therefore, the latter protocol belongs to *property pattern 9-7-2* (see Table 1d).

3.4.1. Two-round private cluster head election protocol

Description. In [55], the authors propose a cluster head election protocol that aims at preventing a passive adversary from predicting and identifying the roles and the associations of the nodes. The general idea of the proposed protocol is to make the observable behavior of each node uniform across all nodes. This is achieved by encrypting all protocol messages and requiring that all nodes send the same amount of messages in a random order. Hence, for a passive adversary, none of the nodes can be distinguished from the other nodes, meaning that the protocol leaks no information on the nodes' roles and associations.

The protocol consists of two rounds of equal duration. Each node sends exactly one encrypted message of the same size in each round. Each message is encrypted with a symmetric key shared between the originator of the message and its neighbors. The local broadcast keys used for encryption are assumed to be established between each node and its neighbors before running the cluster head election protocol. In addition, before starting, each node decides randomly if it wants to send a cluster head announcement in the first or in the second round.

At the beginning of the first round, each node A initializes a timer T_A to a random value that is smaller than the duration of the round. When timer T_A expires, node A sends an encrypted cluster head announcement message if it has not received any cluster head announcements from its neighbors and its earlier decision was to send its cluster head announcement in the first round. Otherwise, it sends an encrypted dummy message.

The second round is similar: Each node A re-initializes its timer T_A to another random value smaller than the duration of the round. When T_A expires, A sends an encrypted cluster head announcement message if it has not received any cluster head announcements from its neighbors (neither in the first nor in the second round) and its earlier decision was to send its cluster head announcement in the second round. Otherwise, it sends an encrypted dummy message.

At the end, those nodes that have sent cluster head announcements become cluster heads. The remaining nodes become cluster members, and each such node will consider the neighbor from which it received the first cluster head announcement to be its cluster head.

Analysis. The above two-round private cluster head election protocol satisfies the termination property, because each node considers the protocol terminated after the second round. The protocol also satisfies the completeness property, because each node always becomes either a cluster head or a cluster member when the protocol terminates.

In addition, if a cluster member considers a neighbor to be its cluster head, then that neighbor must have sent a cluster head announcement, and hence, it must consider itself cluster head too. This means that the protocol satisfies the consistency property, which holds even in face of an active adversary, because the messages are cryptographically protected, therefore, they cannot be easily forged, spoofed, or replayed. However, a compromised node can always behave correctly in the protocol and later on switch to the opposite role that it should play according to the outcome of the protocol. This means that the protocol cannot satisfy the consistency property in case of a compromising adversary.

Passive adversaries cannot manipulate the outcome of the protocol by definition. However, an active adversary can delete all messages destined to a given node by jamming that node, in which case, it can force that node to become a cluster head. In addition, due to the random nature of the protocol, adversaries cannot predict the roles and the associations of the nodes, and due to the special design that makes the message contents unintelligible and the observable behavior of the nodes uniform, they cannot identify roles and associations either.[10]

[10]We set aside the trivial case when the attacker jams the whole sensor network. In this case all nodes would become cluster head, which implies identical operation to the case when no cluster head is elected, i.e., when no clustering is applied.

3.5. Property pattern 9-9-6: Protection for role-based properties against a compromising attacker

The protocol introduced in [19, 20] turns out to be the most advanced from dependability point of view by having *property pattern 9-9-6* (see Table 1e). The latter protocol satisfies the consistency property even in case of a compromising adversary and provides protection for the information regarding the nodes' roles.

3.5.1. Perfectly anonymous data aggregation protocol

Description. In [19, 20], the authors propose a set of protocols for data aggregation in wireless sensor networks. The set consists of a private cluster head election protocol, a private aggregation protocol and a private query protocol. In this survey, only the election protocol is described and analyzed.

The main idea of the election protocol in [19, 20] is to hide the identity of the elected cluster head even from the other participants of the cluster, thus withstanding the effects of node compromisation. The protocol elects the cluster head from a previously given set of nodes, in other words, the nodes' cluster memberships are defined a priori.

The operation of the election protocol consists of two phases. In the first phase, each node decides if it wants to be a cluster head in the sequel. In the second phase, the nodes run an anonymous veto protocol to ensure that at least one cluster head is elected. If no cluster head is elected, all nodes return to the first phase.

In more details, the nodes in the first phase elect to become cluster head with probability p. This local decision is independent of the other nodes' decisions. In the second phase, two authenticated broadcast messages are sent by each node. In the first round, the nodes broadcast random numbers. In the second round, the nodes compute and broadcast some special products using the values from the first round. The special products depend on the nodes decision from the first phase and the messages received so far. The product of the second round values equals to 1 if no node volunteered to be cluster head (no one vetoed to the question: Is no cluster head volunteered?), otherwise it is a long random value. If no cluster head is elected, then the whole procedure is repeated.

After the election, it is sure that at least one cluster head is elected in the cluster, however, the exact number of cluster heads depends on parameter p and the actual random values selected by the nodes. The uncertainty of the number of cluster heads is coped with by the private aggregation and query protocols.

Analysis. The above described perfectly anonymous data aggregation protocol is intended to withstand passive, active and compromising attackers who want to reveal the identity of the elected cluster heads. Even though the issue of termination is not tackled in the paper, we consider it as satisfied. The protocol is complete as the roles are decided by the nodes locally.

The consistency property is met in case of passive and active attackers. In case of a compromising attacker, consistency is met with high probability if the related system parameters are accordingly chosen. Namely, a compromising attacker can volunteer to be the cluster head, and with probability $(1-p)^{(N-c)}$,

where p is the probability of being cluster head for a non-compromised node, N is the number of nodes, and c is the number of compromised nodes, it will be the only cluster head elected. After that the compromised node can be switched off, leaving the cluster without a cluster head. This would violate the consistency property if there is no other cluster head in the cluster. However, the probability of the above attack being successful can be decreased by increasing the value of parameter p. In the following, we assume that p is set to value where consistency is satisfied with high probability.

The messages are digitally authenticated and the protocol provides anonymity, so neither a passive nor an active attacker can tamper with or foretell the roles or associations of the nodes. This means that the protocol meets all the nine properties against a passive or active attacker.

A compromising attacker cannot do anything malicious with the roles, either, as long as the majority of the nodes is not compromised. The roles of the nodes are decided locally, the compromising attacker cannot manipulate or predict the role of the non-compromised nodes. The roles are also unidentifiable, because the content of the messages of the cluster heads and cluster members are provably indistinguishable. The proof is based on the Decisional Diffie-Hellmann problem. This means that the protocol meets the role non-manipulability, role unpredictability, and role unidentifiability properties.

The associations of the nodes can be manipulated by a compromising attacker, because a compromised node can volunteer to be cluster head. Even if the compromised node is not the only cluster head, it will still create new associations in the cluster. Because of the same reason, namely that compromised nodes can become cluster heads, association unpredictability is not met either. (The association unidentifiability property is not analyzed according to the property investigation sequence.)

4. Countermeasures

In this section, we propose high-level enhancement ideas that aim at mending the dependability problems detailed above. Throughout the following description, we edge along the dependability properties defined in Section 2.

4.1. Termination

Most of the protocols do not contain any considerations regarding their termination in case of adversarial activity. During our analysis, however, we always assumed that the termination property is met by each of the proposed solutions, because every protocol can be easily extended to meet this property. Namely, every node in the sensor network should have an internal clock and each protocol execution should at the latest end with a timeout after a predefined time. Another solution can be to define a threshold for the number of operations (e.g., iterations, message transmissions, etc.) after which the protocol operation has to be terminated.

A proper solution is SecLEACH [46], which, being based on LEACH, considers fixed time slots for the rounds and each node knows when the cluster

head election ends. This requires at least loose clock synchronization between the nodes, but eliminates the effect of any kind of attack on the termination property of the protocol, as nodes use only internal information to decide about the end of the election phase.

4.2. Completeness

The completeness property, namely that every node for which the protocol terminates knows its role, is always fulfilled by the different protocols because it is less demanding to meet. For example, if a node cannot determine its cluster head at the end of the election phase, i.e., it has not received any messages containing information about the cluster head, then it should elect itself cluster head and also recruit cluster members (if this is still allowed). This strategy helps maintaining the completeness of unconnected networks, where some nodes have no neighbors, and it works independently from an attacker's activity. To give an example, the Resilient Cluster Leader Election Protocol [42] applies a variant of the described strategy.

4.3. Consistency

In contrast to the completeness property, the consistency property is very hard to satisfy. It requires each node that a non-compromised cluster member is associated to to consider itself cluster head. In other words, the consistency property is not met if there is at least one cluster member node whose associated cluster head is not a cluster head indeed. The problem is that an attacker can easily mislead a selected node regarding its belief about the associated cluster head by sending forged or replayed messages to that node, or by selectively jamming that node.

Forged messages or replayed old messages can convey false information to the selected node. Therefore, such an attack can influence the cluster head election procedure. For example, if the cluster heads are elected based on their broadcast remaining energy values then an attacker may be able to jam a given message, e.g., the message of the upcoming cluster head node A, and replace it with a forged one containing a lower remaining energy value for A. This prevents node B from selecting A as cluster head. Instead, node B will elect another node with the highest remaining energy, although that node does not believe itself cluster head as it elects node A, similarly to all the other nodes in the network. Note that, unfortunately, either forging/replaying or jamming alone is usually sufficient to drive the network into an inconsistent state.

In order to circumvent message forging and replay attacks during the cluster head election, we propose to include authentication techniques (e.g., Message Authentication Code, digital signature, etc.) and anti-replay techniques (e.g., timestamping, nonces, etc.) in the communication protocols applied during the election. One should apply anti-jamming solutions (e.g., frequency hopping, evasion, etc.) against jamming attacks. Alternativey, in stationary networks, the nodes could collect information and maintain their own view of the network (i.e., neighbors, message reception frequency, etc.), thus being able to detect large-scale message losses.

Most of the protocols in this paper apply authentication and message encryption techniques, even many of those belonging to *property pattern 5-2-2*. However, the use of anti-jamming techniques is not wide-spread yet as very few of the investigated papers mention it. Still, a positive example is [42], where the authors use several re-broadcasts in order to cope with jammed messages and to keep the nodes' view of the network synchronized.

4.4. Non-manipulability

Throughout the analyses above, we investigated the non-manipulability and the other subsequent properties only if the termination, completeness and consistency properties were met. Without consistency, which is considered to be a basic property, it is meaningless to speak about more advanced requirements, as a protocol that satisfies advanced properties without satisfying the basic ones is useless in practice anyway.

The typical ways for an attacker to manipulate a node's role or association are message altering, message forging, and (selective) jamming. Message altering means here, for example, a modification of the cluster head advertisement message. Changing the remaining energy value that serves as the basis of election could be a typical example. Message forging means the creation and injection of fake messages, thus diverting the nodes from electing the appropriate cluster head. Selective jamming of a node means the deletion of some of the messages that would otherwise reach the node, thus modifying the internal state of that node (e.g., deleting the messages of better cluster head candidate, which implies that this particular node will elect another cluster head than the other nodes). Defending against these attack types requires authentication and anti-replay techniques, as already detailed in Section 4.3.

4.5. Unpredictability

Providing role unpredictability is highly important for a dependable clustering solution. An adversary with limited resources may try to destroy or mislead the nodes whose destruction or misguidance has the largest effect on the operation of the network or on the quality of the measurements produced by the network. These nodes are obviously the cluster head nodes. Therefore, hiding the identity of the upcoming cluster head nodes is of paramount importance in dependability-related applications. Association unpredictability is interpreted depending on the fulfillment of the role unpredictability property. Namely, the association unpredictability property can only be violated if the role unpredictability property is also violated. Being able to predict the associations of the cluster member nodes provides information for the attacker about the organization of the network. This may lead to targeted attacks against one or the other cluster head through compromising an associated cluster member node. As sensors are usually concerned with their vicinity regarding their measurements, the latter attack can imply distorted measurements for the whole region that the cluster head controls.

Predictability means that the attacker is able to foretell the roles and associations *before* clustering takes place (learning the roles and associations *during*

the clustering by observing the clustering process is addressed in Section 4.6). For this, the attacker needs to know the parameter values central to the clustering, like the list of cluster head candidates in [42]. These can be either learnt from previous cluster head elections, or can be publicly available system parameters. In order to prevent this attack and to provide role unpredictability, one has to have independence from one cluster head election to the other from the viewpoint of the attacker.

On the one hand, this criterion can be satisfied by using independently generated fresh random numbers. The cluster heads can be elected as a function of these numbers either in a completely distributed fashion (as e.g., in [19, 20]) or as an agreement (as e.g., in the commitment scheme-based SANE [49]). Note, however, that using randomness does not necessarily provide unpredictability. As a counterexample, in PANEL [35], the reference point can be considered random as it is calculated using a cryptographic hash function, nevertheless, the input of the hash function is the epoch number, which can be known to the attacker. Therefore, PANEL does not provide unpredictability. To mend this, one has to provide secrecy to parameters that the randomness relies upon.

On the other hand, one could use pre-shared secrets (uploaded to the nodes before the deployment) that could serve as a basis of determining the ID of the upcoming cluster head. This could be done either in a distributed way (i.e., each node could apply the same function to the secret that would give the ID of the next cluster head), or by creating secure communication channels for establishing an agreement. The latter, however, is theoretically the same as using only fresh random numbers from unpredictability point of view.

Providing association unpredictability when role unpredictability is violated can be quite problematic (however, association unpredictability is satisfied by definition if role unpredictability holds). The issue is that cluster members have physical restrictions when it comes to associating to a cluster head. Namely, a cluster member may not be able to join any cluster head as they may be out of reach. Therefore, knowing the cluster heads gives information to the attacker about the associations to some extent. To mitigate this effect, one could randomize the associations, while also taking into consideration the physical limits. A good example here is [19, 20], in which nodes are organized into a Hamiltonian cycle and every member node can be associated to every other node that becomes cluster head during the election.

4.6. Unidentifiability

Role unidentifiability and association unidentifiability are among the most challenging properties to meet. Even when compared to unpredictability, satisfying unidentifiability imposes severe requirements on the protocol. As a matter of fact, there is a temporal difference between satisfying unpredictability and unidentifiability. Unpredictability is violated if an attacker can reveal the upcoming cluster head *before* the election process. Unidentifiability is violated if an attacker can do that *after* observing the election process (see also our property investigation sequence, Figure 1). As a consequence, assuring unidentifiability is just as important as assuring unpredictability.

In order to prevent the attacker from identifying the roles and associations in the network, one has to ensure the uniformity of traffic during the cluster head election process. This can be done by requiring each node to send the same number of messages during the election process independently of its will to become cluster head or not. As a counterexample, in SecLEACH [46], only those nodes send messages during the clustering that aim at becoming cluster heads and therefore, even a passive adversary can identify these nodes. On the contrary, in [55, 19, 20], all the nodes send exactly two messages (each of the same size). Therefore, not even an active attacker is able to identify the upcoming cluster heads using traffic analysis. The drawback of traffic unification is that it introduces an overhead, as each node has to send as many messages as the node sending the most messages. It is important to note that traffic uniformity is not enough to ensure unidentifiability, but encryption is required as well. Without encryption even a passive adversary would be able to overhear and understand all the messages sent, thus he would be able to reveal the identity of the cluster head. Encryption also helps against an active adversary, but it is futile against a compromising adversary who is assumed to know the secret keys as well. This problem can be mitigated by requiring the usage of pairwise or groupwise (where a group can be, e.g., the neighbors of a sensor node) symmetric keys between the sensor nodes. In the latter case, a compromising adversary could only identify a part of the network, namely those nodes that he communicates with. Alternatively, supporting the anonymity of the elected cluster head could also thwart attacks against role unidentifiability. (Note that in [55], groupwise keys are applied, while in [19, 20], anonymity of the cluster head is supported. However, as the protocols fail to satisfy consistency and association unpredictability, respectively, in face of a compromising adversary, we did not investigate the related unidentifiability properties in accordance with Figure 1.)

5. Summary

In this paper, we studied dependable cluster head election protocols proposed for wireless sensor networks. Dependability is a general concept that includes reliability and security aspects. In order to make the analysis possible, we first refined this general concept by identifying and defining specific properties that dependable cluster head election protocols should satisfy, and we defined adversarial models under which those properties can be investigated. We also identified an investigation sequence among the defined properties and adversarial models. Then, we analyzed the most prominent state-of-the-art secure and reliable cluster head election protocols proposed in the literature under the defined adversarial models and with respect to the defined dependability properties. We found that existing cluster head election protocols fall under five categories depending on the set of properties they satisfy. We selected seven protocols for detailed presentation and analysis. These examples are meant to illustrate how different dependability properties can be satisfied or violated. Based on the understanding gained through our analysis, we identified possible

countermeasures that researchers can use in cluster head election protocols to increase the dependability of their solutions.

Our general conclusion is that most existing cluster head election protocols fail to satisfy the desired dependability properties when faced with an active attacker that can forge, inject, modify, and delete protocol messages, or an attacker that can compromise sensor nodes. In addition, most of the protocols leak information about the roles taken by the nodes and their associations even to a passive eavesdropper. This is not desirable, because it allows an attacker to easily identify the cluster head nodes which can become primary targets of physical destruction and jamming attacks. Besides leaking sensitive information, many protocols make it possible for an attacker to predict cluster heads even without observing the execution of the protocol.

To the best of our knowledge, only the cluster head election protocol proposed in [19, 20] satisfies most of the desirable dependability properties both in the passive and active adversary models. Yet, even this protocol fails in protecting the nodes' associations when the adversary can compromise some of the nodes. Hence, we identify the design of a cluster head election protocol that remains dependable in all adversarial models as an open question and an interesting research challenge.

Acknowledgements

The work described in this paper is based on results of the WSAN4CIP project[11], which received research funding from the European Community's 7th Framework Programme. Apart from this, the European Commission has no responsibility for the content of this paper. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Károly Farkas and Ádám Horváth have also been partially supported by the postdoctoral fellowship (PD 72984) of the Hungarian Scientific Research Fund (OTKA). Levente Buttyán has also been partially supported by the Hungarian Academy of Sciences through the Bolyai János Research Fellowship.

The authors are grateful to Bhavani Shankar and to the anonymous reviewers for their thoughtful and detailed feedback which has helped to improve this survey.

References

[1] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-Efficient Communication Protocol for Wireless Microsensor Networks, in: Proceedings of the Hawaii International Conference on System Sciences (HICSS 2000), Maui, Hawaii, USA.

[11] http://www.wsan4cip.eu/

[2] X. Min, S. Wei-ren, J. Chang-jiang, Z. Ying, Energy efficient clustering algorithm for maximizing lifetime of wireless sensor networks, International Journal of Electronics and Communications 64 (2010) 289–298.

[3] O. Younis, S. Fahmy, HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks, IEEE Transactions on Mobile Computing 3 (2004) 366–379.

[4] S. Soro, W. B. Heinzelman, Cluster head election techniques for coverage preservation in wireless sensor networks, Elsevier Ad Hoc Networks 7 (2009) 955–972.

[5] N. Dimokas, D. Katsaros, Y. Manolopoulos, Energy-efficient distributed clustering in wireless sensor networks, Elsevier Journal of Parallel and Distributed Computing 70 (2010) 371–383.

[6] T. Moscibroda, R. Wattenhofer, Efficient Computation of Maximal Independent Sets in Unstructured Multi-Hop Radio Networks, in: Proceedings of the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2004), Fort Lauderdale, FL, USA.

[7] A. Vaseashta, S. Vaseashta, A Survey of Sensor Network Security, Sensors & Transducers 94 (2008) 91–102.

[8] D. Boyle, T. Newe, Securing Wireless Sensor Networks: Security Architectures, Journal of Networks 3 (2008) 65–77.

[9] D. Boyle, T. Newe, Security protocols for use with wireless sensor networks, in: Proceedings of the Third International Conference on Wireless and Mobile Communications (ICWMC 2007).

[10] M. Anand, E. Cronin, M. Sherr, M. Blaze, Z. Ives, I. Lee, Sensor network security: More interesting than you think, in: Proceedings of the First USENIX Workshop on Hot Topics in Security (HotSec 2006).

[11] J. P. Walters, Z. Liang, W. Shi, V. Chaudhary, Wireless sensor network security: A survey, in: Y. Xiao (Ed.), Proceedings of Distributed, Grid, and Pervasive Computing (2007), CRC Press, 2007.

[12] Y. Wang, G. Attebury, B. Ramamurthy, A Survey of Security Issues in Wireless Sensor Networks, IEEE Communications Surveys & Tutorials 8 (2006) 2–23.

[13] Y. Zhou, Y. Fang, Y. Zhang, Securing Wireless Sensor Networks: A Survey, IEEE Communications Surveys & Tutorials 10 (2008) 6–28.

[14] X. Chen, K. Makki, K. Yen, N. Pissinou, Sensor Networks Security: A Survey, IEEE Communications Surveys & Tutorials 11 (2009) 52–73.

[15] P. Kumarawadu, D. J. Dechene, M. Luccini, A. Sauer, Algorithms for node clustering in wireless sensor networks: A survey, in: Proceedings of the 4th International Conference on Information and Automation for Sustainability (ICIAFS 2008).

[16] A. Abbasi, M. Younis, A Survey on Clustering Algorithms for Wireless Sensor Networks, Elsevier Computer Communications 30 (2007) 2826–2841.

[17] O. Younis, M. Krunz, S. Ramasubramanian, Node Clustering in Wireless Sensor Networks: Recent Developments and Deployment Challenges, IEEE Network Magazine 20 (2006) 20–25.

[18] S. Basagni, M. Mastrogiovanni, A. Panconesi, C. Petrioli, Localized Protocols for Ad Hoc Clustering and Backbone Formation: A Performance Comparison, IEEE Transactions on Parallel and Distributed Systems 17 (2006) 292–306.

[19] L. Buttyán, T. Holczer, Perfectly anonymous data aggregation in wireless sensor networks, in: Proceedings of The 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2010), IEEE, San Francisco, 2010.

[20] T. Holczer, L. Buttyán, Anonymous aggregator election and data aggregation in wireless sensor networks, International Journal of Distributed Sensor Networks (2011) 1–19. Article ID 828414.

[21] A. Proano, L. Lazos, Selective jamming attacks in wireless networks, in: Proceedings of the IEEE International Conference on Communications (ICC 2010), pp. 1–6.

[22] A. Mukherjee, A. L. Swindlehurst, A full-duplex active eavesdropper in MIMO wiretap channels: Construction and countermeasures, in: In Proceedings of ASILOMAR Conference on Signals, Systems, and Computers (2011).

[23] L. Lamport, R. Shostak, M. Pease, The Byzantine generals problem, in: N. Suri, C. J. Walter, M. M. Hugue (Eds.), Proceedings of Advances in Ultra-Dependable Distributed Systems (2009), IEEE Computer Society Press, 1995.

[24] D. Liu, Resilient Cluster Formation for Sensor Networks, in: Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS 2007), Toronto, ON, Canada.

[25] K. Sun, P. Peng, P. Ning, C. Wang, Secure Distributed Cluster Formation in Wireless Sensor Networks, in: Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC 2006), Miami Beach, FL, USA.

[26] A. Taherkordi, M. Taleghan, M. Sharifi, Dependability Considerations in Wireless Sensor Networks Applications, Journal of Networks 1 (2006) 28–35.

[27] E. D. Cristofaro, J.-M. Bohli, D. Westhoff, FAIR: Fuzzy-based Aggregation providing In-network Resilience for real-time Wireless Sensor Networks, in: Proceedings of the 2nd ACM Conference on Wireless Network Security (WiSec 2009), Zurich, Switzerland.

[28] S. Q. Ren, J. S. Park, Fault-Tolerance Data Aggregation for Clustering Wireless Sensor Network, Wireless Personal Communications 51 (2009) 179–192.

[29] R. Azarderskhsh, A. Reyhani-Masoleh, Secure clustering and symmetric key establishment in heterogeneous wireless sensor networks, EURASIP Journal on Wireless Communications and Networking 2011 (2011) 1–12. Article ID 893592.

[30] X. Fu, Z. Yu, A reliable and efficient clustering algorithm for wireless sensor networks using fuzzy petri nets, in: Proceedings of the 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM 2010), pp. 1–4.

[31] D. Sinha, R. Chaki, SRCHS A stable reliable cluster head selection protocol, in: A. zcan, J. Zizka, D. Nagamalai (Eds.), Recent Trends in Wireless and Mobile Networks, volume 162 of *Communications in Computer and Information Science*, Springer Berlin Heidelberg, 2011, pp. 35–44.

[32] X. Zhenghong, C. Zhigang, A secure routing protocol with intrusion detection for clustering wireless sensor networks, in: Proceedings of the 2010 International Forum on Information Technology and Applications (IFITA 2010), pp. 230–233.

[33] S. Basagni, C. Petrioli, R. Petroccia, Fail-Safe Hierarchical Organization for Wireless Sensor Networks, in: Proceedings of the Military Communications Conference (MILCOM 2007), Orlando, FL, USA.

[34] L. Buttyán, P. Schaffer, PANEL: Position-Based Aggregator Node Election in Wireless Sensor Networks, in: Proceedings of the 4th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2007), Pisa, Italy.

[35] L. Buttyán, P. Schaffer, PANEL: Position-based aggregator node election in wireless sensor networks, International Journal of Distributed Sensor Networks 2010 (2010).

[36] M. Demirbas, A. Arora, V. Mittal, FLOC: A Fast Local Clustering Service for Wireless Sensor Networks, in: Proceedings of the Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS 2004), Florence, Italy.

[37] G. Gupta, M. Younis, Fault-Tolerant Clustering of Wireless Sensor Networks, in: Proceedings of the IEEE Wireless Communication and Networks Conference (WCNC 2003), New Orleans, LA, USA.

[38] K. Wang, S. Ayyash, T. Little, P. Basu, Attribute-Based Clustering for Information Dissemination in Wireless Sensor Networks, in: Proceedings of the 2nd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2005), Santa Clara, CA, USA.

[39] P. Wang, C. Li, J. Zheng, H. Mouftah, A Dependable Clustering Protocol for Survivable Underwater Sensor Networks, in: Proceedings of the IEEE International Conference on Communications (ICC 2008), Beijing, China.

[40] O. Younis, S. Fahmy, P. Santi, An Architecture for Robust Sensor Network Communications, International Journal of Distributed Sensor Networks 1 (2005) 305–327.

[41] H. Zhang, A. Arora, GS3: Scalable Self-Configuration and Self-Healing in Wireless Networks, in: Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC 2002), Monterey, CA, USA.

[42] Q. Dong, D. Liu, Resilient cluster leader election for wireless sensor networks, in: Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON 2009), IEEE, pp. 1–9.

[43] Y.-J. Han, M.-W. Park, T.-M. Chung, SecDEACH: Secure and resilient dynamic clustering protocol preserving data privacy in WSNs, in: Proceedings of International Conference on Computational Science and Its Applications (ICCSA 2010), pp. 142–157.

[44] P. Nie, Z. Jin, Y. Gong, Mires++: a reliable, energy-aware clustering algorithm for wireless sensor networks, in: Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems (MSWIM 2010), pp. 178–186.

[45] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, M. Yung, Perfectly-secure key distribution of dynamic conferences, in: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptography (CRYPTO 1993), pp. 471–486.

[46] L. B. Oliveira, A. Ferreira, M. A. Vilaca, H. C. Wong, M. Bern, R. Dahab, A. A. F. Loureiro, SecLEACH – On the security of clustered sensor networks, Signal Processing 87 (2007) 2882–2895.

[47] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, D. E. Culler, SPINS: Security protocols for sensor networks, Wireless Networks 8 (2002) 521–534.

[48] L. Eschenauer, V. D. Gligor, A key-management scheme for distributed sensor networks, in: ACM Conference on Computer and Communications Security (CCS 2002).

[49] M. Sirivianos, D. Westhoff, F. Armknecht, J. Girao, Non-Manipulable Aggregator Node Election Protocols for Wireless Sensor Networks, in: Proceedings of the 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt 2007), Limassol, Cyprus.

[50] F. Kuhn, T. Moscibroda, R. Wattenhofer, Fault-Tolerant Clustering in Ad Hoc and Sensor Networks, in: Proceedings of the 26th International Conference on Distributed Computing Systems (ICDCS 2006), Lisbon, Portugal.

[51] G. Wang, G. Cho, Secure cluster head election using mark based exclusion in wireless sensor networks, IEICE Transactions (2010) 2925–2935.

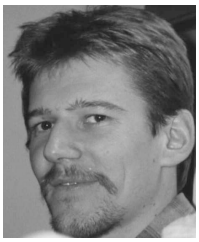
[52] G. Wang, G. Cho, Secure cluster head sensor elections using signal strength estimation and ordered transmissions, Sensors 9 (2009) 4709–4727.

[53] R. C. Merkle, Secure communications over insecure channels, Communications of the ACM (1978).

[54] G. T. C. Castelluccia, E. Mykletun, Efficient aggregation of encrypted data in wireless sensor networks, in: Proceedings of the 2nd Annual International Conference on Mobile and Ubiquitous Systems (MOBIQUITOUS 2005), San Diego, California.

[55] L. Buttyán, T. Holczer, Private cluster head election in wireless sensor networks, in: Proceedings of the IEEE Workshop on Wireless and Sensor Network Security (WSNS 2009).

Péter Schaffer received his M.Sc. and Ph.D. degrees in Computer Science from the Budapest University of Technology and Economics (BME) in 2005 and 2009, respectively. During his Ph.D. studies, he was member of the Laboratory of Cryptography and Systems Security (CrySyS) where his advisor was Prof. Levente Buttyán. Since October 2009, Péter holds a Research Associate position at the Interdisciplinary Centre for Security, Reliability and Trust (SnT) directed by Prof. Björn Ottersten, University of Luxembourg (UL). Péter's research interests are in computer security and privacy, with an emphasis on security and privacy of large-scale networks and systems. http://www.securityandtrust.lu.



Károly Farkas received his Ph.D. degree in Computer Science in 2007 from ETH Zurich, Switzerland, and his M.Sc. degree in Computer Science in 1998 from the Budapest University of Technology and Economics, Hungary. Currently he is working as an associate professor at Budapest University of Technology and Economics and at University of West Hungary, Sopron, Hungary. His research interests cover the field of communication networks, especially autonomic, self-organized and wireless mobile ad hoc networks. His core research area deals with service provisioning in mobile ad hoc networks. He has published more than 50 scientific papers in different journals, conferences and workshops and he has given a plenty of regular and invited talks. In the years past, he supervised a number of student theses, participated in several research projects, coordinated the preparation of an EU IST research project proposal and acted as program committee member, reviewer and organizer of numerous scientific conferences. He is a member of IEEE and fellow of the European Multimedia Academy.

Ádám Horváth received the M.Sc. degree in Computer Science from the Budapest University of Technology and Economics (BME) in 2007. After his M.Sc. he joined the Institute of Informatics and Economics at the University of West Hungary as a Ph.D. student under the supervision of Károly Farkas. His research interests are in security of wireless networks and mathematical modeling, mainly focused on Markovian processes. https://inf.nyme.hu/∼horvath/.

Tamás Holczer received his M.Sc. degree in Computer Science from the Budapest University of Technology and Economics (BME) in 2005. After his M.Sc. he joined the Laboratory of Cryptography and Systems Security (CrySyS) where his advisor is Prof. Levente Buttyán. His research interests are security and privacy of ad hoc networks, especially the privacy aspects of sensor networks, vehicular ad hoc networks, and RFID systems. He is a member of IEEE.

Levente Buttyán earned the Ph.D. degree from the Swiss Federal Institute of Technology – Lausanne (EPFL) in 2002, and received the M.Sc. degree in Computer Science from the Budapest University of Technology and Economics (BME) in 1995. In 2003, he joined the Department of Telecommunications at BME, where he currently holds a position as an Associate Professor and leads the Laboratory of Cryptography and Systems Security (CrySyS). His research interests are in

the design and analysis of security protocols and privacy enhancing mechanisms for wireless networked embedded systems (including wireless sensor networks, mesh networks, vehicular communications, and RFID systems), and the application of formal methods in security engineering. He has carried out research in various international research projects (e.g., UbiSecSens, SeVeCom, EU-MESH, WSAN4CIP), in which he had task leader and work package leader roles. Besides research, he has been teaching courses on network security and electronic commerce in the MSc program at BME, and gave invited lectures at various other places. He is an Associate Editor of the IEEE Transactions on Mobile Computing and an Area Editor of Elsevier Computer Communications. He is a Steering Committee member of the ACM Conference on Wireless Network Security, and he served on the Technical Program Committees of around 30 international conferences and workshops, most of which were related to wireless network security. He is a professional member of the ACM, and an affiliate of the IEEE Computer Society.