



ELSEVIER

SCIENCE @ DIRECT®

Ad Hoc Networks xxx (2006) xxx–xxx

Ad Hoc  
Networks[www.elsevier.com/locate/adhoc](http://www.elsevier.com/locate/adhoc)

## TinyPEDS: Tiny persistent encrypted data storage in asynchronous wireless sensor networks

Joao Girao<sup>a</sup>, Dirk Westhoff<sup>a,\*</sup>, Einar Mykletun<sup>b</sup>, Toshinori Araki<sup>c</sup>

<sup>a</sup> NEC Europe Ltd., 69115 Heidelberg, Germany

<sup>b</sup> University of California, Irvine, United States

<sup>c</sup> NEC Internet Systems Research Labs, Kawasaki, Japan

Received 5 May 2006; accepted 25 May 2006

### 10 Abstract

11 In wireless sensor networks there is a need to securely store monitored data in a distributed way whenever it is either not  
12 desired or simply not possible to transmit regional volatile information to an authorised recipient in real-time. In partic-  
13 ular, for wireless sensor network applications with an asynchronous character, the wireless sensor network itself needs to  
14 store the monitored data. Since nodes may disappear over time, a replicated and read-protected, but yet space- and energy-  
15 efficient, data storage is mandatory. In this work we provide and analyse an approach for a *tiny Persistent Encrypted Data*  
16 *Storage (tinyPEDS)* of the environmental fingerprint for asynchronous wireless sensor networks. Even if parts of the net-  
17 work are exhausted, restoring rules ensure that, with a high probability, environmental information from past is still  
18 available.

19 © 2006 Published by Elsevier B.V.

20 *Keywords:* Wireless sensor networks; Data encryption; Data storage; Energy consumption; Homomorphic encryption transformation

### 22 1. Introduction

23 When categorizing wireless sensor networks  
24 (WSN)s with respect to the frequency and real-time  
25 responsiveness with which the environmental infor-  
26 mation is provided to authorised parties, in princi-  
27 ple we face two cases: The first type of WSNs,  
28 which we term *synchronous* WSNs are WSNs where  
29 the monitored data is fluctual and the system is

most likely to be used for some real-time control 30  
monitoring. In contrast, we define an *asynchronous* 31  
WSN as a network that provides information to 32  
an authorised reader only seldomly. Instead, the 33  
network continuously monitors and stores environ- 34  
mental data as a function over the time and over the 35  
monitored region. Consequently, after a period of 36  
monitoring and storing, the WSN contains a fine 37  
granular environmental fingerprint. Thus, comple- 38  
mentary to the functionalities of a synchronous 39  
WSN, an asynchronous WSN also needs to provide 40  
components of a distributed database. However, 41  
there are several reasons why the implementation 42  
of a distributed database for an asynchronous 43

\* Corresponding author.

E-mail addresses: [joao.girao@netlab.nec.de](mailto:joao.girao@netlab.nec.de) (J. Girao), [dirk.westhoff@netlab.nec.de](mailto:dirk.westhoff@netlab.nec.de) (D. Westhoff), [mykletun@ics.uci.edu](mailto:mykletun@ics.uci.edu) (E. Mykletun), [t-araki@ek.jp.nec.com](mailto:t-araki@ek.jp.nec.com) (T. Araki).

44 WSN is much more challenging than that of its cen- 96  
 45 tralised counterpart located at a powerful server. 97  
 46 First and foremost, we cannot assume the devices 98  
 47 of a WSN to be tamper-resistant. Consequently, 99  
 48 assuming that the WSN is deployed in public, 100  
 49 untrusted, or even hostile environments, the entries 101  
 50 in the database must be protected and concealed. 102  
 51 Secondly, the overall storage space of a WSN is 103  
 52 extremely limited and to orders of magnitude more 104  
 53 restricted compared to its conventional database 105  
 54 counterpart. Even a large scaled WSN where thou- 106  
 55 sands or even millions of sensor nodes may serve as 107  
 56 storing cells, will not even compare to the storage 108  
 57 capacity of a traditional database. Thirdly, since 109  
 58 in WSNs energy consumption is the major metric, 110  
 59 database entries and database queries need to be 111  
 60 translated in terms of number of hops, transmission 112  
 61 distance and CPU cycles to accurately estimate the 113  
 62 resulting energy consumption of the database oper- 114  
 63 ations. Finally, single nodes, or even whole regions 115  
 64 of the WSN, may exhaust earlier than others. Con- 116  
 65 sequently, a strategy for replicating the data storing 117  
 66 at different devices is advantageous. However, repli- 118  
 67 cated storage of data at multiple sensor nodes also 119  
 68 increases the energy consumption for storage, data 120  
 69 queries and data response operations. Also, persist- 121  
 70 ent memory, which is an extremely scarce resource 122  
 71 in WSNs, should not be wasted with too much 123  
 72 redundant information. 124

73 It is the objective of this work to provide an in- 125  
 74 network approach to securely and reliably store 126  
 75 the monitored information of an asynchronous 127  
 76 WSN in an aggregated and replicated way, while 128  
 77 at the same time keeping the storage process and 129  
 78 the query process energy-efficient. Compared to 130  
 79 the continuous monitoring and storing activities of 131  
 80 the WSN, we assume database queries and database 132  
 81 responses to be rather seldom. In case of a disaster, 133  
 82 where considerable parts of the network abruptly 134  
 83 disappear, it should still be possible to request the 135  
 84 content which was originally monitored in the 136  
 85 exhausted part of the network. One can even imag- 137  
 86 ine a scenario where only once after the final collec- 138  
 87 tion of the sensor nodes, the nodes provide their 139  
 88 monitored data to the reader. Under such circum- 140  
 89 stances our approach provides a reasonable degree 141  
 90 of information accuracy and reliability with respect 142  
 91 to range queries. Range queries are the queries 143  
 92 where only events within a certain range are desired. 144  
 93 More precisely, it is the contribution of this work to 145  
 94 provide a read-protected collaborative storage man-  
 95 agement for multi-resolution storage based on the

age and/or the region of the monitored data. We 96  
 provide read-protection during data distribution, 97  
 data storage and query response. Range queries 98  
 can be addressed with different granularity accord- 99  
 ing to region and age. Furthermore, responses of 100  
 the WSN to range queries from an authorised 101  
 reader can either offer the average over a region 102  
 and/or the age, or they can offer a specific peak 103  
 value over the defined region and/or age of the mon- 104  
 itored data. For data protection, we apply a specific 105  
 class of encryption transformations which we show 106  
 to have value in the context of WSNs. 107

The rest of this paper is organised as follows. In 108  
 Section 2 we present related work. In Sections 3 and 109  
 4 we present the assumed network model and the 110  
 addressed threat model and provide an instantiation 111  
 of an optimal storage policy under such settings. In 112  
 Section 5 we introduce two classes of homomorphic 113  
 encryption transformations and in Section 6 we dis- 114  
 cuss how to apply such encryption schemes in the 115  
 context of WSNs. Section 7 presents the *tiny Persis-* 116  
*tent Encrypted Data Storage (tinyPEDS)*, our 117  
 approach for a distributed and persistent encrypted 118  
 data storage for asynchronous wireless sensor net- 119  
 works. Section 8 introduces controlled flooding 120  
 strategies for database queries and database 121  
 responses for *tinyPEDS* whereas in Section 9 we 122  
 provide restoring rules for the disaster case. In Sec- 123  
 tions 10 and 11 we analyse the security and validate 124  
 the performance of our encrypted storage architec- 125  
 ture. In the Section 12 we outline how to address 126  
 encrypted comparison. Section 13 concludes the 127  
 work and gives an outlook. 128

## 2. Related work 129

Madden et al. in [15] and Hellerstein et al. in [13] 130  
 provide an SQL-based query model for in-network 131  
 aggregation in WSNs addressing specific monitoring 132  
 durations of the network. Queries address monitor- 133  
 ing periods in the present and in the future. How- 134  
 ever, although in [13] the concept of storage points 135  
 allows a buffered streaming view of recent data, 136  
 the fully-fledged architecture to store monitored 137  
 data of an event in the past within the WSN is not 138  
 addressed in their work. 139

The problem of how to use the limited persistent 140  
 storage capacity of a sensor node to store sampled 141  
 data effectively has been discussed by Tilak, Abu- 142  
 Ghazaleh and Heinzemann in [21]. The authors 143  
 provide a cluster-based collaborative storage 144  
 approach and compare it to a local buffering tech- 145

146 nique. Collaborative storage is a promising  
147 approach for storage management because it  
148 enables the use of spatial data aggregation and  
149 redundancy control among neighboring sensors to  
150 compress the stored data and to optimize the stor-  
151 age use. However, the concealment of the data  
152 stored is not in the focus of their work.

153 In [10, 24, 3] some of the authors of this paper  
154 provided security concepts for a *synchronous*  
155 WSN. Both approaches provide end-to-end encryp-  
156 tion for real-time responsive reverse multicast traffic  
157 from the monitoring nodes to the sink node. The  
158 essential difference between approaches in [10]  
159 respectively [24 and 3] is that they provide end-to-  
160 end encryption with respect to different types of  
161 in-network processing for the aggregating and for-  
162 warding intermediate nodes. In [10, 24] we provided  
163 encryption in presence of in-network processing  
164 based on additive operations, whereas in [3] we pro-  
165 vided a concealed transmission of real-time data if  
166 comparison operations are performed at the aggre-  
167 gating intermediate nodes. For the first, the applied  
168 cryptographic technique to ensure end-to-end  
169 encryption is an *additive privacy homomorphism* like  
170 e.g. proposed by Domingo-Ferrer in [9] whereas the  
171 latter is based on the *order preserving encryption*  
172 *scheme (OPES)* by Agrawal et al. [2].

173 Recently Castelluccia, Mykletun and Tsudik pre-  
174 sented an efficient approach for the aggregation of  
175 encrypted data in wireless sensor networks [4] which  
176 is also based on the additively homomorphic prop-  
177 erty of an encryption scheme. This approach is  
178 proved to be perfectly secure. It uses different keys  
179 per sensor at the cost of a mandatory transmission  
180 of the sensors' ID list of the involved monitoring  
181 nodes. Although, when applied to large scaled  
182 WSNs with a flat structure, such a requirement con-  
183 tradicts with the aim of reducing the message size  
184 and easily results in unacceptably large messages,  
185 we believe that for cluster-based WSNs the  
186 approach is suitable.

187 However, all the aforementioned approaches  
188 [4,10,3] lack in the sense that due to their storage  
189 requirements of sensitive key material they provide  
190 only a moderate level of system security. Conse-  
191 quently, in [17] we also considered asymmetric  
192 approaches where only the public key needs to be  
193 stored on the non-tamper resistant devices.

194 The above contributions provide valuable build-  
195 ing blocks to come up with an encrypted data stor-  
196 age for asynchronous and real-time uncritical  
197 WSNs. However, the fully fledged distributed and

198 encrypted long term data storage architecture is still  
199 far from being realised.

### 3. Network and threat model 200

201 The WSN considered in this work is static and  
202 densely distributed. It is presented by a graph  
203  $\mathcal{G} = (\mathcal{N}, \mathcal{L})$  with  $|\mathcal{N}|$  nodes and  $|\mathcal{L}|$  links. Each  
204 node represents a wireless sensor node, e.g., a  
205 Mica-z mote, and each link represents a bidirec-  
206 tional communication channel over a shared me-  
207 dium, e.g., the RF channel specified by IEEE  
208 802.15.4 (security suite Null) [5,22]. There is one sin-  
209 gle stated node  $R$ , the reader device, with virtually  
210 unlimited power and storage capacity which may  
211 be mobile. After a period of monitoring and storing  
212 within the WSN the reader device is placed ran-  
213 domly but preferably in the center of the plane cov-  
214 ered by  $\mathcal{G}$ . Per epoch  $\tau$  during the lifetime of the  
215 WSN, a set of aggregator nodes  $\mathcal{A}_\tau$ , a set of for-  
216 warding nodes  $\mathcal{F}_\tau$ , a set of sensing nodes  $\mathcal{S}_\tau$  and  
217 a set of idle or sleeping nodes  $\mathcal{I}_\tau$  with  
218  $\mathcal{A}_\tau \cap \mathcal{F}_\tau \cap \mathcal{S}_\tau \cap \mathcal{I}_\tau = \emptyset$  and  $\mathcal{A}_\tau \cup \mathcal{F}_\tau \cup \mathcal{S}_\tau \cup \mathcal{I}_\tau =$   
219  $\mathcal{N}$  is elected by the network. Let  $x_{in}$  and  $x_{out}$  be  
220 the number of incoming and outgoing messages at  
221 a node  $x$ 's network layer. If  $(x_{in}, x_{out}) = (0, 1)$ , then  
222  $x \in \mathcal{S}_\tau$ , if  $(x_{in}, x_{out}) = (1, 1)$ , then  $x \in \mathcal{F}_\tau$ , and finally  
223 if  $(x_{in}, x_{out}) = (n, 1)$ ,  $n > 1$  then  $x \in \mathcal{A}_\tau$ . The latter  
224 nodes compute the aggregation function  $out =$   
225  $f(in_1, \dots, in_n)$  on the received data  $in_i$ ,  $i = 1, \dots, n$   
226 with  $f: \{0,1\}^k \times \dots \times \{0,1\}^k \rightarrow \{0,1\}^{k+l}$  and  $k + l \ll$   
227  $n \cdot k$ . For  $R$ ,  $(x_{in}, x_{out}) = (n, 0)$ . All  $x \in \mathcal{I}_\tau$  are not  
228 available for sensing and routing in epoch  $\tau$ . At  
229 epoch  $\tau + 1$ ,  $\mathcal{A}_{\tau+1}$  varies from  $\mathcal{A}_\tau$ , where  $|D|$  with  
230  $D = \mathcal{A}_\tau \cap \mathcal{A}_{\tau+1}$  is a metric for the quality of the  
231 aggregator nodes' election process. The smaller the  
232  $|D|$ , the more energy balanced the aggregator pro-  
233 cess tends to be. Consequently, also  $\mathcal{F}_{\tau+1}$ ,  $\mathcal{S}_{\tau+1}$   
234 and  $\mathcal{I}_{\tau+1}$  may differ from their counterparts in  
235 epoch  $\tau$ . Ideally,  $\mathcal{A}_\tau \cup \mathcal{F}_\tau$  form a minimum domi-  
236 nating set connecting each node  $\overline{\mathcal{N} \cap \mathcal{I}_\tau}$  such that  
237 there exists at least one bidirectional path between  
238 any pair of nodes. We term  $\mathcal{A}_\tau \cup \mathcal{F}_\tau$  the connected  
239 backbone of the WSN in epoch  $\tau$ . Since finding the  
240 minimum dominating set is an NP complete prob-  
241 lem, heuristics are needed here.

242 The attacker model we assume results from the  
243 specific device restrictions, as well as from the prop-  
244 erties of the shared medium. An attacker can eaves-  
245 drop traffic on the wireless channel, it can read data  
246 from the sensor nodes memory (since we do not  
247 assume nodes to be equipped with tamper-resistant

units) and finally, it can monitor environmental data. We are considering large scaled and relatively permanent WSNs. However, we assume that the attacker's capability to monitor environmental data is restricted due to the monitoring range and/or due to the duration of monitoring. From the above observations we follow that the classical Dolev–Yao threat model [8] does not hold in the world of WSNs. It is essential to broaden this model to also address the capturing of sensitive data which is stored in the communicating end-points.

#### 4. Disaster model and collaborative storage policy

Another aspect to consider is adapted to handle secure storage in a disaster-prone environment. Clusters of nodes might die due to what we classify as “disastrous” events. This differs from WSNs in which nodes die uniformly, typically due to energy exhaustion or malfunction. We do not exclude this type of disappearing from our thinking but the goal of a storage strategy in such a setting should be to replicate the data in such a manner that it is likely to survive regional limited disasters, while minimizing the energy costs associated with the replication of data.

Disasters are expected to occur only rarely, if at all. On the contrary, the typical querying of the environment should be supported during regular behavior. At the time a disaster occurs, we assume that the WSN administrator is notified about it and immediately tries to retrieve as much information from the network as possible. We envision large scale WSNs in which a disaster might take out a significant portion of the deployed sensors. A second assumption we make is that we have an estimate of the maximum damage range of possible disasters. By using this knowledge, we can determine the distance needed between two nodes, one replicating the other's data, by which at least one of the nodes will survive a disaster. Let  $r$  represent the disaster radius, then a pair of nodes need to be stored at a distance of at least  $\gamma = 2r$  apart. We now describe a replicated storage policy optimized for restoring and energy saving under such settings. There is a cost associated with every additional node at which a sensor's data is replicated. An optimal replication strategy should therefore ensure that data is only replicated as much as is necessary to retain it after a disaster. Obviously, the minimum number of replication nodes is one, and this node should be located at least  $\gamma$  distance away from the originating

sensor and not much further to save transmission energy. Any larger number of replication nodes would imply a larger cost, although it would most likely increase the robustness of the WSN. To summarize: Under the assumptions

- *Assumption 1*: due to possible disasters, nodes tend to die in clusters, and
- *Assumption 2*: before the WSN is rolled-out we have some way of approximating the upper radius  $r$  of a disaster area, within which nodes are disabled.

The most optimal (energy) strategy is to pair up every node  $x \in \mathcal{N}$  with a corresponding replicating node  $y \in \mathcal{N}$ , such that  $dist(x,y) > \gamma$ , thus ensuring that the data originating at  $x$  survives a disaster.

Notice that the query model for distributed database entries of the WSN is different prior and after a disaster strikes. In the former, we envision a query as selecting the aggregate from a specific age and region to learn its value, while in the latter, the query might ask for any information that still resides in the network, such as to salvage as much as possible. We are very well aware of the fact that the energy-efficiency of a distributed database query, beside the concrete storage policy, also depends on the parameters which describe the concrete WSN topology. Parameters like cluster size, number of clusters, cluster levels in the hierarchy, as well as the nodes' transmission range may significantly vary with respect to the concrete WSN application. They impact the effectiveness of the database query process. However, although we continue to describe our approach for a specific *tinyPEDS* friendly set of WSN parameters, we hereby point out that *tinyPEDS* is adaptable to nearly all kinds of WSN topologies.

#### 5. Encryption schemes

Next we introduce two classes of encryption transformations that we apply in the remainder of this work to the *tinyPEDS* approach. Their application ensures the encryption of environmental data for a concealed and indeed space-saving data storage. We assume two *additively* privacy homomorphisms, say  $PH_s$  and  $PH_a$ , which we define as follows:

**Definition 1** (*symmetric additive privacy homomorphism* ( $PH_s$ )). Let an encryption transformation be

298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345

346  $E:K \times Q \rightarrow R$  and the corresponding decryption  
 347 function be  $D:K \times R \rightarrow Q$ . Given  $a_1, a_2 \in Q$  and  
 348  $k \in K$ , a  $PH_s$  shall be based on a symmetric key  $k$ .  
 349 It provides

$$352 \quad a_1 + a_2 = D_k(E_k(a_1) \oplus E_k(a_2)). \quad (1)$$

353 In **Definition 1** the symbol “+” represents an addi-  
 354 tively operation on words from the plaintext alpha-  
 355 bet and the symbol “ $\oplus$ ” represents the  
 356 corresponding additive operation on words from  
 357 the ciphertext alphabet.

358 **Definition 2** (*asymmetric additive privacy homomor-*  
 359 *phism* ( $PH_a$ )). Let an encryption transformation be  
 360  $E:K_p \times Q \rightarrow R$  and the corresponding decryption  
 361 function be  $D:K_q \times R \rightarrow Q$ . Given  $a_1, a_2 \in Q$  and  
 362  $(p, q) \in (K_p, K_q)$ , a  $PH_a$  is based on a public/private  
 363 key-pair  $(p, q)$ . It provides

$$366 \quad a_1 + a_2 = D_q(E_p(a_1) \diamond E_p(a_2)). \quad (2)$$

367 In similarity with Eq. (1) from the **Definition 1** for  
 368 the  $PH_a$ , the symbol “ $\diamond$ ” in the **Definition 2** is the  
 369 additive operation on words from the ciphertext  
 370 alphabet.

371 Due to its homomorphic properties, a direct  
 372 application of such schemes for simple encryption  
 373 purposes is only of moderate interest since mallea-  
 374 bility may destroy chosen-ciphertext security. How-  
 375 ever, for some security concepts exactly this  
 376 homomorphic feature is a necessary prerequisite.  
 377 In the remainder of this work we apply the  $PH_s$  pro-  
 378 posed by Castelluccia, Mykletun and Tsudik [4]  
 379 with

$$381 \quad E_k(a) = a + k \bmod m \quad (3)$$

382 and

$$384 \quad D_k(a) = E_k(a) - k \bmod m \quad (4)$$

385 where  $a \in [0, m - 1]$  (respectively  $Q$ ) and  $k \in [0, m -$   
 386  $1]$  (respectively  $K$ ). Let  $c_1 = E_{k_1}(a_1)$  and  $c_2 = E_{k_2}(a_2)$   
 387 with  $k = k_1 + k_2$  and  $a_1, a_2 \in [0, m - 1]$ , then  
 388  $D_k(c_1 + c_2) = a_1 + a_2$ . Note that  $k$  is a keystream  
 389 that can be generated by a streamcipher, such as  
 390 RC4, keyed with a node’s secret key and a unique  
 391 nonce.

392 One candidate for a  $PH_a$  which fulfills the  
 393 requirement of Eq. (2) is the encryption transforma-  
 394 tion proposed by Okamoto and Uchiyama [18].  
 395 Their scheme provides a strong security level,  
 396 namely a security as secure as factoring. However,  
 397 the minimum size of each plaintext  $a$  with  $|a| \leq$   
 398 341 bits is always  $|E_k(a)| = 1024$  bits and the execu-

tion times for encryption are approximately two 399  
 times that of an ECC signature generation with a 400  
 key size of 163 bits which is in the range of 6s to 401  
 10s on the Mica-z Motes reference platform. Com- 402  
 pared to the process of transmitting or receiving 403  
 over a moderate transmission distance, the Okam- 404  
 oto and Uchiyama encryption consumes roughly 405  
 ten times more energy based on the same amount 406  
 of data. The addition operation on encrypted data 407  
 is neglectable with respect to its energy-consump- 408  
 tion. Other candidates for a  $PH_a$  are presented in 409  
 [19], namely an embodiment of the Naccache and 410  
 Stern cryptosystem, an elliptic curve version of the 411  
 Okamoto and Uchiyama cryptosystem and an 412  
 encryption transformation by Paillier. A more 413  
 promising  $PH_a$  candidate for the requirements of 414  
 an energy-restricted WSN is the appliance of the 415  
 ElGamal public-key encryption [16] on elliptic curve 416  
 ( $E$ ) points [17]. The EC-ElGamal encryption scheme 417  
 is based on the EC discrete logarithm problem 418  
 (ECDLP). Such a choice reduces the size of the 419  
 ciphertext to two times the key-length which is typ- 420  
 ically 163 bits when using standard elliptic curves. 421  
 We apply 422

$$M = \text{map}(a) \quad (5)$$

$$E_p(M) = (R, S) \quad \text{where } R = kG, \quad S = M + kY \quad (6) \quad 424$$

with the public key  $(E, p, G, Y)$  (respectively  $K_p$ ) with 425  
 $G, Y \in F_p$  and  $G$  be a generator point. The function 426  
 $\text{map}()$  is a deterministic mapping function used to 427  
 map plaintext values  $a$  into “plaintext” curve points 428  
 $M$  and vice versa such that  $\text{map}(a_1 + a_2) =$  429  
 $\text{map}(a_1) + \text{map}(a_2)$ . Decryption subsequently ap- 430  
 plies the reverse mapping function  $\text{rmap}()$  431

$$D_q(E_p(M)) = -xR + S = -xkG + M + xkG \quad (7)$$

$$a = \text{rmap}(M) \quad (8) \quad 433$$

with the private key (respectively  $K_p$ ). In [17] we pre- 434  
 sented candidates for a mapping function sustaining 435  
 the homomorphic properties of the encryption 436  
 scheme. For an example of a mapping function we 437  
 refer to [Appendix A.1](#). 438

## 6. Discussion on the usage of PHs 439

The additive homomorphic property of a PH is 440  
 a feature which is of particular relevance for an 441  
 appliance in WSNs. Properly used, it provides the 442  
 conceptual framework to efficiently conceal the 443  
 environmental fingerprint of the covered region as 444

a function over the time by at the same moment considering the extreme restrictions of the destination platform and on the radio. Obviously, in such a restricted environment there is the ultimate need to collaboratively store and to collaboratively transmit data in a condensed and aggregated manner, while still providing an appropriate degree of information to the authorised reader. In this context please note that e.g. for the RFM TR 1000 radio transceiver and an Atmel ATmega 128L microcontroller the relationship between energy consumption for computation and communication is  $1 \mu\text{J}$  to transmit a single bit,  $0.5 \mu\text{J}$  to receive a single bit, and  $8 \text{ nJ}$  for processing an instruction [14]. This results in a ratio of approximately 190 for communication to computation which documents that communication is considerably more expensive due to energy consumption than computation.

Considering and comparing the usage of a  $PH_a$  and a  $PH_s$  in a WSN, a  $PH_a$  frequently not only provides a better security with respect to the hardness of the underlying mathematical problems, it also offers a better system security due to the storage policy of sensitive data. Here, even when considering a WSN with non tamper-resistant devices, the storage of the public key  $p$  on the nodes does not reveal any sensitive information in case a set of nodes gets corrupted. In contrast, when applying  $PH_s$  to WSNs the symmetric keys  $k$  are stored at the (most-likely) non-tamper resistant devices. However, for the latter scheme, the size of a ciphertext resulting from one byte plaintext depends on the chosen modulus  $m$  and is only of three bytes to four bytes [4] compared to a 41 bytes ciphertext when using e.g. the EC-ElGamal reference  $PH_a$  with a 163 bit key length. Consequently, since the message size linearly impacts the energy consumption for transmission, a  $PH_s$  is preferable from this viewpoint. With seven bytes signaling data, a  $PH_s$  encrypted TinyOS packet respectively an IEEE 802.15.4 packet are of size 10–11 bytes. However, a  $PH_a$  encrypted message of a total size of 48 bytes also fits into a single TinyOS respectively IEEE 802.15.4 packet. Nevertheless, due to its length, a  $PH_a$  encrypted message causes comparably more energy consumption for transmission than a  $PH_s$  encrypted one. Finally, even though compared to other  $PH_a$  candidates the encryption, addition and decryption for the EC-ElGamal are much more economic in terms

of energy, the EC-ElGamal should be applied as less as possible.

Another observation that impacts the design decisions for an encrypted and reliable collaborative data storage system for WSNs is the tendency that, with a broader coverage and a condensed storage of the monitored data due to range or age, there might be a stronger need to protect this information with a higher degree of security. On the contrary, an attacker who is physically located within the WSN, and who can either eavesdrop the traffic from neighboring nodes or sense environmental data within a small region itself, gains only a very limited insight of the whole system knowledge. Furthermore, if the attacker is located within the WSN only for a minor fraction of the WSN's whole lifetime her gain is even more limited.

## 7. Encrypted and aggregated data storage

Under the aforementioned requirements, limitations and observations, we propose an approach for an encrypted and aggregated collaborative data storage in WSNs which provides some striking advantages:

1. stored data is encrypted and even the storing node cannot decrypt the ciphered values,
2. transmission costs for a collaborative and distributed data storage are minimised,
3. persistent storage space is balanced over multiple sensor nodes and reduced close to the minimum,
4. even if parts of the network are exhausted, the remaining nodes of the WSN, with a high probability, still contain information to restore the database entries which got lost,
5. nodes know from which region and for which epoch they store data. However, they do not know the values they are storing,
6. since a node is aware from which region it stores data, it also knows if a distributed data query from the reader device is addressed to it or not.

The principle idea of the *tiny Persistent Encrypted Data Storage (tinyPEDS)* approach is to apply the  $PH_s$  and the  $PH_a$  to aggregate encrypted data and to ensure due to their nested arrangement a higher system security for condensed information. We apply as  $PH_s$  the encryption scheme from Castelluccia et al. and as  $PH_a$  the EC-ElGamal encryption scheme.

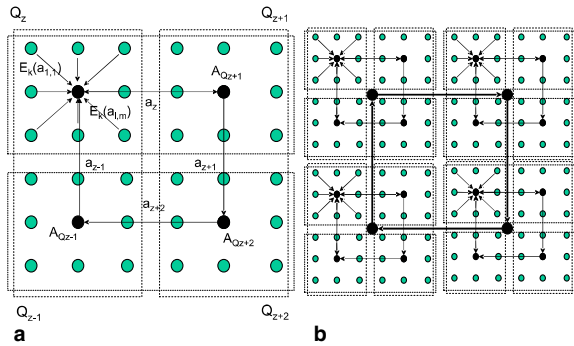


Fig. 1. (a) *TinyPEDS* cluster structure, and (b) hierarchical usage of *TinyPEDS* with quarters and subquarters.

544 After the WSN's roll-out, initially subdivide the  
 545 WSN into clusters<sup>1</sup>  $\mathcal{Q}_z$ ,  $1 \leq z \leq \omega$  with  $\mathcal{N} =$   
 546  $\bigcup_{z \leq \omega} \mathcal{Q}_z$ . Fig. 1(a) illustrates this for  $\omega = 4$ .<sup>2</sup> Elect  
 547 for each quarter per epoch  $\tau$  a new aggregator node  
 548  $A_{\mathcal{Q}_z}$  e.g. by applying a *low energy adaptive clustering*  
 549 *hierarchy* (LEACH) [12] derivative or a cluster head  
 550 election algorithm like proposed in [21]. Since aggre-  
 551 gator nodes are physically equal to other nodes due to  
 552 their heavy work load they have to be re-elected  
 553 from time to time. If necessary, due to the size of the  
 554 WSN, iteratively subdivide quarters into subquar-  
 555 ters such that ideally it holds

$$557 \forall A_{\mathcal{Q}_z} : \gamma \leq \text{dist}(A_{\mathcal{Q}_z}, A_{\mathcal{Q}_{z+1}}) \leq \gamma + \Delta. \quad (9)$$

558 In this formula we denote the margin of nodes with a  
 559 still acceptable distance for replicating data of an  
 560 adjacent cluster with  $\Delta$ . An exemplary setting with  
 561 two levels of a hierarchy is illustrated in Fig. 1(b).  
 562 We assume an aggregator node  $A_{\mathcal{Q}_z} \in \mathcal{A}_\tau$  to be  
 563 responsible in epoch  $\tau$  from the moment  $t$  to the mo-  
 564 ment  $t + 1$  for aggregating the monitored data from  
 565 time slots  $\theta$  with  $\theta \in [t, t + 1]$  received from all sensor  
 566 nodes  $s \in \mathcal{S}_t \cap \mathcal{Q}_z$ . Each sensor node  $s_{i,j} \in \mathcal{S}_t \cap \mathcal{Q}_z$   
 567 with  $1 \leq i \leq l$ ,  $1 \leq j \leq m$  of an  $l \times m$  “dimensioned”  
 568 quarter  $\mathcal{Q}_z$  monitors environmental data  $a_{i,j}$  and en-  
 569 crypts, per time slot  $\theta$ , one characteristic value by  
 570 applying the  $PH_s$  from Castelluccia et al. such that  
 571  $E_{k_{i,j}}(a_{i,j})$ . Per time slot  $\theta$ , all the sensor nodes  
 572 transmit their ciphers:  $\forall s_{i,j}; 1 \leq i \leq l, 1 \leq j \leq m$ :

$$s_{i,j} \rightarrow A_{\mathcal{Q}_z} : E_{k_{i,j}}(a_{i,j}) \quad (10) \quad 574$$

At the end of each time slot  $\theta$  of epoch  $\tau$  the aggre- 575  
 gator node  $A_{\mathcal{Q}_z}$  of quarter  $\mathcal{Q}_z$  computes 576

$$a_z^\theta = \bigoplus_{i=1}^l (\bigoplus_{j=1}^m E_{k_{i,j}}(a_{i,j})). \quad (11) \quad 579$$

$A_{\mathcal{Q}_z}$  persistently stores  $a_z^\theta$ . 580

At the end of each epoch  $\tau$ , the aggregator node 581  
 $A_{\mathcal{Q}_z}$  in addition to Eq. (11) sends the encrypted envi- 582  
 ronmental fingerprint of  $\mathcal{Q}_z$  to the aggregator node 583  
 of quarter  $\mathcal{Q}_{z+1}$  584

$$A_{\mathcal{Q}_z} \rightarrow A_{\mathcal{Q}_{z+1}} : a_z^\theta \quad (12) \quad 586$$

or, if  $z = \omega$ , to the aggregator node of the quarter  $\mathcal{Q}_1$  587

$$A_{\mathcal{Q}_z} \rightarrow A_{\mathcal{Q}_1} : a_z^\theta. \quad (13) \quad 589$$

In an optional setting,  $a_z^\theta$  can even be aggregated 590  
 over  $[t, t + 1]$  from epoch  $\tau$  before being transmitted 591  
 to  $A_{\mathcal{Q}_{z+1}}$ . 592

When receiving the environmental fingerprint 593  
 from quarter  $\mathcal{Q}_{z-1}$  at the end of epoch  $\tau$ , the aggre- 594  
 gator  $A_{\mathcal{Q}_z}$  adds  $a_z^\theta$  and  $a_{z-1}^\theta$  and applies the EC-EIG- 595  
 amal  $PH_a$  to compute 596

$$E_p(a_z^\theta \oplus a_{z-1}^\theta). \quad (14) \quad 598$$

$A_{\mathcal{Q}_z}$  persistently stores this cipher. 599

The monitoring pattern which was applied 600  
 implicitly can be described as a cyclic monitoring 601  
 wave over the quarters from which the collaborative 602  
 storage pattern is derived. Fig. 1(a) illustrates the 603  
 traffic flow of *tinyPEDS* for data storage during 604  
 the WSN's lifetime whereas Fig. 1(b) illustrates the 605  
 applicability of the *tinyPEDS* approach in a hierar- 606  
 chical setting. More concretely, in a preferable set- 607  
 ting of *tinyPEDS* we propose after the  $\theta$ th time 608  
 slot of epoch  $\tau$  within the WSN an aggregator node 609  
 $A_{\mathcal{Q}_z}$  to persistently store 610  
 611

$$\text{storage}_{[t,t+\theta]} := E_p(a_z^\theta) || E_p(a_z^\theta \oplus a_{z-1}^\theta) || \text{age}_{t-1}^{2_z \cup 2_{z-1}} || \text{age}_{t-1}^{2_z} \quad (15) \quad 613$$

with 614

$$\text{age}_{t-1}^{2_z \cup 2_{z-1}} := \text{age}_{t-2}^{2_z \cup 2_{z-1}} \diamond E_p(a_z^\theta \oplus a_{z-1}^\theta) \quad \text{for } \theta = t - 1 \quad (16) \quad 616$$

and 617

$$\text{age}_{t-1}^{2_z} := \text{age}_{t-2}^{2_z} \diamond E_p(a_z^\theta) \quad \text{for } \theta = t - 1 \quad (17) \quad 619$$

whereas 620

$$\text{age}_1^{2_z \cup 2_{z-1}} = E_p(a_z^1 \oplus a_{z-1}^1) \quad (18) \quad 622$$

<sup>1</sup> Note that clusters not necessarily need to be equally shaped nor do they need to be quadratic. However the clusters need to be ordered clockwise and in either case clusters of the same hierarchy level belong to each other.

<sup>2</sup> For a better illustration we are choosing four quarters to be the number of clusters per hierarchy level in the remainder of this work.

623 and

$$625 \text{ } age_1^{2z} = E_p(a_z^1). \quad (19)$$

626 The symbol “||” denotes the concatenation of two  
627 messages. Note that the appliance of the two intro-  
628 duced PHs in such a nested arrangement, like it is  
629 proposed for the usage in *tinyPEDS*, holds if and  
630 only if the additive operation of the  $PH_s$  on cipher-  
631 text words fits to the additive operation of the  $PH_a$   
632 on plaintext words. More precisely, the symbol  
633 “ $\oplus$ ” from the Definition 1 and the symbol “+”  
634 from the Definition 2 are the same operations. Also  
635 the set  $Q$  from  $PH_s$  needs to be contained in the set  
636  $R$  from  $PH_a$ . By using the introduced reference PHs  
637 this precondition holds.

## 638 8. Query flooding and query response

639 Next we describe how the reader device requests  
640 data from the WSN. More precisely, we discuss  
641 how database queries from the reader device can be  
642 addressed in presence of the introduced collaborative  
643 storage approach and how the *tinyPEDS* WSN  
644 architecture handles distributed database queries.

### 645 8.1. Controlled query flooding

646 A distributed database query from the reader has  
647 the following type:  $query := \langle region, duration, aggrega-$   
648  $tion, TTL, QT \rangle$  where  $region$  is any subregion of  
649 granularity  $\mathcal{Q}$  from  $\mathcal{N}$ ,  $duration$  is any subinterval  
650 over the WSN’s current lifetime  $[t_x, t_y]$  from  $[t_0,$   
651  $t_{actual}]$  with  $t_0 \leq t_x \leq t_y \leq t_{actual}$  and  $aggregation \in$   
652  $\{+, >, <\}$ . In Section 12 we show that also aggrega-  
653 tion functions of type “ $<$ ” and “ $>$ ” can be handled.  
654 The parameter  $TTL$  is the time-to-live field which  
655 indicates the flooding range of the query. The query  
656 begins with  $t_{l_{max}}$  representing the maximum hop-  
657 distance the query will travel within the WSN. The  
658 query type field  $QT := \{C, D\}$  allows the sensor nodes  
659 to differentiate between a continuous database  
660 query (“ $C$ ”) and queries which are addressed only  
661 in case a disaster appeared (“ $D$ ”). A database query  
662 from the device  $R$  of type  $\langle region, [t_x, t_y], aggrega-$   
663  $tion, ttl_{max}, C \rangle$  is handled by a receiving sensor  $s$  as  
664 denoted in Algorithm 1. In case of a disaster where  
665 major parts of  $\mathcal{Q}_z$  were lost, the reader device  $R$   
666 sends a query of type  $\langle region, duration, aggrega-$   
667  $tion, ttl_{max}, D \rangle$ . Such a query is only sent in case a  
668 continuous database query did not succeed. We

denote the behavior of a sensor node receiving a 669  
disaster query in the Algorithm 2. 670

---

### Algorithm 1. Continuous query

---

```

674 if  $s \in \mathcal{Q}_z$  AND  $\mathcal{Q}_z \subseteq region$  then
675   if  $ttl_{current} > 1$  then
676      $ttl_{current} = ttl_{current} - 1$ 
677      $s \rightarrow * : \langle region, [t_x, t_y], aggregation, ttl_{current}, C \rangle$ 
678     if  $aggregation = true$  AND  $storage_{[t, t+1]} \cap region \neq \emptyset$ 
679       AND  $t_x \leq t \leq t_y$ , then
680        $s \rightarrow R : \langle storage_{[t, t+1]} \rangle$ 
681     end if
682   end if
683 else
684    $ttl_{current} = 0$ 
685 end if

```

---



---

### Algorithm 2. Disaster query

---

```

693 if  $s \in \mathcal{N} \setminus \mathcal{Q}_z$  then
694   if  $ttl_{current} > 1$  then
695      $ttl_{current} = ttl_{current} - 1$ 
696      $s \rightarrow * : \langle region, [t_x, t_y], aggregation, ttl_{current}, D \rangle$ 
697     if  $aggregation = true$  AND  $storage_{[t, t+1]} \cap region \neq \emptyset$ 
698       AND  $t_x \leq t \leq t_y$ , then
699        $s \rightarrow R : \langle storage_{[t, t+1]} \rangle$ 
700     end if
701   end if
702 else
703    $ttl_{current} = 0$ 
704 end if

```

---

712 The fundamental difference between a continu- 712  
ous query type and a disaster query type is that 713  
the disaster query, by definition, floods the comple- 714  
mentary region of the area where the data was origi- 715  
nally monitored. Note that both types of flooding 716  
messages, respectively distributed database queries, 717  
are also applicable in case the WSN is subdivided 718  
in a hierarchy of quarters. Here, an aggregator node 719  
of the upper level adopts to the role of a reader at a 720  
lower level. Generally a WSN of  $l$  levels has  $4^l$  leaf 721  
quarters. Typically,  $region$  can be noted as 722  
 $region = \langle level_1 \triangleright level_2 \dots \triangleright level_l \rangle$  with  $level_i \in$  723  
 $\mathcal{P}(ALL)$  and  $ALL := \{\mathcal{Q}_{(i,1)}, \dots, \mathcal{Q}_{(i,4)}\}$  for  $1 \leq i \leq l$ . 724  
For example, the database query  $\langle \mathcal{Q}_{(1,2)} \triangleright \mathcal{Q}_{(2,1)} \cup$  725  
 $\mathcal{Q}_{(2,3)}, [t_x, t_y], aggregation, ttl_{max}, C \rangle$  results in an 726  
energy-saving controlled query flooding like 727  
depicted in Fig. 2(a). Its counterpart for the disaster 728  
case of nodes which were lost from  $\mathcal{Q}_z$  e.g. 729  
 $\langle \mathcal{Q}_{(1,2)} \triangleright \mathcal{Q}_{(2,1)} \cup \mathcal{Q}_{(2,3)}, [t_x, t_y], aggregation, ttl_{max}, D \rangle$  730  
results in the controlled flooding as it is illustrated in 731  
Fig. 2(b). Although in principle mobile, w.l.o.g. we 732

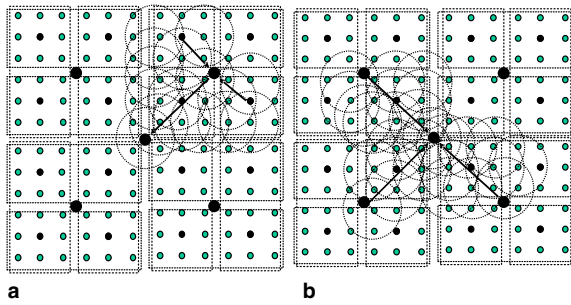


Fig. 2. Continuous query and disaster query.

733 are assuming that the reader device, when initiating  
 734 a distributed database query, is located in the center  
 735 of the WSN. Note that continuous database queries  
 736 are very energy-efficient, especially for hierarchical  
 737 WSNs. We will refine this statement in Section 11.

### 738 8.2. Aggregated data response

739 Choosing a  $PH_a$  instead of a conventional  
 740 encryption scheme for encrypting data at the end  
 741 of each epoch, has a prominent advantage in the  
 742 phase of the database response. In the most general  
 743 case, the requested information is distributed over  
 744 multiple sensor nodes which need to respond to a  
 745 data-base query. Moreover, these nodes are most  
 746 likely to be located in each other's neighborhood.  
 747 Consequently, there is the possibility to perform  
 748 in-network processing also during the data response  
 749 phase. With respect to the traffic flow, this phase is  
 750 similar to our previous work on *concealed data*  
 751 *aggregation for synchronous WSNs* [10]. The traffic  
 752 pattern of a distributed database response can be  
 753 classified as reverse multicast traffic to the reader  
 754 device. However, since the environmental data  
 755 within the *tinyPEDS* approach needs to be prepared  
 756 for long-term storage, in the context of this work we  
 757 are using a provably secure  $PH_a$  instead of a  $PH_s$  as  
 758 it was proposed in previous work for real-time mon-  
 759 itoring. Obviously, the transmission of the relatively  
 760 large cipher of a  $PH_a$  encrypted plaintext compared  
 761 to that of a  $PH_s$  encrypted one is a disadvantage.  
 762 We argue that the number of database queries is  
 763 comparably marginal to the continuous monitoring  
 764 and proceeding of secure and reliable distributed  
 765 data storage. Another argument to defend the  
 766 choice of a provably secure  $PH_a$  with respect to its  
 767 generally poorer transmission costs is the fact that  
 768 with an ElGamal encryption over elliptic curve  
 769 points there is a  $PH_a$  candidate available which is  
 770 characterised by having moderate size of the cipher-

771 text. With a ciphertext size of e.g. 41 bytes, the  
 772 ciphertext perfectly fits into a single IEEE 802.15.4  
 773 standardised packet and thus it is still suitable for  
 774 a transmission. Obviously, the optimal position of  
 775 an aggregator node, which is responsible for aggre-  
 776 gating traffic of a database response to the reader  
 777 device, is located between the responding sensor  
 778 nodes and the reader device. As a heuristic for such  
 779 a node election, and under the assumption that the  
 780 reader is located in the center of the WSN, we pro-  
 781 pose to apply the following rule:

**Heuristic:** (aggregator election for database  
 782 response) *Elect*  $s \in \mathcal{A}_{\text{actual}} \cap \text{level}_{l-1}$  which recently  
 783 received a database query of type  $\langle \text{region}, [t_x, t_y],$   
 784  $\text{aggregation}, \text{ttl}_{\text{current}}, C \rangle$  with  $\text{region} = \langle \text{level}_1 \triangleright$   
 785  $\text{level}_2 \dots \triangleright \text{level}_l \rangle$  to aggregate responses to the data  
 786 query from nodes of  $\text{level}_l$ .  
 787

788 Sensor nodes which have been elected for a data  
 789 aggregation of the response traffic according to the  
 790 above heuristic apply the additive operation  $\diamond$  of  
 791 the  $PH_a$  on the incoming ciphertexts. Each of these  
 792 ciphers represents a different epoch  $t \in [t_x, t_y]$  of the  
 793 corresponding query. This does not necessarily mean  
 794 that an aggregator node needs to perform  
 795  $(t_y - t_x - 1)$  times the operation  $\diamond$  on the received  
 796 ciphers. Since with an increasing lifetime of the  
 797 WSN, some sensor nodes will most probably have  
 798 been responsible for the storage of condensed and  
 799 ciphered data for more than one epoch, it may hap-  
 800 pen that they have stored multiple units, say  $p$  storage  
 801 units, e.g.  $\text{storage}_{[1,2]}, \text{storage}_{[2,3]}, \dots, \text{storage}_{[p,p+1]}$ .  
 802 W.l.o.g. we use a sequential ordering  $t_x \leq t_1 \leq$   
 803  $t_2 \leq \dots \leq t_p \leq t_y$  here. In such a case a sensor node  
 804 from  $\text{level}_l$  aggregates all its query relevant storage  
 805 units and computes

$$\text{storage}_{[1,p]} = \diamond_{i=1}^p \text{storage}_{[i,i+1]} \quad (20) \quad 807$$

before transmitting 808

$$s \rightarrow A_{\text{level}_{l-1}} : \langle \text{storage}_{[1,p]}, p \rangle. \quad (21) \quad 810$$

811 The responsible aggregator node  $A_{\text{level}_{l-1}}$  is either  
 812 waiting a pre-defined system time or it continues un-  
 813 til the summation of all the received  $p$  values is equal  
 814 to  $t_y - t_x$ . Subsequently it applies  $\diamond$  to the received  
 815 ciphers and transmits

$$A_{\text{level}_{l-1}} \rightarrow R : \langle \text{storage}_{[t_x, t_y]}, t_y - t_x \rangle. \quad (22) \quad 817$$

### 9. Restoring rules for disappeared quarters 818

819 We observe that with a storage policy, such as the  
 820 one introduced in Section 7, the environmental data

821 which was originally monitored, e.g. in quarter  $\mathcal{Q}_{z-1}$ ,  
 822 is partly stored in an aggregated and encrypted form  
 823 together with environmental data from another  
 824 quarter at the aggregator node  $A_{\mathcal{Q}_{z-1}}$ , from quarter  
 825  $\mathcal{Q}_{z-1}$ , as well as at the aggregator node  $A_{\mathcal{Q}_z}$ , from  
 826 quarter  $\mathcal{Q}_z$ . This observation can be used for the fol-  
 827 lowing restoring rules for any situation where a  
 828 quarter  $\mathcal{Q}_z$  with  $1 \leq z \leq 4$  or a particular  $A_{\mathcal{Q}_z}$ , which  
 829 has been elected for epoch  $t$ , is exhausted or  
 830 unavailable.

831 **Restoring Rule 1.** A collaborative database query  
 832  $\langle \mathcal{Q}_z \cup \mathcal{Q}_{z-1}, [t_x, t_y], +, ttI_{\max}, D \rangle$  can also in absence of  
 833  $\mathcal{Q}_z$  and in particularly in absence of  $A_{\mathcal{Q}_z}$  be handled  
 834 by the remaining WSN as follows: The aggregator  
 835 nodes from  $\mathcal{Q}_z$ ,  $\mathcal{Q}_{z+1}$  and  $\mathcal{Q}_{z-1}$ , which were respon-  
 836 sible in epochs  $t \in [t_x, t_y]$ , send

$$A_{\mathcal{Q}_{z-1}} \rightarrow R : E_p(a_{z-1}^0 \oplus a_z^0) \quad (23)$$

$$A_{\mathcal{Q}_{z+1}} \rightarrow R : E_p(a_{z+1}^0 \oplus a_{z+2}^0) \quad (24)$$

838  $A_{\mathcal{Q}_{z+2}} \rightarrow R : E_p(a_{z+2}^0 \oplus a_{z-1}^0). \quad (25)$

839  $R$  applies the private key  $q$  to the decryption trans-  
 840 formation of  $PH_a$  to decrypt the incoming ciphers.  
 841 Subsequently, after applying the decryption func-  
 842 tion of  $PH_s$ ,  $R$  adds the persistently stored data  
 843 from the unavailable quarter's direct neighbors  
 844 ( $\mathcal{Q}_{z-1}, \mathcal{Q}_{z+1}$ ) and subtracts the environmental finger-  
 845 print from its opposite quarter  $\mathcal{Q}_{z+2}$ . Obviously,  
 846 the final result is equal to the environmental finger-  
 847 print from region  $\mathcal{Q}_z \cup \mathcal{Q}_{z-1}$  which has been stored in  
 848 the disappeared quarter  $\mathcal{Q}_z$ :

$$D_q(a_z^0 \oplus a_{z-1}^0) + D_q(a_{z+1}^0 \oplus a_{z+2}^0) - D_q(a_{z+2}^0 \oplus a_{z-1}^0) \\ 850 = D_q(a_z^0 \oplus a_{z+1}^0) \quad (26)$$

851 **Restoring Rule 2.** A collaborative database query of  
 852 type  $\langle \mathcal{Q}_z, [t_x, t_y], +, ttI_{\max}, D \rangle$  can also in absence of  $\mathcal{Q}_z$   
 853 and in particularly in absence of  $A_{\mathcal{Q}_z}$  be handled by  
 854 the remaining sensor nodes of the WSN as follows:  
 855 Apply *Restoring Rule 1* and subtract from the result  
 856  $D_k(a_{z-1})$ :

858  $D_q(a_z^0 \oplus a_{z-1}^0) - D_q(a_{z-1}^0) = D_q(a_z^0) \quad (27)$

859 Both restoring rules hold in cases where only one  
 860 aggregator node (or quarter) per level and per epoch  
 861 is exhausted. If two or more aggregator nodes from  
 862 different quarters of the same level exhaust, data is  
 863 irrevocably lost.

## 10. Security analysis 864

865 According to the extended Dolev–Yao attacker 865  
 866 model, which is in particular applicable to WSNs 866  
 867 with non-tamper resistant devices, a complete *tiny-* 867  
 868 *PEDS* security analysis includes an evaluation of 868  
 869 the applied cryptoschemes, as well as an evaluation 869  
 870 of the proposed security architecture. Obviously, 870  
 871 the weakest component identifies the security level 871  
 872 of the complete system. 872

*Security of the cryptoschemes.* The security of the 873  
 874  $PH_a$  ElGamal encryption scheme is based on the 874  
 875 intractability of the discrete logarithm problem in 875  
 876 the group  $G$ . The group  $G$  should be chosen such 876  
 877 that (i)  $G$  should be relatively easy to apply, and 877  
 878 (ii) the discrete logarithm problem in  $G$  should be 878  
 879 computationally infeasible. The group of points 879  
 880 on an elliptic curve over binary fields appears 880  
 881 to meet these two criteria. The security of the 881  
 882  $PH_s$  proposed by Castellucia, Mykletun and Tsu- 882  
 883 dik is proven to be perfectly secure [4]. The proof 883  
 884 is listed in the [Appendix A.2](#). 884

*Security of the architecture.* We reduce the secu- 885  
 886 rity analysis of the *tinyPEDS* architecture to a 886  
 887 discussion on the storage of sensitive key mate- 887  
 888 rial on non tamper-resistant devices. Each sen- 888  
 889 sor node  $s_i$  stores the two keys  $k_i$  and  $q$ . The 889  
 890 reader device stores the keys  $\sum_{i=1}^{|S|} k_i$  and the pri- 890  
 891 vate key  $p$ . For the  $PH_a$  all sensor nodes  $s_i$  need 891  
 892 to store the same public key  $p$  and only the 892  
 893 reader device needs to store the sensitive private 893  
 894 key  $q$  (We assume the reader device to be 894  
 895 equipped with a tamper-resistant module.). Such 895  
 896 a key storage reveals no information to an 896  
 897 attacker who picks up nodes and reads the pub- 897  
 898 lic key out of one (or more) randomly chosen 898  
 899 sensor node(s). For the  $PH_s$  each node  $s_i$  stores 899  
 900 a different symmetric key  $k_i$  it solely shares with 900  
 901 the reader device. Consequently, the gain of an 901  
 902 attacker when breaking one sensor node is lim- 902  
 903 ited to the transmission link between this sensor 903  
 904 node and the actual aggregator node of this 904  
 905 epoch. All other links between the actual aggre- 905  
 906 gator node and its sensing nodes are not 906  
 907 affected. 907  
 908

909 Even if the attacker picks up a set of sensor 909  
 910 nodes his gain is always limited to a small subregion 910  
 911 of the WSN. Recall that by applying a  $PH_s$  for the 911  
 912 continuous monitoring phase instead of using a 912  
 913  $PH_a$ , we avoided having to handle disproportional 913

914 long ciphers to be continuously transmitted over  
 915 the wireless. Finally, note that the authenticity  
 916 of the involved sensor nodes is definitively  
 917 required but out of the scope of this work. Also  
 918 resilient data aggregation to deal with manipu-  
 919 lated environmental data is an issue which is not  
 920 further considered here. For both we refer to  
 921 available work on sensor networks from the  
 922 literature.

## 923 11. Performance and simulation

### 924 11.1. Performance of the reference PHs

925 We measured the costs of our reference candi-  
 926 dates for  $PH_s$  and  $PH_a$ , namely the Castelluccia  
 927 et al. approach and the ElGamal cryptoscheme on  
 928 elliptic curve points. We analysed their performance  
 929 in terms of computation for encryption, addition  
 930 and decryption as well as the resulting size of the  
 931 ciphertexts. These values are summarised in Table  
 932 1. Here  $xm + ya$  means  $x$  multiplications modulo  
 933 32 plus  $y$  additions modulo 32. We assume the  
 934 plaintext of a  $PH_s$  encryption to be 16-bit (which  
 935 provides a reasonable accuracy for almost all types  
 936 of monitored data) and for the  $PH_a$  encryption to  
 937 be 32-bit. Such plaintext values reflect the required  
 938 nested arrangement of the two homomorphic  
 939 schemes where the plaintext input of the  $PH_a$  tends  
 940 to be larger than the plaintext input of the  $PH_s$ .  
 941 More concretely we are assuming the following  
 942 parameters:

- 943 • **Castelluccia et al. [4]**: 16-bit plaintext, 32-bit  
 944 modulus, 32-bit ciphertext;
- 945 • **EC-ElGamal [17]**: 32-bit plaintext, 163-bit modu-  
 946 lus, a mapping function representing monitored  
 947 values as elliptic curve points and its reverse  
 948 mapping by conserving the homomorphic feature  
 949 of the EC-ElGamal scheme.

950 For the mapping function we assume that each  
 951 EC-ElGamal encryption at the first-level aggrega-  
 952 tors corresponds to 10 ciphertexts. Then the  
 953 decrypted aggregated value will be  $32 + 4 = 36$  bits  
 954 long (4 from  $\log(10)$ ). If we use the  $O(\sqrt{z})$  method  
 955 to balance storage versus computation for the  
 956 baby-step giant-step brute force reverse mapping  
 957 function, then we will store  $2^{18}$  pre-computed values  
 958 at the decryptor, and we will perform on average  $2^{17}$   
 959 ECC additions when reverse mapping the plaintext.  
 960 For the reverse mapping and the EC-ElGamal  
 961 encryption it takes approximately 3.3 seconds  
 962 assuming approximately 40,000 point additions  
 963 per second.  
 964

965 The computation costs and in particular the  
 966 encryption costs for our reference  $PH_a$  illustrate  
 967 our design decision to use a  $PH_a$  encryption only  
 968 once per epoch contrary to a slot-wise application  
 969 of the  $PH_s$  encryption operation. A concrete ratio  
 970 of the number of slots per epoch and the pause  
 971 times between the slots is an application specific  
 972 configuration which needs to be carefully balanced  
 973 with respect to the required system security, its  
 974 monitoring accuracy, as well as the aimed energy-  
 975 efficiency. We want to point out that in an *asynchro-*  
 976 *nous* WSN with the objective of an encrypted long  
 977 term data storage execution time is not the major  
 978 metric. Nevertheless execution time relates to micro-  
 979 controller instructions which directly translates into  
 980 energy consumption. However, as we will see below,  
 981 in particular at the aggregator node, compared to  
 982 energy consumption for communication this is still  
 983 marginal. In [11] Gura et al. show how to speed-  
 984 optimize ECC to be reasonable also in WSNs. They  
 985 implemented the ECC point multiplication in  
 986 assembly code using optimized multiplication algo-  
 987 rithms, well suited for the underlying hardware.  
 988 They measure the time to compute  $xP$  over a 160-  
 989 bit ECC curve on an Atmel ATmega128 at 8 mhz  
 990 to take 0.81 sec. For a 160-bit number, we expect

Table 1

Performance of Reference PHs:  $PH_s$  from Castelluccia et al. and  $PH_a$  EC-ElGamal

	Encryption	Add	Decryption	Bandwidth [byte]
$PH_s$ (Castelluccia et al.)	1 m + keystream	1 m	1 m + keystream	4
$PH_a$ (EC ElGamal)		$10 p $ (1)		41
	$2\frac{15}{2} p  + \frac{15}{2} m $		$\frac{15}{2} p  + 5 + map$	

All values are measured in terms of 32-bit moduli computations, i.e. 2m and 3a represent 2 modular multiplications and 3 modular additions, respectively.

991 240 ( $3/2 * 160$ ) point additions/doublings. Although  
 992 we suggest the use of a 163-bit curve this should not  
 993 cause a large difference. With respect to the *Tiny-*  
 994 *PEDS* EC-ElGamal computation the above point  
 995 multiplication occurs three times when sensors  
 996 encrypt, namely to compute  $M = mG$  (mapping),  
 997  $R = kG$  and  $S = M + kY$ . However, none of the  
 998 three require a “full” point multiplication, meaning  
 999 that the constant  $k$  is not 163 bits, and for  $R$  and  $S$ ,  
 1000 it is only 80 bits. Theoretically, this would mean  
 1001 that the computation of  $R$  and  $S$  take approxi-  
 1002 mately 0.41 seconds each, so the encryption could  
 1003 be done in below 1 second.

### 1004 11.2. Performance of *tinyPEDS*

1005 When analysing the effect of the *tinyPEDS* archi-  
 1006 tecture with respect to the energy-consumption we  
 1007 have to mainly consider the impact on the physical  
 1008 layer and on the MAC layer, e.g. the IEEE 802.15.4  
 1009 WPAN MAC protocol. The IEEE 802.15.4 proto-  
 1010 col is inherently asymmetric for a PAN coordinator  
 1011 and its neighbor nodes. This also reflects the energy  
 1012 consumption of an aggregator node  $A_{\mathcal{Q}_2}$  and the  
 1013 remaining nodes of a *tinyPEDS* quarter. Basically  
 1014 the ratio of an active period to an inactive period  
 1015 is dramatically different for the  $A_{\mathcal{Q}_2}$  compared to  
 1016 the sensing and sending nodes  $s$ . Whereas in princi-  
 1017 ple an  $s$  only needs to be active in transmit mode in  
 1018 one of the *Guaranteed Time Slots* (GTS), the  $A_{\mathcal{Q}_2}$   
 1019 needs to be active in receive mode the whole active  
 1020 period of an IEEE 802.15.4 superframe<sup>3</sup>. This trans-  
 1021 lates into roughly 8 times more energy consumption  
 1022 due to communication for  $A_{\mathcal{Q}_2}$  than for a sensing  
 1023 sensor node. Finally, note that the IEEE 802.15.4  
 1024 data field has variable size and can go up to  
 1025 102 bytes. This means that both, a  $PH_s$  encrypted  
 1026 two bytes plaintext and a  $PH_a$  aggregated and  
 1027 encrypted cipher fit into an IEEE 802.15.4 data field  
 1028 and can be transmitted in a single GTS. This indi-  
 1029 cates that, if properly adapted, the *tinyPEDS*  
 1030 impact on the WSN’s overall energy-consumption  
 1031 is indeed moderate. Both, data traffic during contin-  
 1032 uous monitoring and data traffic for aggregated  
 1033 data response do perfectly fit into the MAC layer  
 1034 architecture. Recall that distributed database que-  
 1035 ries are expected to occur only seldomly. Thus, a

1036  $PH_a$  encrypted plaintext needs to be transmitted  
 1037 only rarely.

1038 To conclude, for an indeed energy-efficient pro-  
 1039 cessing of the *tinyPEDS* architecture, one needs to  
 1040 properly adapt the *tinyPEDS* settings to the config-  
 1041 uration possibilities of the MAC layer protocol, e.g.  
 1042 the IEEE 802.15.4 standard. In particular this  
 1043 means:

- The number of nodes within one cluster  $\mathcal{Q}_2$  1044  
 should correspond to the number of GTS within 1045  
 an IEEE 802.15.4 superframe. 1046
- A *tinyPEDS* time slot  $\theta$  should correspond to an 1047  
 IEEE 802.15.4 superframe. 1048
- To avoid collisions over the wireless one should 1049  
 reduce communication during the *Contention* 1050  
*Access Period* (CAP) slots as far as possible. In 1051  
 particular this means that the epochs  $\tau$  for which 1052  
 a node is elected as aggregator node should be 1053  
 relatively large to reduce the ratio of collision 1054  
 endangered CAP communication for the re-sync- 1055  
 ronisation of the cluster members after an 1056  
 election. 1057
- According to the concrete WSN application’s 1058  
 requirements due to the monitoring accuracy, 1059  
 the inactive period of an  $A_{\mathcal{Q}_2}$  should be as long 1060  
 as possible, or, vice versa, its activity period 1061  
 should be as short as possible. 1062

1064 Recall that the relationship between communica-  
 1065 tion and computation is approximately 190:1. With  
 1066 such a ratio in mind one can easily infer that the  
 1067 above listed cross-layer design decisions are to  
 1068 orders of magnitude more energy-saving compared  
 1069 to the negative energy impact for computations at  
 1070 the microcontroller for the two reference PHs  
 1071 (Fig. 3).

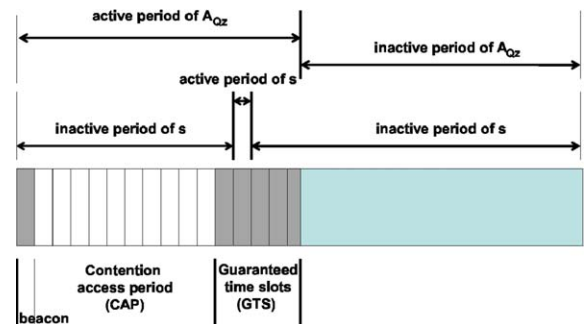


Fig. 3. Superframe structure of IEEE 802.01.4 and energy asymmetry of aggregator node vs. sensing nodes.

<sup>3</sup> This is true when  $A_{\mathcal{Q}_2}$  is synchronised with its neighbors and there is no need to communicate within the Contention Access Period (CAP).

1072 Finally note that with a setting like proposed in  
 1073 Eq. (15), the *tinyPEDS* architecture translates into  
 1074 the storage size of four times 41 bytes per epoch.  
 1075 Under the assumption that e.g. one half of the  
 1076 Mica-z Mote's 4kB RAM is available for the stor-  
 1077 age of the environmental fingerprint this means that  
 1078 each sensor node is enabled to be approximately 12  
 1079 epochs in the role of an aggregator node. We expect  
 1080 the remaining RAM to be occupied for running  
 1081 *tinyPEDS*,  $PH_s$ ,  $PH_a$  and various *tinyOS*  
 1082 components.

### 1083 11.3. Simulation

1084 For the simulation, we used the GloMoSim sim-  
 1085 ulation version 2.0 [25]. The simulation setup is listed  
 1086 in Table 2. The simulation results validate the fol-  
 1087 lowing aspects:

- 1088 • The storage policy is robust even in presence of a
- 1089 major fraction of exhausted nodes.
- 1090 • Database queries and database responses work
- 1091 properly in presence of a fraction of exhausted
- 1092 nodes.
- 1093 • Continuous database queries are efficient in
- 1094 terms of relayed messages.

1095 Figs. 4 and 5 show fractions of involved sensor  
 1096 nodes (marked black) for two particular database  
 1097 queries querying data addressed to the gray marked  
 1098 regions. The considered WSN has three hierarchy  
 1099 levels. The database query from the reader, for  
 1100 Fig. 4, is  $\langle \mathcal{Q}_{(1,1)} \triangleright \mathcal{Q}_{(2,4)} \triangleright \mathcal{Q}_{(3,4)}, [t_x, t_y], +, 20, C \rangle$   
 1101 whereas, for Fig. 5 it is  $\langle \mathcal{Q}_{(1,1)} \triangleright \mathcal{Q}_{(2,4)} \Delta \mathcal{Q}_{(3,2)} \cup$   
 1102  $\mathcal{Q}_{(3,4)}, [t_x, t_y], +, 20, C \rangle$ . Since intermediate nodes  
 1103 know their relative positions with respect to the  
 1104 query destination they can control the flooding by  
 1105 dropping query messages in case they belong to  
 1106 quarters which are not relevant for a particular  
 1107 database query. This ensures for the continuous  
 1108

Table 2  
GloMoSim simulation parameters

WSN size	400 × 400
Quadrant size	50
Number of nodes	240–407
Node's transmission range	50
Hierarchy levels	2–3
Number of quarters	4
Node placement	Random
Radio layer	CSMA
Propagation pathloss	Two-way

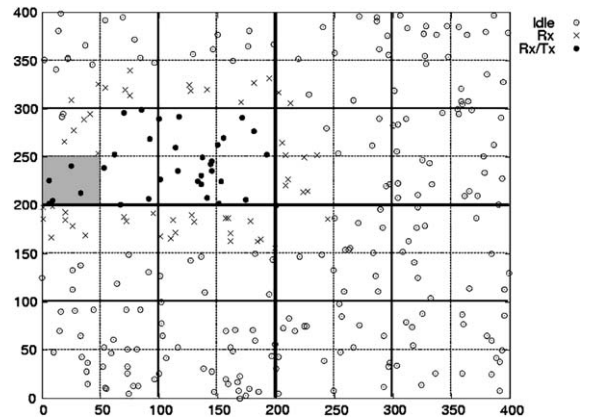


Fig. 4. Controlled flooding of the continuous database query  $\langle \mathcal{Q}_{(1,1)} \triangleright \mathcal{Q}_{(2,4)} \triangleright \mathcal{Q}_{(3,4)}, [t_x, t_y], +, 20, C \rangle$  in a WSN with  $l=3$  and  $\omega=4$ .

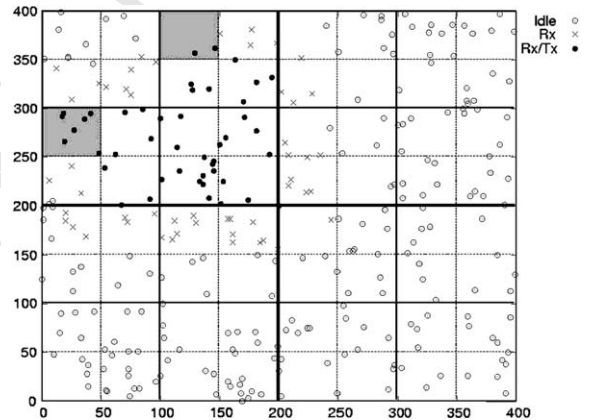


Fig. 5. Controlled flooding of the continuous database query  $\langle \mathcal{Q}_{(1,1)} \triangleright \mathcal{Q}_{(2,4)} \Delta \mathcal{Q}_{(3,2)} \cup \mathcal{Q}_{(3,4)}, [t_x, t_y], +, 20, C \rangle$  in a WSN with  $l=3$  and  $\omega=4$ .

1109 case a moderate propagation of the database query  
 1110 only to the relevant regions and subregions of the  
 1111 WSN as shown in the Figs. 4 and 5.

1112 Figs. 6 and 7 illustrate for a WSN with three lev-  
 1113 els of hierarchy how different fractions of exhausted  
 1114 nodes influence the connectivity within the WSN.  
 1115 The simulation start with 10.6 resp. 18.2 neighbour-  
 1116 ing nodes. Each curve represents the nodes' connec-  
 1117 tivity within the region size of one specific level. E.g.  
 1118 "Level 3" provides the overall connectivity of nodes  
 1119 within one single quarter of level three. Obviously  
 1120 the curve for "Level 1" illustrates the connectivity  
 1121 between all nodes of the WSN whereas deeper levels  
 1122 illustrate the connectivity within specific quarters  
 1123 of the WSN. The results acknowledge that in smaller

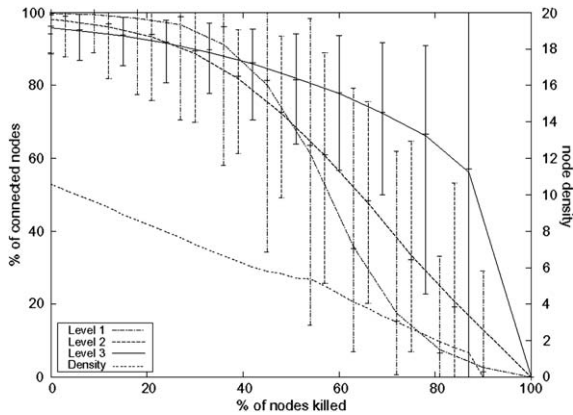


Fig. 6. Connectivity graphs for quarters of three different hierarchy levels with 240 nodes in total. The smallest quadrants are of unit size  $50 \times 50$  and the size of the WSN is  $400 \times 400$ . The transmission radius is 50 units resulting in an initial average number of 10.6 neighbors. The density curve shows the average number of one-hop neighbors per node.

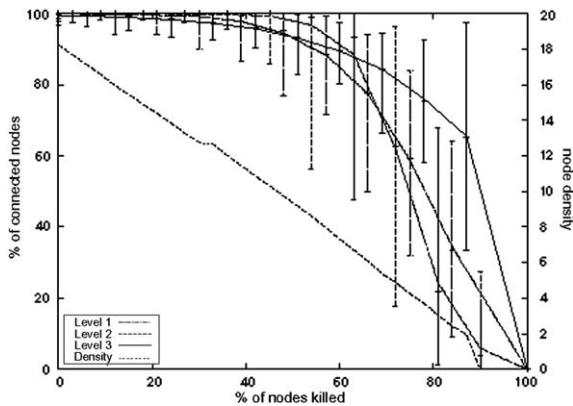


Fig. 7. Connectivity graphs for quarters of three different hierarchy levels with 407 nodes in total. The smallest quadrants are of unit size  $50 \times 50$  and the size of the WSN is  $400 \times 400$ . The transmission radius is 50 units resulting in an initial average number of 18.2 neighbors. The density curve shows the average number of one-hop neighbors per node.

1124 regions, with shorter average path length, end-to-  
 1125 end connectivity remains higher than for the whole  
 1126 WSN. Generally we can observe that the storage  
 1127 policy of the WSN needs good connectivity at all  
 1128 levels. On the contrary, for database requests and  
 1129 database responses in its continuous case the con-  
 1130 nectivity at level one is not as important as the con-  
 1131 nectivity at the deeper levels.

1132 The illustrated results validate that, under such  
 1133 settings, even up to 40% of the sensor nodes may  
 1134 exhaust over the time to still let *tinyPEDS* proceed  
 1135 well. This holds for the continuous data monitoring

and storage process but also for database queries 1136  
 and responses. If we want to translate this critical 1137  
 point into the minimum number of neighbors under 1138  
 such settings, each node should have at least six 1139  
 neighbors alive to allow *tinyPEDS* to work properly 1140  
 (connectivity in the range of 90%). With an 1141  
 increased ratio of exhausted nodes we observe the 1142  
 tendency that database queries and database 1143  
 responses may still have acceptable throughput 1144  
 whereas continuous replicating of the monitored 1145  
 data to a neighboring quarter may be more and 1146  
 more difficult. Consequently, there will be a phase 1147  
 during the lifetime of the WSN where a replicated 1148  
 data storage is already error-prone although the 1149  
 query and response process to request data from 1150  
 the past may still work properly. 1151

## 12. Encrypted comparison 1152

*TinyPEDS*, as it is described so far, offers a con- 1153  
 densed storage of encrypted data which represent 1154  
 the sum of the monitored environmental values. This 1155  
 has most value when computing the average value 1156  
 over the time or region or when performing move- 1157  
 ment detection on encrypted data. However, in [20] 1158  
 Rivest et al. have shown that any privacy homomor- 1159  
 phism, no matter if it is deterministic or probabilis- 1160  
 tic, when being homomorphic with respect to the 1161  
 comparison operations, it is insecure against cipher- 1162  
 text only attacks. Consequently one has to find other 1163  
 solutions when supporting database queries based 1164  
 on comparison operations. For this objective we 1165  
 propose to use the *order preserving encryption* 1166  
*scheme (OPES)* [2] which has been originally pro- 1167  
 posed for ordering in relational databases and which 1168  
 offers security against ciphertext only attacks. In 1169  
 previous work [3] we have shown that the *OPES* 1170  
 scheme is, in principle, portable on the sensor nodes. 1171  
 Nevertheless, due to the codesize and the memory 1172  
 occupied, running  $PH_s$ ,  $PH_a$  and *OPES* on the same 1173  
 sensor node platform is unrealistic. Instead, we 1174  
 propose to pre-configure the nodes before their roll-out: 1175  
 half of the sensor nodes run the code for  $PH_s$  plus 1176  
 $PH_a$  and the remaining half of the sensor nodes runs 1177  
*OPES*. With the roll-out, both types of nodes are 1178  
 equally distributed over the covered region. They 1179  
 span two overlapping types of WSNs, namely 1180  
 $WSN_{PH}$  and  $WSN_{OPES}$ . The  $WSN_{PH}$  is responsible 1181  
 for a condensed and encrypted sum representation 1182  
 whereas the  $WSN_{OPES}$  does the same for the mini- 1183  
 mum or maximum. Nevertheless, both WSNs are 1184  
 not fully disjunctive since nodes from both WSN 1185

1186 forward traffic also from the other WSN. With such  
1187 a roll-out, database queries can also address com-  
1188 parison operations by applying exactly the same  
1189 database queries and database responses as intro-  
1190 duced in Section 8.

### 1191 13. Conclusion

1192 We introduced the concept of persistent  
1193 encrypted long-term data storage in asynchronous  
1194 WSNs. We distributed and stored the environmental  
1195 fingerprint of an area covered by a WSN in a con-  
1196 densed and concealed manner by applying two types  
1197 of homomorphic encryption transformations. For  
1198 encryption during aggregation we used a symmetric  
1199 privacy homomorphic encryption scheme, whereas  
1200 for a long term replicated storage of the data we  
1201 applied an asymmetric privacy homomorphism  
1202 which obviously is the better choice from the security  
1203 engineering viewpoint. To also handle database que-  
1204 ries on minimum or maximum operations we pro-  
1205 pose to run *OPES* for a fraction of the sensor  
1206 nodes and roll-out two overlaying types of networks.

### 1207 14. Uncited references

1208 6,7,23.

### 1209 Acknowledgements

1210 The authors would like to thank Satoshi Obana  
1211 and Marc Stoecklin for helpful comments. This  
1212 work is supported in part by the European Commis-  
1213 sion within the STREP UbiSecSens (<http://www.ist-ubisecsens.org>). The views and conclusions con-  
1214 tained herein are those of the authors and should  
1215 not be interpreted as necessarily representing the  
1216 official policies or endorsements, either expressed  
1217 or implied, of the UbiSecSens project or the Euro-  
1218 pean Commission.  
1219

### 1220 Appendix A

#### 1221 A.1. Homomorphic mapping function

1222 In order to utilize the additive homomorphic  
1223 property of the EC-ElGamal encryption scheme,  
1224 referred to in Section 5, we also require a corre-  
1225 sponding homomorphic mapping function. Specifi-  
1226 cally, we need the following property to hold: for  
1227 all  $a_1, a_2 \in F_p, \text{map}(a_1 + a_2) = \text{map}(a_1) + \text{map}(a_2)$ .  
1228 The homomorphic mapping function that we use

is similar to the one proposed by VoteHere in [1], 1229  
and is based upon using multiples of a generator ele- 1230  
ment to represent mapped values. The concept of 1231  
generators is familiar in finite fields and all its prop- 1232  
erties apply to elliptic curves, including the fact that 1233  
a generator point that is continuously added to itself 1234  
will enumerate all elements in the field. Our 1235  
approach is to map plaintext value  $j$  to the EC point 1236  
 $jG$ , and reverse mapping entails extracting  $j$  from  $jG$ . 1237  
This realizes our desire for a homomorphic 1238  
mapping function as the following operations hold: 1239  
for  $i, j \in F_p, (i + j)G = iG + jG$ , where  $p$  is the prime 1240  
defining the curve. The value  $i = 0$  is represented 1241  
by the point at infinity, which is the identity element 1242  
in elliptic curve groups. As can be seen, the mapping 1243  
function only involves addition of points, and can 1244  
be optimized through the use of the add-and-double 1245  
algorithm (the equivalent of square-and-multiply) 1246  
and pre-computation of points at regular intervals. 1247

#### 1248 A.2. Security Proof of $PH_s$

The homomorphic encryption scheme  $PH_s$  is 1249  
very similar to a xor-based stream cipher. Its secu- 1250  
rity can be proven using a similar proof. The secu- 1251  
rity relies in two important features: (1) the key- 1252  
stream changes from one message to another and 1253  
(2) all the operations are performed modulo an inte- 1254  
ger  $M$ . These two features protect our scheme from 1255  
frequency analysis attacks. In fact, it can be proven 1256  
that our scheme is *perfectly* secure. 1257

**Proof.** For plaintext space  $m$ , key-stream space  $K$ , 1258  
let  $K = |m|$ ,  $a \in [0, m - 1]$ ,  $c \in [0, m - 1]$ . 1259  
Set  $k^* = c - a \pmod{m}$ . Then: 1260

$$\begin{aligned} kK[Enc(k, a, m) = c] &= kK[k + a = c \pmod{m}] \\ &= kK[k = c - a \pmod{m}] \\ &= kK[k = k^*] \end{aligned} \quad 1262$$

If we assume that the maximum number of cipher- 1263  
texts to be added is  $n$  and that each plaintext is  $l$ - 1264  
bit long, we must have  $m = 2^{l + \lceil \log(n) \rceil}$ , i.e.,  $|m| =$  1265  
 $l + \lceil \log(n) \rceil$ . If  $c_i = (a_i + k_i)$ , then the probability 1266  
that  $c_i \in [0, 2^l - 1]$  is twice the probability that 1267  
 $c_i \in [2^l, m - 1]$ . More specifically, we have: 1268

$$\begin{aligned} kK[k = k^*] &= 1/(2^l + m) \text{ if } c > 2^l \text{ and } k \\ kK[k = k^*] &= 2/(2^l + m) \text{ if } c < 2^l. \end{aligned} \quad 1269$$

Since these two equations hold for every  $a \in m$ , it 1271  
follows that for every  $a_1, a_2 \in M$  we have 1272

$$kK[Enc(k, a_1, m) = c] = kK[Enc(k, a_2, m) = c] \quad 1274$$

which establishes perfect security of our scheme.  $\square$  1275

1276 **References**

- 1277 [1] J.M. Adler, W. Dai, R.L. Green, C.A. Neff, Computational  
1278 details of the votehere homomorphic election system, in:  
1279 ASIACRYPT 2000, Kyoto, Japan, December 2000.
- 1280 [2] R. Agrawal, J. Kiernan, R. Srikant, Y. Xu, Order preserving  
1281 encryption for numeric data, in: Special Interest Group on  
1282 Management of Data (ACM SIGMOD 2004), June 2004.
- 1283 [3] M. Acharya, J. Girao, D. Westhoff, Secure comparison of  
1284 encrypted data in wireless sensor networks, in: 3rd Interna-  
1285 tional Symposium on Modeling and Optimization in Mobile,  
1286 Ad Hoc, and Wireless Networks (WiOpt'05), Trentino, Italy,  
1287 April 2005.
- 1288 [4] C. Castelluccia, E. Mykletun, G. Tsudik, Efficient Aggrega-  
1289 tion of encrypted data in wireless sensor networks, in: 2nd  
1290 Annual International Conference on Mobile and Ubiquitous  
1291 Systems: Networking and Services, San Diego, CA, USA,  
1292 July 2005.
- 1293 [5] Wireless medium access control and physical layer specifica-  
1294 tions for low-rate wireless personal area networks. IEEE  
1295 Standard, 802.15.4-2003, May2003.
- 1296 [6] N.B. Chang, M. Liu, Optimal controlled flooding search in  
1297 a large wireless network, in: 3rd International Symposium  
1298 on Modeling and Optimization in Mobile, Ad Hoc, and  
1299 Wireless Networks (WiOpt'05), Trentino, Italy, April  
1300 2005.
- 1301 [7] N.B. Chang, M. Liu, Revisiting the TTL-based controlled  
1302 flooding search: optimality and randomization, in: 10th  
1303 Annual International Conference on Mobile Computing and  
1304 Networking (ACM Mobicom'04), September 2004, Phila-  
1305 delphia, PA, USA, 2004.
- 1306 [8] D. Dolev, A.C. Yao, On the security of public-key protocols,  
1307 IEEE Transactions on Information Theory 29 (2) (1983)  
1308 198–208.
- 1309 [9] J. Domingo-Ferrer, A provably secure additive and multi-  
1310 plicative privacy homomorphism, in: Information Security  
1311 Conference (ISC'02), Springer LNCS, vol. 2433, 2002, pp.  
1312 471–483.
- 1313 [10] J. Girao, D. Westhoff, M. Schneider, CDA: Concealed data  
1314 aggregation for reverse multicast traffic in wireless sensor  
1315 networks, in: IEEE International Conference on Communi-  
1316 cations (ICC'05), Seoul, Korea, May 2005.
- 1317 [11] N. Gura, A. Patel, A. Wander, H. Eberle, S.C. Shantz,  
1318 Comparing elliptic curve cryptography and RSA on 8-bit  
1319 CPUs, Cryptographic Hardware and Embedded Systems  
1320 (CHES), Cambridge, MA, USA, 2004, pp. 119–132.
- 1321 [12] W. Heinzlmann, Application-Specific Protocol Architec-  
1322 tures for Wireless Networks, PhD thesis, Massachusetts  
1323 Institute of Technology, 2000.
- 1324 [13] J.M. Hellerstein, W. Hong, S.Madden, K. Stanek, Beyond  
1325 average: Towards sophisticated sensing with queries, in:  
1326 Workshop on Information Processing in Sensor Networks  
1327 (IPSN'03), Palo Alto, California, USA, April 2003.
- 1328 [14] H. Karl, A. Willig, Protocols and Architectures for Wireless  
1329 Sensor Networks, Wiley, 2005.
- 1330 [15] S. Madden, M.J. Franklin, J. Hellerstein, W. Hong, TAG: a  
1331 Tiny aggregation service for ad-hoc sensor networks, in: 5th  
1332 Symposium on Operating Systems Design and Implementa-  
1333 tion (OSDI'02), 2002.
- 1334 [16] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, Handbook  
1335 of Applied Cryptography, Discrete Mathematics and its  
1336 Applications, CRC Press, 1997.

- 1337 [17] E. Mykletun, J. Girao, D. Westhoff, Re-visited: Public key  
1338 based homomorphic cryptoschemes for data concealment in  
1339 wireless sensor networks, in: IEEE International Conference  
1340 on Communications (ICC'06), Istanbul, Turkey, June 2006.
- 1341 [18] T. Okamoto, S. Uchiyama, A new Public-Key Cryptosystem  
1342 as Secure as Factoring, in: Advances in Cryptology –  
1343 EUROCRYPT'98, 1998, pp. 303–318.
- 1344 [19] P. Paillier, Trapdoor discrete logarithms on elliptic curves  
1345 over rings, in: Advances in Cryptology – ASIACRYPT'00,  
1346 2000, pp. 573–584.
- 1347 [20] R.L. Rivest, L. Adleman, M.L. Dertouzos, On data banks  
1348 and privacy homomorphisms, Foundations on Secure Com-  
1349 putation, Academia Press, 1978, pages 169–179.
- 1350 [21] S. Tilak, N.B. Abu-Ghazaleh, W. Heinzlmann, Collabora-  
1351 tive storage management in sensor networks, International  
1352 Journal of Ad Hoc Ubiquitous computing (2005).
- 1353 [22] N. Sastry, D. Wagner, Security considerations for IEEE  
1354 802.15.4 networks, in: ACM Workshop on Wireless Security  
1355 (WiSe 2004), co-located with MobiCom 2004 Conference,  
1356 pp. 32–42.
- 1357 [23] D. Wagner, Cryptanalysis of an algebraic privacy homomor-  
1358 phism (revised version), in: Proceedings of the 6th Informa-  
1359 tion Security Conference (ISC'03), Bristol, UK, October 2003.
- 1360 [24] D. Westhoff, J. Girao, M. Acharya, Concealed Data  
1361 Aggregation for Reverse Multicast Traffic in Wireless Sensor  
1362 Networks: Encryption, Key Pre-distribution and Routing, to  
1363 appear as regular paper in IEEE Transactions on Mobile  
1364 Computing, August 2006.
- 1365 [25] X. Zeng, R. Bagrodia, M. Gerla, GloMoSim: a library for  
1366 parallel simulation of large-scale wireless networks, in:  
1367 Proceedings of the 12th Workshop on Parallel and Distrib-  
1368 uted Simulation, Banff, Alberta, CA, May 1998.
- 1369



ad-hoc networks. He is a student member of both the IEEE and ACM.



Dirk Westhoff holds a PhD from the Distance University of Hagen, Germany. He is a Chief Researcher at NEC Europe Ltd. R&D Network Laboratories in Heidelberg, Germany. He is co-founder of the ESAS (European Workshop on Security in Ad Hoc and Sensor Networks) Workshop series published by Springer. He is involved in the TPC of several ACM and IEEE conferences. His research interests include wireless security, ad hoc and sensor network security, and many other security aspects of distributed mobile communication.

1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400

1401

1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414

**Einar Mykletun** is currently a PhD candidate in the School of Information and Computer Science at the University of California, Irvine. His current research interest is focused on providing security in outsourced databases, including the use of secure hardware and techniques to allow an untrusted database server to run SQL queries over encrypted data. This work was done while he was an intern at NEC Europe Network Laboratories.



**Toshinori Araki** received B.S. and M.E. degrees in computer science in 2003 and 2005 from Tokyo Institute of Technology. He joined Internet Systems Research Laboratories, NEC in 2005. His research interests are cryptography and information security.

1419  
1420  
1421  
1422  
1423  
1424  
14251403  
14161418  
1426

UNCORRECTED PROOF