Pervasive and Mobile Computing 7 (2011) 495–508

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Intrusion-resilient integrity in data-centric unattended WSNs

Roberto Di Pietro [a], Claudio Soriente [b], Angelo Spognardi [c,*], Gene Tsudik [d]

[a] UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Spain

[b] DLSIIS, Universidad Politécnica de Madrid, Spain

[c] Dipartimento di Informatica, Università di Roma "La Sapienza", Italy

[d] Computer Science Department, University of California Irvine, United States

ARTICLE INFO

Article history:
Received 17 May 2010
Accepted 24 December 2010
Available online 31 December 2010

Keywords:
UWSN
Data integrity
Wireless sensor network security

ABSTRACT

Unattended Wireless Sensor Networks (UWSNs) operate in autonomous or disconnected mode: sensed data is collected periodically by an itinerant sink. Between successive sink visits, sensor-collected data is subject to some unique vulnerabilities. In particular, while the network is unattended, a mobile adversary (capable of subverting up to a fraction of sensors at a time) can migrate between compromised sets of sensors and inject fraudulent data.

In this paper, we provide two collaborative authentication techniques that allow an UWSN to maintain integrity and authenticity of sensor data – in the presence of a mobile adversary – until the next sink visit. Proposed schemes use simple, standard, and inexpensive symmetric cryptographic primitives, coupled with key evolution and few message exchanges. We study their security and effectiveness, both analytically and via simulations. We also assess their robustness and show how to achieve the desired trade-off between performance and security.

1. Introduction

Unattended Wireless Sensor Networks (UWSNs) are an emerging type of sensor network characterized by mostly autonomous operations. Unlike more traditional WSN settings, which assume constant supervision by a sink and real-time data collection, a UWSN is only periodically visited by a sink. Since UWSN nodes are forced to store sensed data locally until the next sink visit, UWSNs become susceptible to a wider range of attacks than common WSNs. In particular, between successive sink visits a *mobile adversary* (μADV) can compromise sensors, manipulate stored data, and abandon the network without leaving a trace. In the past few years, several research results [1–7] have investigated security threats unique to UWSNs, identified several types of adversarial behavior (varying in its goals) and suggested mitigating techniques.

In this paper, we focus on the integrity of sensed data in UWSNs by considering the type of adversary that aims to replace authentic data obtained by sensors with its own chosen values. Within this framework, we make several **contributions**. First, we overview current solutions for the authentication problem in UWSNs and show that they only offer weak protection against a mobile adversary. Second, we construct two techniques – based on sensor cooperation – that provide much better security with just small increase in overhead. Both techniques are reasonably practical for energy-constrained sensor networks since they only rely on symmetric cryptography and involve few message exchanges. We also analyze proposed techniques in terms of robustness (i.e., dealing with issues such as message loss and sensor failures). Finally, we propose

[*] Corresponding author.
E-mail addresses: roberto.dipietro@urv.cat (R. Di Pietro), csoriente@fi.upm.es (C. Soriente), spognardi@di.uniroma1.it (A. Spognardi), gts@ics.uci.edu (G. Tsudik).

1574-1192/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.pmcj.2010.12.003

several extensions that reduce energy consumption without weakening security. All results are backed by thorough analysis and extensive simulations.

The rest of the paper is organized as follows. Section 2 overviews related work while Section 3 introduces the network assumptions and the adversarial model. Section 4 shows that traditional authentication techniques fail to provide sufficient security in UWSNs. Two new cooperative schemes are introduced in Section 5, security analysis is provided in Section 6 and simulation results and their discussion are presented in Section 7. Section 8 shows how parameters can be tuned to reach the desired level of security. Section 9 shows how to extend the proposed protocols to decrease overhead, while energy consumption analysis is provided in Section 10. Section 11 considers authentication of sensed data when the whole network has been compromised and Section 12 ends the paper with some concluding remarks.

2. Related work

Data authentication in WSNs has been the subject of many research papers, however they assume that the network is constantly attended by the sink. Several such schemes, such as [8,9], have been proposed for efficient sink-to-sensor broadcast authentication. In contrast, we focus on intermittent sensor-to-sink data authentication in the envisaged UWSN setting and in the presence of a mobile adversary.

In other prior work, it is assumed that sensors can detect false/fake data via collaborative mechanisms for value endorsement [10–14]. Assuming a suitable key pre-distribution scheme,[1] sensors verify data collected by peers, in order to detect false measurements and avoid their transmission to the sink.

An authentication framework for hierarchical sensor networks is presented in [15]. It leverages network heterogeneity to relegate expensive crypto-operations to the most powerful sensors.

As mentioned in Section 1, some recent results considered UWSNs but focused on different problems and mobile adversary sub-types, such as data survival [3,1,2,4] and self-healing [5,6] for intrusion-tolerant secrecy.

We note that data authentication for unattended sensors was first investigated in [16], where forward-secure aggregate authentication techniques were proposed that optimize storage requirements. However, such measures are only effective against a relatively weak adversary, referred to as *reactive* in [4]. Moreover, [16] does not assume a UWSN setting but, rather, a collection of non-communicating unattended sensors.

A recent result showing some overlap with our problem has been presented in [17]: a mechanism based on aggregate signature schemes and Time-Release Encryption is used to achieve data authentication in UWSN. The proposal exploits both Symmetric and Elliptic Curve Cryptography to realize two complex and energy demanding schemes that are provably secure under appropriate computational assumptions. In contrast, our proposal is much simpler and relies only on traditional cryptography.

Techniques proposed in this paper share some features with the cooperative key-healing approaches detailed in [5,6]. In both [5] and [6], sensors regain security after compromise using randomness provided by non-compromised peers. Secrets derived from combining peer contributions with prior state, are used either as keys (with symmetric encryption) or as randomizers (with public key encryption). However, symmetric key techniques are very fragile since they assume no lost messages and no sensor failures. In contrast, solutions explored in this paper rely on purely symmetric techniques and offer robustness in spite of imperfect communication and potential sensor failures. Also, our techniques achieve higher security than [5,6], while incurring the same bandwidth overhead.

Security threats in a WSN where data are collected by a mobile sink were studied in [18–20]. One result that also explores the unattended setting is the pDCS system [21] geared for Data-Centric Sensor Networks, where the goal is maintaining privacy of sensor-collected data. The concept of *parasitic* adversaries introduced in Gossicrypt [22] is close to our notion of the mobile adversary, however, Gossicrypt relies on the sink's constant presence.

3. System model

This section describes our assumptions about the network and the adversary. Notation is summarized in Table 1.

3.1. Network assumptions

We assume a homogeneous UWSN composed of a set of sensors $\{s_1, s_2, \dots, s_n\}$ uniformly distributed over a certain geographical area. Time is divided into collection rounds and each sensor is programmed to perform per-round sensing. The UWSN is always connected and any two sensors can communicate either directly or through peers, according to some underlying routing protocol (this assumption will be relaxed later in the paper). There is a system-wide parameter – v – denoting the maximum number of data collection rounds between successive sink visits. Each sensor s_j has sufficient storage to accommodate $O(v)$ data items. Each s_j also shares a unique secret key and a unique seed with the sink: the former

[1] A research area in its own right.

Table 1
Notation.

μADV	The mobile adversary
n	Size of the UWSN
v	Max. # of rounds between successive sink visits
C_r	Set of compromised sensors at round r
k	μADV's compromise power — maximum size of C_r
$s_1 \dots s_n$	Sensors
s_i	Target sensor
$\bar{r}$	Target data collection round
d_j^r	Data collected by s_j in round r
K_j^r	Key used by s_j at round r
z_j^r	MAC computed by s_j at round r using K_j^r
t	Numbers of co-authenticators

is used to compute message authentication codes (MACs), while the latter initializes a Pseudo-Random Number Generator (PRNG). At each visit, the sink securely refreshes all keys and secret seed values for all sensors and resets the round counter.

The envisaged adversary (μADV) strives to substitute a value (target data) obtained by s_i (target sensor) at round $\bar{r}$ (target data collection round) with a selected value — not relevant to the extent of this paper. Any sensor and any round are equally likely to be chosen as μADV's targets. We assume that μADV learns the identity of s_i at the end of round $\bar{r}$ and, from that moment on, it has $v - \bar{r}$ rounds to accomplish its goal. μADV succeeds if, at the end of round v, the sink visits the network and accepts the value injected by μADV as the genuine measurement by s_i at round $\bar{r}$. Sensors are neither aware of the identity of the target sensor nor of the target data collection round. Thus, each data item must be equally protected.

From μADV's perspective, time is divided into equal and fixed *compromise rounds*. For ease of illustration, and without loss of generality, we assume that the compromise and collection rounds are of the same duration and are synchronized, that is both types of round start and end at the same time. μADV can compromise up to a fixed number k $(k < n)$ of sensors at any round. At the beginning of each round, μADV selects the set of sensors to compromise (C_r) and migrates to them in one atomic movement. For each newly-compromised sensor, μADV learns all keys, reads all storage/memory, and eavesdrops on all incoming and outgoing communication. Finally, we assume that μADV does not interfere with sensor behaviors, in order to remain undetected. In particular, it does not delete or delay messages and does not introduce spurious messages. μADV's only goal is to find and replace the target data.

Since, μADV begins compromising sensors right after the sink leaves the network, it could be that $s_i \in C_{\bar{r}}$, i.e., μADV might have compromised the target sensor at target data collection round. This case is not interesting as μADV wins by dumb luck. We focus on a more interesting and more likely scenario where $s_i \notin C_{\bar{r}}$ and possibly $s_i \in C_r,\ r < \bar{r}$. In other words, μADV does not compromise s_i while target data is being collected. However, it might have compromised (and then released) that sensor at some earlier round.

4. Basic techniques

Since μADV is focused on replacing (modifying or forging) data, authentication is a natural defence. It can be attained via either symmetric MACs or public key signature. Although it is normal to expect the latter to offer better security, we show that – surprisingly – this is not true in UWSNs. To demonstrate this, we now outline two intuitive authentication techniques.

With MACs, each sensor s_j has a unique authentication key K_j shared with the sink. At round r, s_j collects d_j^r and computes $z_j^r = MAC(K_j, d_j^r)$, where $MAC(\cdot)$ can be the standard HMAC [23] based on, say, SHA-2 [24]. During its next visit, the sink collects all tuples $\{r, d_j^r, z_j^r\}_{1 \le r \le v}$ and proceeds to verify their authenticity with K_j.

With public key signatures, each sensor s_j is initialized with a key-pair $\{SK_j, PK_j\}$. SK_j is used to sign collected data, while PK_j is used by anyone (e.g., the sink) for signature verification. At round r, s_j collects d_j^r and computes σ_j^r as the signature on d_j^r under SK_j. Whenever the sink collects tuples $\{r, d_j^r, \sigma_j^r\}_{1 \le r \le v}$, it verifies each σ_j^r with PK_j.

Unfortunately, neither approach is effective against μADV: right after learning the identity of s_i, μADV compromises it and learns K_i (in case of MACs) or SK_i (in case of public key signatures). Knowledge of either key allows μADV to produce valid MACs or signatures on behalf of s_i, for any round.

Protection against such a powerful μADV can only be achieved if the underlying authentication scheme provides both *forward and backward* secrecy.[2] The former is easy to achieve with simple per-round key evolution. Backward secrecy, on the other hand, is much harder to obtain. We note that key-insulated schemes [25–28] are unsuitable for our setting. The main reason is that, in such schemes, at least one of the inputs for the key evolution function must come from a separate (un-compromisable) entity, such as a remote trusted third party or a local tamper-resistant hardware device. In the envisaged UWSN setting, neither per-sensor secure hardware nor a constantly present trusted third party is realistic. Thus, we are constrained to pursue authentication schemes that emulate key insulation, leveraging whatever meager resources are available in the considered UWSN scenario.

[2] Sometimes the combination of forward and backward secrecy is referred to as 'key insulation'.

5. Cooperative mechanisms

The main shortcoming of the aforementioned intuitive approaches is that the authentication tag of a given data item depends on the current key of a single sensor. Once that sensor is compromised, its key becomes exposed and data can be easily replaced. Involving several sensors in the authentication process for a single data item can lead to more secure schemes. In the following, we construct and evaluate two simple schemes that enhance UWSN security using so-called *co-authenticators*. We say that s_q is a *co-authenticator* of s_j if s_j requires s_q's cooperation to authenticate its data. *Security* is assessed as the probability that target data cannot be replaced along with a valid tag. Since public key signatures do not offer any particular advantage (Section 4) and are computationally expensive, we only consider schemes based on symmetric cryptography.

5.1. CoMAC

The main idea in this scheme (called CoMAC — Collaborative MAC) is that each sensor obtains help from a set of randomly chosen peers to authenticate its data, while storing the resulting authentication tags.

At round $r = 1$, s_j has an initial key K_j^1 shared with the sink. At the end of each round r, s_j evolves its current key K_j^r via a cryptographically suitable hash function $F(\cdot)$ to compute the next round's key. At each round, s_j runs Algorithms 1 and 2. The former computes (and stores) a MAC computed over the sensed data. It also sends sensed data to t randomly chosen co-authenticators. Algorithm 2 receives data from peers, computes/stores MACs, and computes the next round key.

Algorithm 1 CoMAC: *MAIN*

```
Sense $d_j^r$
Compute $z_j^r = MAC(K_j^r, d_j^r)$
Store $\{r, d_j^r, z_j^r\}$
Set $S_j^r = SELECT\_DISTINCT(t, n, j)$
for $p = 1 \ldots t$ do
    Send $d_j^r$ to $s_{S_j^r[p]}$
end for
```

Algorithm 2 CoMAC: *RECEIVE*

```
Set $R_j^r = \emptyset$
$c = 1$
while round not over do
    Receive $d_p^r$
    Set $R_j^r[c] = p$
    Compute $MAC(K_j^r, d_p^r)$
    Store $\{r, s_p, MAC(K_j^r, d_p^r)\}$
    Set $c = c + 1$
end while
Store $\{r, R_j^r\}$
Compute $K_j^{r+1} = F(K_j^r || R_j^r[1] || \ldots || R_j^r[|R_j^r|])$
```

The function *SELECT_DISTINCT* (t, n, j) returns an array of t distinct elements randomly chosen from the set $\{1, \ldots, n\} \backslash \{j\}$. Its output is based on the results of a Pseudo-Random Number Generator (PRNG) initialized with a secret seed chosen by the sink. Let S_j^r denote this array for sensor s_j at round r.

During its next visit, the sink acquires collected data along with corresponding MACs. For each d_j^r, it verifies z_j^r with K_j^r and then repeats the process for each co-authenticator $s_q \in S_j^r$. If all $t+1$ MACs are successfully verified, the sink authenticates d_j^r.

Note that, since the sink knows all PRNG seeds, it can compute S_j^r for any s_j and for any round r. Similarly, the sink can re-compute any K_j^{r+1} from K_j^r as well as the identities of all sensors that s_j received a message from, at round r (i.e., R_j^r).

In CoMAC each MAC authenticates a single data item. As we discuss below, security can be improved by bundling batches of MACs computed over different data items from multiple sensors.

5.2. Extensive cooperation (ExCo)

In ExCo each s_j sends the MAC computed over its own data to t randomly chosen co-authenticator peers. Each sensor bundles its MAC and all MACs received from other sensors into a single authentication tag. At round r, s_j runs Algorithms 3 and 4.

Algorithm 3 ExCo: *MAIN*

```
Sense $d_j^r$
Compute $z_j^r = MAC(K_j^r, d_j^r)$
Set $S_j^r = SELECT\_DISTINCT(t, n, j)$
for $c = 1 \ldots t$ do
    Send $z_j^r$ to $s_{S_j^r[c]}$
end for
```

Algorithm 4 ExCo: *RECEIVE*

```
Set $R_j^r = \emptyset$
Set $\Lambda_j^r = \emptyset$
$c = 1$
while round is not over do
    Receive $z_p^r$
    Set $R_j^r[c] = p$
    Set $\Lambda_j^r[c] = z_p^r$
    Set $c = c + 1$
end while
Compute $H_j^r = MAC(z_j^r || \Lambda_j^r[1] || \ldots || \Lambda_j^r[|R_j^r|])$
Compute $K_j^{r+1} = F(K_j^r || R_j^r[1] || \ldots || R_j^r[|R_j^r|])$
Store $\{r, d_j^r, H_j^r, R_j^r\}$
```

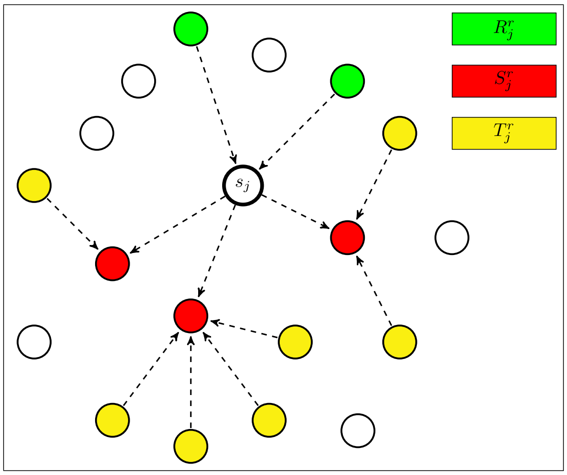


Fig. 1. ExCo. Colored sensors are the one involved in the authentication of data collected by s_j at round r.

The former algorithm computes the MAC of current data and sends it to t randomly chosen peers. The latter algorithm receives MACs from peers, aggregates them in a single authentication tag, and computes the next round key.

Data verification in ExCo is similar to CoMAC. Once the sink acquires d_j^r, it verifies all authentication tags involving that value. In particular, to authenticate d_j^r, the sink must verify H_j^r and $\{H_q^r\}_{s_q \in S_j^r}$. Verification of H_j^r requires K_j^r and $\{K_p^r, d_p^r\}_{s_p \in R_j^r}$. For any $s_q \in S_j^r$, verification of H_q^r requires $\{K_q^r, d_q^r\}$ and $\{K_p^r, d_p^r\}_{s_p \in T_j^r}$ where $T_j^r = \{\bigcup_{s_p \in S_j^r} R_p^r\}$. As an example, Fig. 1 shows a network where colored sensors are the ones involved in the authentication of a value sensed by s_j at round r. An arrow between s_j and s_q means that $s_q \in S_j^r$. In order to verify d_j^r, the sink needs K_j^r as well as key and sensed data at round r from each sensor in $S_j^r \cup R_j^r \cup T_j^r$.

6. Security analysis

In this section we analyze μADV's ability to replace target data and its authentication tag(s). We start observing that the latter is proportional to $\frac{|\mathcal{K}^r|}{|\mathbb{K}^r|}$, where $\mathbb{K}^r = \{K_1^r, \ldots, K_n^r\}$ and $\mathcal{K}^r$ is the subset of keys in $\mathbb{K}^r$ that are known to μADV.

We take a conservative view and assume that μADV is proactive, roaming the network before learning $\bar{r}$ and hoping to discover as many keys as possible. Moreover, we assume that in order to maximize the total number of collected keys (by

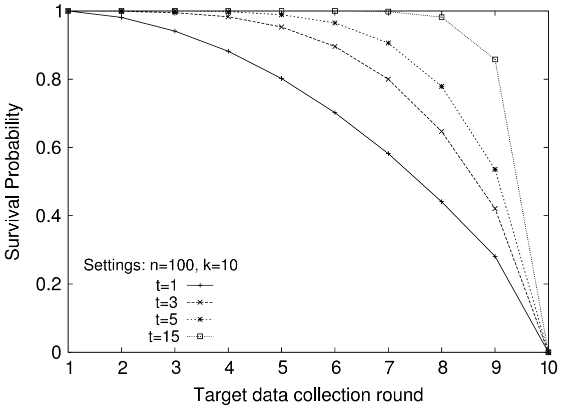


Fig. 2. CoMAC. Survival probability of the target data.

round $\bar{r}$), μADV compromises sensors in a round-robin fashion, choosing C_r such that $C_r \cap \{C_1 \cup \cdots \cup C_{r-1}\} = \emptyset$. With this strategy, μADV can 'visit' all sensors by round $\lceil \frac{n}{k} \rceil$ and, from that point on, it can replace any data subsequently collected by any sensor.

6.1. CoMAC

To win the 'game' against CoMAC, μADV must replace authentication tags generated by s_i and $S_i^{\bar{r}}$ at round $\bar{r}$. For this to happen, s_i must have been compromised by round $\bar{r} - 1$ (that is, $s_i \in \{C_1 \cup \cdots \cup C_{\bar{r}-1}\}$) and sensors in $S_i^{\bar{r}}$ must have been compromised by round $\bar{r}$ (that is, $S_i^{\bar{r}} \in \{C_1 \cup \cdots \cup C_{\bar{r}}\}$). The probability of this event can be expressed as:

$$P_{\mu\text{ADV}}(n, k, \bar{r}, t) = \begin{cases} \dfrac{k(\bar{r} - 1)}{n} \cdot \displaystyle\prod_{i=1}^{t} \dfrac{k\bar{r} - i}{n - i} & \text{if } k\bar{r} < n \\ 1 & \text{otherwise.} \end{cases}$$

The first term on the left accounts for the probability that s_i is compromised before round $\bar{r}$, while the other terms represent the probability that sensors in $S_i^{\bar{r}}$ are compromised up to round $\bar{r}$. Consequently, the security of the scheme is:

$$P_{\text{CoMAC}}(n, k, \bar{r}, t) = 1 - P_{\mu\text{ADV}}(n, k, \bar{r}, t). \tag{1}$$

Fig. 2 plots P_{CoMAC} in a 100-node UWSN, where μADV can compromise $k = 10$ sensors per round. Even if target data collection occurs after μADV compromises a substantial portion of the network, the probability of target data non-forgery is rather high, even for small values of t.

6.2. ExCo

With ExCo, μADV must replace (that is, must be able to recompute) all the authentication tags generated using $d_i^{\bar{r}}$. In particular, μADV must replace $H_i^{\bar{r}}$ as well as $H_j^{\bar{r}}$ for each $s_j \in S_i^{\bar{r}}$.

For each of the above authentication tags, μADV must know $K_i^{\bar{r}}$. Computation of $H_i^{\bar{r}}$ requires also $(K_p^{\bar{r}}, d_p^{\bar{r}})$ for each $s_p \in R_i^{\bar{r}}$. Furthermore, for each $s_j \in S_i^{\bar{r}}$, computation of $H_j^{\bar{r}}$ requires knowledge of both $(K_j^{\bar{r}}, d_j^{\bar{r}})$ and $\{(K_p^{\bar{r}}, d_p^{\bar{r}})\}_{s_p \in T_i^{\bar{r}}}$. To summarize, μADV must have compromised s_i at any round $r < \bar{r}$ and sensors in $S_i^{\bar{r}} \cup R_i^{\bar{r}} \cup T_i^{\bar{r}}$ at any round $r \leq \bar{r}$, as shown in Fig. 1.

To assess the effort needed for μADV to succeed, we estimate the probability of any sensor being in at least one of the sets $R_i^{\bar{r}}, S_i^{\bar{r}}, T_i^{\bar{r}}$. Let $\mathcal{R}$ (respectively $\mathcal{S}$, $\mathcal{T}$) denote the event that a sensor is in $R_i^{\bar{r}}$ (respectively, $S_i^{\bar{r}}$, $T_i^{\bar{r}}$) and let $\overline{\mathcal{R}}$ (respectively $\overline{\mathcal{S}}$, $\overline{\mathcal{T}}$) denote the event that a sensor is NOT in the set $R_i^{\bar{r}}$ ($S_i^{\bar{r}}$, $T_i^{\bar{r}}$). We are interested in $P[\overline{\mathcal{T}} \wedge \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] = P[\overline{\mathcal{T}} | \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] \cdot P[\overline{\mathcal{R}} \wedge \overline{\mathcal{S}}]$.

Note that $P[\overline{\mathcal{T}} | \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] = P[\overline{\mathcal{T}} | \overline{\mathcal{R}}]$, since belonging to set $T_i^{\bar{r}}$ is completely independent from belonging to set $S_i^{\bar{r}}$. Indeed, sensor $s_j \in S_i^{\bar{r}}$ because it has been chosen as a co-authenticator by s_i, but $s_j \in T_i^{\bar{r}}$ because itself chose as a co-authenticator another sensor in $S_i^{\bar{r}}$. $P[\overline{\mathcal{T}} | \overline{\mathcal{R}}]$ can be evaluated as follows:

$$P[\overline{\mathcal{T}} | \overline{\mathcal{R}}] = \left(1 - \frac{t}{n-2}\right) \cdots \left(1 - \frac{t}{n - 2 - (t-1)}\right) = \prod_{j=0}^{t-1} \left(1 - \frac{t}{n-2-j}\right) \geq \left(1 - \frac{t}{n-2}\right)^t. \tag{2}$$

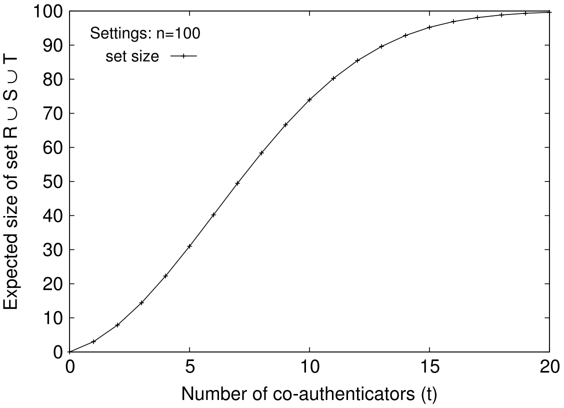


Fig. 3. ExCo. Expected size of $R \cup S \cup T$.

The last relation holds since sensor $s_j \notin T_i^{\bar{r}}$ only if s_j sent all its t MACs to sensors not belonging to $S_i^{\bar{r}}$ (namely $n - t$ sensors) *and* sensor s_j did not send any MAC to s_i (since $s_j \notin R_i^{\bar{r}}$). As s_j cannot choose itself as a recipient for a MAC, we have that $s_j \notin T_i^{\bar{r}}$ because it has chosen t times from the same set with $n - t - 2$ sensors, always excluding s_i, s_j and any sensor in $S_i^{\bar{r}}$.

Combining $P[\overline{\mathcal{T}}|\overline{\mathcal{R}}]$ with the fact that $P[\mathcal{R}] = P[\mathcal{S}] = \frac{t}{n-1}$, we can estimate $\nu(n, t)$, the number of sensors involved in the computation of the authentication tags of generic data d_j^r.

$$\nu(n, t) = n\left(1 - P[\overline{\mathcal{T}} \wedge \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}]\right) \geq n\left(1 - \left(1 - \frac{t}{n-1}\right)^2 \left(1 - \frac{t}{n-2}\right)^t\right) \geq n\left(1 - \left(1 - \frac{t}{n-1}\right)^{t+2}\right). \tag{3}$$

Fig. 3 reports the expected size of the set $R_j^r \cup S_j^r \cup T_j^r$, obtained via Eq. (3), fixing n to 100 and varying the number of co-authenticators t. For any t, ExCo achieves better security than CoMAC because each data is authenticated using secret material from a larger number of sensors ($\nu(n, t) + 1$ against $t + 1$) — without affecting communication overhead.

These results are also supported by our simulations as shown in next section (see Fig. 5).

7. Simulation and discussion

In this section we present simulation results and discuss the robustness of the proposed schemes.

7.1. Simulation results

Simulations were run with the OMNET++ Discrete Event Simulator System [29,30]. We set $n = 100$ and $k = 10$; for each scenario, μADV is active for $\bar{r}$ rounds and corrupts k sensors at each round. Each sensor acquires one data item per round and authenticates it according to the selected protocol. After $\bar{r}$ rounds, we randomly select target data among all data items collected during the current round and check if μADV has all the required keys used to produce the authentication tags. Experiments were repeated 1000 times for each configuration of t and $\bar{r}$; we report the resulting averages.

Fig. 4 reports the size of the set $R_j^r \cup S_j^r \cup T_j^r$ for $t = 3$, varying the size of the network from 100 to 1000: it reports the ratio between the number of times the set resulted in a certain size and the total number of trials. The expected size was 14.287 and 14.906 for $n = 100$ and $n = 1000$, respectively, while plugging the same parameters in Eq. (3) returns 14.42 and 14.94, respectively.

Figs. 5 and 6 compare the effectiveness of CoMAC and ExCo, respectively, showing the survival probability of target data, as described in Section 6. In Fig. 5 the x axis is the number of compromise rounds before $\bar{r}$, while the y axis is the expected probability of non-forgery of the target data. Since we assume $s_i \in C^{\bar{r}}$, if $\bar{r} = 1$ μADV has zero probability of success.[3] However, if target data is sensed at round $\lceil \frac{n}{k} \rceil$ or after, μADV succeeds with certainty, since enough rounds have passed to allow it to compromise the whole network. CoMAC and ExCo aim at increasing security of target data collected at any round between 1 and $\lceil \frac{n}{k} \rceil$.

Fig. 6 compares the two schemes with 3*D* plots of the expected probability of target data non-forgery: the x axis represents the number of co-authenticators (t) and the y axis is the number of compromise rounds before the target data collection round. ExCo results are consistently better than those of CoMAC, for the same configuration of t and $\bar{r}$. For example, given

[3] Relaxing this assumption, μADV would win with probability $\frac{k}{n}$.

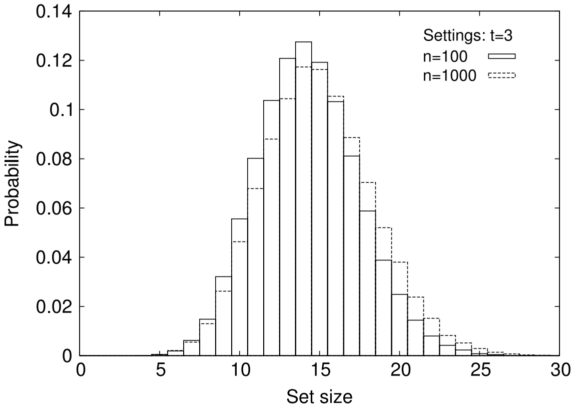


Fig. 4. ExCo. Distribution of the size of $R \cup S \cup T$.

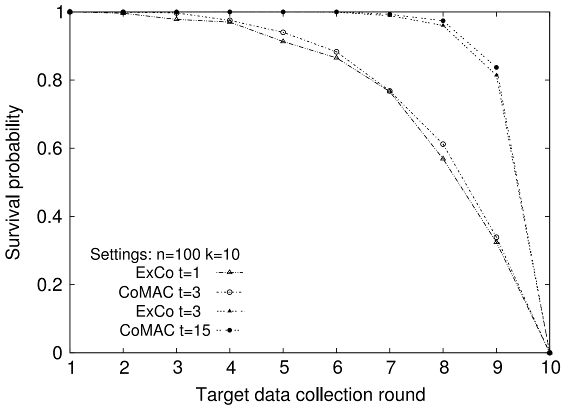


Fig. 5. Comparison between CoMAC and ExCo.

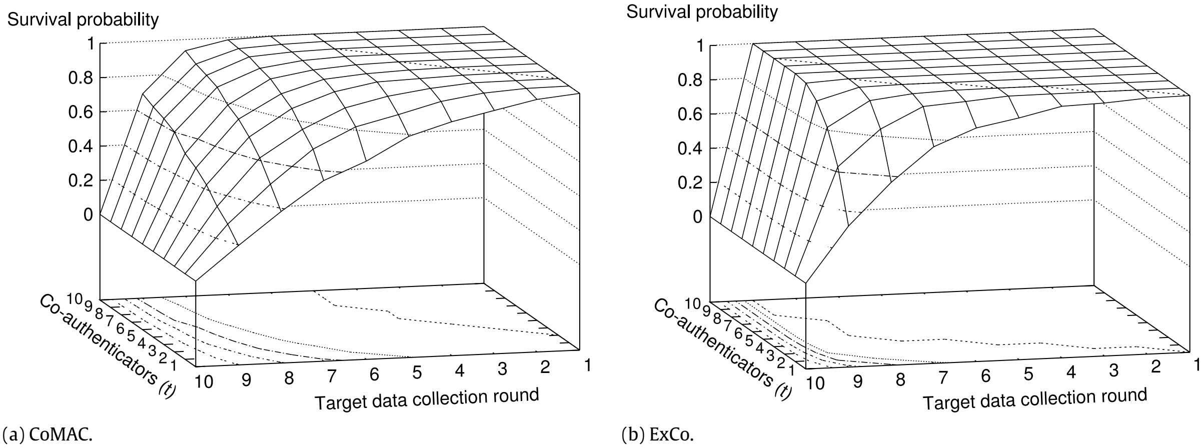


Fig. 6. Comparison between CoMAC and ExCo.

$t = 3$, probability of target data non-forgery in ExCo is 0.8, even if μADV has compromised 90% of the sensors. Given the same ratio of compromised sensors, CoMAC only reaches 0.3 probability. Even if μADV compromises 90% of the network before target data is collected, ExCo with $t \geq 4$, yields the probability of target data non-forgery exceeding 0.92, while CoMAC requires $t = 20$ to reach the same probability.

7.2. Robustness

The analysis in the previous sections is valid for an *ideal* UWSN where communication is loss-less and sensors do not fail. In this section, we analyze the proposed protocols in a more *realistic* setting, where messages can be lost (i.e. not delivered) with probability λ and sensors can fail with probability ϕ.

Message delivery failure: recall that messages are authenticated $t+1$ times with CoMAC and, on the average, $\nu(n,t)+1$ times with ExCo. With unreliable message delivery, the number of sensors authenticating d_j^r would drop by a factor of λ in both schemes.

Sensor failure: if a sensor fails, its resident data and all MACs computed over its own and other sensors' data are lost. Assume that s_q fails at round r. With CoMAC, for each round $r' < r$, each $d_j^{r'}$ such that $s_q \in S_j^{r'}$ loses one MAC. Also, all data sensed by s_q is lost and their MACs can be discarded. With ExCo, one authentication tag for $d_p^{r'}$ sensed by $s_p \in \{S_q^{r'} \cup R_q^{r'} \cup T_q^{r'}\}$ will not be verifiable. Comparing ExCo and CoMAC, the higher security achieved by ExCo is traded off against lower robustness.

If messages are sometimes lost and sensors fail, the sink might find MACs that lack corresponding data, or data with fewer MACs than expected ($t+1$ or $\nu(n,t)+1$ for CoMAC and ExCo, respectively). In the former case, MACs need to be discarded. If sensors become aware that a peer failed,[4] they can discard all MACs computed over data collected by the failed sensor. In the second case, data authentication is still viable, but in a weakened form. We consider the number of MACs collected by the sink as a natural degree of confidence regarding the authenticity of data: the more MACs, the greater the effort μADV needs to make to forge authentication tag(s) of target data.

8. Dynamic number of co-authenticators

μADV's probability of success is proportional to $\frac{\mathcal{K}^r}{\mathbb{K}^r}$. If μADV moves in a round-robin fashion, the latter probability increases with the round counter. That is, sensors are required to spend more *effort* – cooperate with more co-authenticators – to protect data collected in later rounds, compared to the effort necessary to protect data collected right after the sink has left the network.

To keep a fixed survival probability as rounds go by, it is possible to set the parameter t as a function of r. The sink would regard d_j^r as *genuine* only if the expected number of collected tags computed on d_j^r are all valid *and* the survival probability (as computed in Section 6) is not smaller than the system (tunable) parameter α.

8.1. Dynamic CoMAC

For Eq. (1), $P_{\text{CoMAC}}(n,k,r,t) \geq \alpha$ is equivalent to $P_{\mu\text{ADV}}(n,k,r,t) \leq 1-\alpha$. To find the smallest t that satisfies the latter inequality, notice that:

$$P_{\mu\text{ADV}}(n,k,r,t) = \frac{k(\bar{r}-1)}{n} \cdot \prod_{i=1}^{t} \frac{k\bar{r}-i}{n-i} \leq \frac{k(\bar{r}-1)}{n} \cdot \left(\frac{k\bar{r}-1}{n-1}\right)^t \leq 1-\alpha.$$

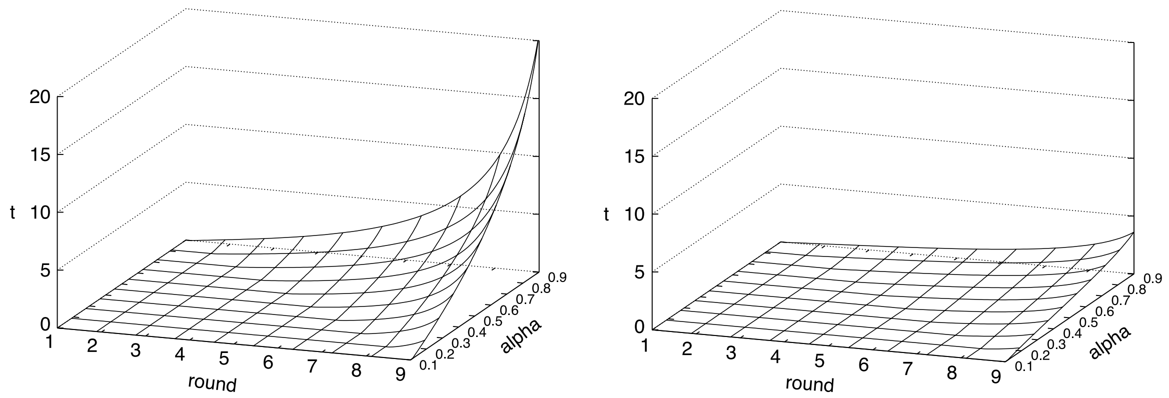
Thus the smallest t that satisfies the above inequality is given by:

$$t = \begin{cases} \left\lceil \log_{\frac{k\bar{r}-1}{n-1}} \left(\frac{(1-\alpha)n}{k\bar{r}-1}\right) \right\rceil & \text{if } k\bar{r} < n \\ \text{undefined} & \text{otherwise.} \end{cases} \tag{4}$$

Note that if $k\bar{r} \geq n\mu$ADV controls the whole network and sensor cooperation is just useless.

Fig. 7(a) plots t against $\bar{r}$ and α. If α is greater than 0.8 and target data is collected when μADV has compromised a large portion of the sensors, the network incurs a sensible communication overhead. The number of co-authenticators can be also tuned in order to guarantee that a minimal number of authentication tags per data will be collected by the sink, despite message loss and sensor failure. Let λ and ϕ be the probability that a message is lost (not delivered) and that a sensor fails, respectively. With CoMAC, s_j at round r has to choose t such that $t = \left\lceil \frac{\log_{\frac{k\bar{r}-1}{n-1}}\left(\frac{(1-\alpha)n}{k\bar{r}-1}\right)}{1-\lambda-\phi} \right\rceil$ if $k\bar{r} < n$.

[4] This mechanism is out of the scope of this paper. However, note that this information could be inferred, for instance from the routing protocol [31].



(a) CoMAC.

(b) ExCo.

Fig. 7. Tuning t to have a fixed survival probability.

8.2. Dynamic ExCo

Recall that in ExCo, the number of sensors involved in the authentication of a data item is given by Eq. (3). At round r, s_j will choose t such that $\nu(n,t) \geq \lceil \log_{\frac{k\bar{r}-1}{n-1}}(\frac{(1-\alpha)n}{k\bar{r}-1})\rceil$. First we use Eq. (4) to compute the number of co-authenticators required to reach the desired survival probability; let this value be τ. Then we solve the following equation for t:

$$n\left(1-\left(1-\frac{t}{n-1}\right)^{t+2}\right) = \left\lceil \log_{\frac{k\bar{r}-1}{n-1}}\left(\frac{(1-\alpha)n}{k\bar{r}-1}\right)\right\rceil$$

$$\Rightarrow \ln\left(1-\frac{t}{n-1}\right)\cdot(t+2) = \ln\left(1-\frac{\tau}{n}\right) \Rightarrow -\frac{t}{n-1}\cdot(t+2) \approx -\frac{\tau}{n}$$

$$\Rightarrow t \approx \sqrt{1+\tau\cdot\frac{n-1}{n}} - 1. \tag{5}$$

If we take into account λ and ϕ, s_j will chose t such that $\nu(n,t) \geq \left\lceil \frac{\log_{\frac{k\bar{r}-1}{n-1}}\left(\frac{(1-\alpha)n}{k\bar{r}-1}\right)}{1-\lambda-\phi}\right\rceil$.

Fig. 7(b) plots t against $\bar{r}$ and α. The number of s_j's co-authenticators at any round is almost constant, despite α and the sensors compromised thus far.

9. TTL-based co-authentication

Both CoMAC and ExCo require each sensor to communicate with t peers at each round. This means $n \cdot t$ new messages and, on average, $n \cdot t \cdot \sqrt{n}$ one-hop transmissions per round. Since transmission is expensive, it would be desirable to lower the communication overhead while keeping the same level of security. In this section we introduce TTL-CoMAC, an extension of CoMAC that uses a technique similar to the TTL of the IPv4 protocol, to decrease the communication overhead of the protocol described in Section 5.1. The approach we propose is inspired by *gossiping* [31,32]. We only present details for TTL-CoMAC; ExCo can be extended with TTL in a similar way.

TTL-CoMAC differs from CoMAC in the way s_j picks its co-authenticators. In the latter, co-authenticators were randomly chosen among all sensors. However, several sensors between s_j and its co-authenticators would spend their energy to forward s_j's request, without actively participating in the authentication of s_j's data. The idea behind TTL-CoMAC is to have those forwarding sensors become s_j's co-authenticators.

Let $d(s_j)$ be the one-hop neighborhood of s_j. In TTL-CoMAC, s_j prepares a co-authentication request $\{s_j, r, d_j^r, TTL = t\}$ and sends it to a random neighbor (Algorithm 5).

Algorithm 5 TTL-CoMAC: *MAIN*

```
Sense $d_j^r$
Compute $z_j^r = MAC(K_j^r, d_j^r)$
Store $(r, d_j^r, z_j^r)$
Set $s_q \leftarrow_{\$} d(s_j)$
Send $\{s_j, r, d_j^r, t\}$ to $s_q$
```

Algorithm 6 TTL-CoMAC: *RECEIVE*

```
Set $R_j^r = \emptyset$
Set $X_j^r = \emptyset$
Set $c = 1$
while round not over do
  Receive $\{s_l, r, d_l^r, TTL\}$ from $s_p$
  if $TTL > 0$ then
    if $(s_l, *, *) \notin X_j^r$ then
      Compute $MAC(K_j^r, d_l^r)$
      Store $\left(r, s_l, MAC(K_j^r, d_l^r)\right)$
      Set $R_j^r[c] = l$
      Set $c = c + 1$
      if $TTL > 1$ then
        $s_q \leftarrow_{\$} d(s_j) \setminus \{\{s_p\} \cup \{s_l\}\}$
        $X_j^r = X_j^r \cup \{(s_l, s_p, s_q)\}$
        Send $\{s_l, r, d_l^r, TTL - 1\}$ to $s_q$
      end if
    else
      $W = d(s_j) \setminus \{X_j^r \cap \{(s_l, *, *)\} \cup \{s_p\}\}$
      if $W = \emptyset$ then
        $s_q \leftarrow_{\$} d(s_j) \setminus \{\{s_p\} \cup \{s_l\}\}$
      else
        $s_q \leftarrow_{\$} d(s_j) \setminus W$
        $X_j^r = X_j^r \cup \{(s_l, s_p, s_q)\}$
      end if
      Send $\{s_l, r, d_l^r, TTL\}$ to $s_q$
    end if
  end if
end while
Store $R_j^r$
Compute $K_j^{r+1} = F(K_j^r || R_j^r[1] || \ldots || R_j^r[|R_j^r|])$
```

When s_j receives from s_p a co-authentication request (Algorithm 6) such as $\{s_l, r, d_l^r, TTL\}$, the former checks if $TTL > 1$; in this case it authenticates d_l^r, decreases the TTL field of the request and forwards it to a random sensor in $d(s_j) \setminus \{s_p\} \cup \{s_l\}$. If $TTL = 1$, data is authenticated but the message is discarded. s_j also keeps a log of received requests as a list of tuples $\{s_l, s_p, s_q\}$, where s_p and s_q are the sensor the request was received from and the one it was sent to, respectively. When s_j receives an already processed request from s_p, it only forwards the request to a random sensor the request was not previously forwarded to. If no such sensor exists, s_j forwards the request to any neighbor but s_p and s_l.

In TTL-CoMAC, each sensor sends one co-authentication request per round, that is forwarded $O(t)$ times, for a total of $O(n \cdot t)$ transmissions per round. As all sensors along the route of the request take part in the authentication process, it is easy to see that the average probability that target data survives unforged is $P_{\text{TTL-CoMAC}}(n, k, \bar{r}, t) = P_{\text{CoMAC}}(n, k, \bar{r}, t)$. In other words, TTL-CoMAC keeps the same security level of CoMAC but decreases communication overheard of a factor of $\sqrt{n}$. The same reasoning can be extended to TTL-ExCo as well.

TTL-based collaborative authentication protocols are effective against the adversary considered in this paper, but might perform worse than the protocols of Section 5 in the presence of a smarter adversary. In both TTL-Comac and TTL-ExCo, any two co-authenticators are one hop away and, as a consequence, all peers involved in the authentication of a particular data item are located in the network area around the originating sensors. A smart μADV might focus its compromise activity in one particular area of the network and would succeed in forging data on behalf of sensors in μADV's compromise area. However, we stress that μADV's target sensor is not known in advance and μADV has no knowledge of which might be the area to compromise to maximize its chance of forging target data.

10. Battery usage estimation

Sensors' dependence on limited battery power makes energy overhead an important aspect of any protocol for UWSNs. In this section, we evaluate the battery usage of the proposed schemes with respect to computation and communication overhead factors.

Computation: Omitting normal CPU duty cycle costs, computational overhead mainly comes from the use of cryptography. Several researchers [33,34] have investigated the cost of cryptography in WSNs and all of them endorse the use of symmetric-

Table 2
Computation and Communication costs.

(a) Computation cost		(b) Communication cost		
Protocol	# *HMAC* (per node)	Protocol	Generated msg (per node)	Routing cost (network)
CoMAC	$O(t)$	CoMAC	t	$O(t \cdot \sqrt{(n)})$
ExCo	$O(1)$	ExCo	t	$O(t \cdot \sqrt{(n)})$
TTL-CoMAC	$O(t)$	TTL-CoMAC	1	$O(t)$
TTL-ExCo	$O(1)$	TTL-ExCo	1	$O(t)$

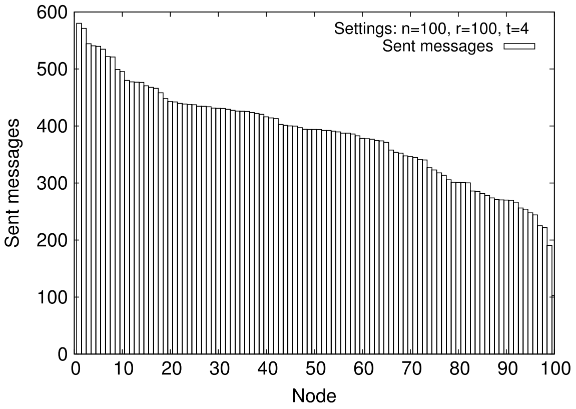


Fig. 8. TTL-CoMAC. Number of messages sent by each node.

key primitives based on block ciphers to obtain MACs. Without considering a specific cipher, we estimate the overhead of our protocols by measuring the number of MAC operations each sensor has to perform (see Table 2(a)). CoMAC and TTL-CoMAC require each sensor to compute, on average, $O(t)$ MACs per round; ExCo and TTL-ExCo only require $O(1)$ MACs per round.

Communication: The energy spent on message exchanges in CoMAC and Ex-Co depends heavily on a given sensor's location in the UWSN and the underlying routing protocol. Both CoMAC and ExCo assume an underlying routing protocol that provides full connectivity. Moreover, note that WSN routing suffers from many undesirable issues due to the packet relaying performed by each node [31,35,32].

TTL-CoMAC and TTL-ExCo are less affected by parameters such as position and routing. The use of gossiping allows us to ignore routing/geographical constraints and to obtain a more balanced energy consumption throughout the network. To assess the gain obtained with gossiping, we report average results of simulations performed using OMNET++. We considered an UWSN composed of 100 sensors uniformly distributed over a square field of 40,000 m^2 (200 m $*$ 200 m). Sensor transmission radius was set to 30 m, which resulted in an average neighborhood of 8 peers (standard deviation 3.28). Simulations were run 1000 times with each run having 100 rounds. Fig. 8 shows the distribution of messages; each sensor forwards, on average, 385 messages (standard deviation is 88.2), despite the unavoidable communication constraints suffered by nodes close to area boundaries. Since $t = 4$, the expected number of messages forwarded by every node at each round is 400 (Table 2(b)).

Fig. 9 plots the deployment field and shows the number of messages forwarded by each node in the network as a mean to estimate energy consumption. It shows that battery usage is almost evenly spread among all nodes. In both CoMAC and ExCo, it is natural to expect higher overhead for sensors in the center of the network, due to routing choices. Local choices on where to forward messages made in TTL-CoMAC and TTL-ExCo result in a more efficient and balanced energy consumption.

11. Self-healing for better security

The security of the proposed authentication schemes decreases with the number of rounds the UWSN remains unattended (Fig. 5). Even if the number of co-authenticators is tuned as in Section 8, survival probability drops to zero for data collected from round $\lceil \frac{n}{k} \rceil$ on. This is because each sensor evolves its key in a standard pseudo-random fashion that only achieves forward secrecy. By round $\lceil \frac{n}{k} \rceil$, μADV learns the key of each sensor and, by mimicking the key evolution process, it can compute keys for any such sensor for all future rounds. To remedy this, either the sink has to visit the UWSN before (or at) round $\lceil \frac{n}{k} \rceil$, or we need to provide backward secrecy for the authentication keys. Specifically, if $v \geq \lceil \frac{n}{k} \rceil$, we need a mechanism that allows a previously compromised sensor to compute a key unknown to μADV.

Some recent results [5,6] proposed sensor collaboration techniques to regain secrecy after sensor compromise. These techniques consider the same adversary and divide the UWSN into three disjoint sets: (1) sensors that are currently compromised (red), (2) sensors that have been previously compromised — their keys are known by μADV (yellow), and (3) sensors with keys not known to μADV (green). The last set is composed of sensors that either have never been compromised

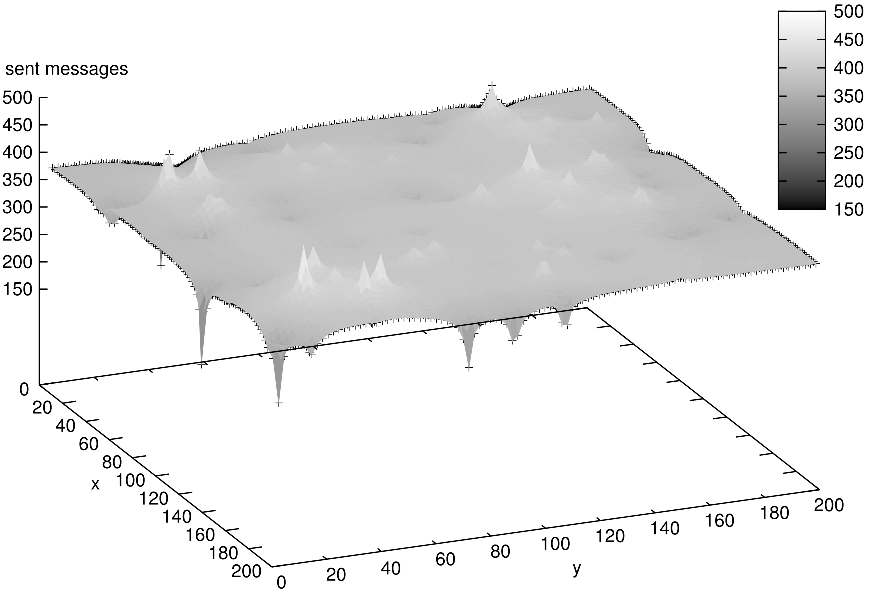


Fig. 9. TTL-CoMAC. Distribution of messages throughout the network.

or have regained key secrecy after compromise. To regain key secrecy, sensors exchange randomness and use received random contributions along with their current state to compute next-round keys. A red sensor cannot regain secrecy until μADV releases it. Whereas, a yellow sensor that receives a random contribution from at least one green peer computes a next-round key that μADV cannot learn.

Using the same terminology in both CoMac and ExCo approaches, a sensor that is compromised and later released goes from green to red, and finally, to yellow where it remains, i.e., never becomes green again. If collaborative self-healing is used (as in [5,6]), yellow sensors can regain key secrecy and μADV would be thus unable to forge authentication tags.

Since we use generic MACs for authentication, the sink must be able to re-compute each sensor's per-round key in order to verify tags, i.e., the sink must "synchronize" each key it shares with each sensor. As noted in [6], if we use key evolution through sensor collaboration, key synchronization by the sink becomes difficult or even impossible in the presence of sensor failures.

To overcome this problem in the context of data confidentiality, [6] resorts to public key cryptography. Each sensor computes a session key through the collaborative key update mechanism and encrypts the latter with the sink public key. Later, the sink decrypts the session key with its private key and uses the session key to decrypt ciphertexts of sensor measurements.

Unfortunately, the use of public key cryptography does not help data authentication. μADV can still delete the target data along with its authentication tag and the ciphertext of the authentication key. It can then pick a random authentication key, compute a MAC on arbitrary data and encrypt the authentication key with the sink's public key. The sink would consider fraudulent data to be genuine and the forgery would succeed.

To the best of our knowledge, no other self-healing protocol is suitable for the UWSN scenario, and effective authentication of data collected after total compromise of the network remains an open research problem.

12. Conclusion

In this paper, we focused on a mobile adversary attempting to replace authentic data in UWSNs. We proposed several techniques based on sensor cooperation that achieve much higher security than prior results, with relatively low overhead. We explored the effectiveness of proposed techniques, both analytically and via simulations. We also demonstrated how they cope with real network issues, such as message loss and sensor failure.

Acknowledgements

Claudio Soriente has been supported by Madrid Regional Council – CAM under the project CLOUDS (S2009TIC-1692), the Spanish Research Agency – MICINN under project CloudStorm (TIN2010-19077) and the European Commission under project MASSIF (FP7-257475).

Angelo Spognardi is partially supported by the Prevention, Preparedness and Consequence Management of Terrorism and other Security-related Risks Programme European Commission – Directorate-General Home Affairs, under the ExTraBIRE project, HOME/2009/CIPS/AG/C2-065.

References

[1] R. Di Pietro, L.V. Mancini, C. Soriente, A. Spognardi, G. Tsudik, Catch me (if you can): data survival in unattended sensor networks, in: 6th Annual IEEE International Conference on Pervasive Computing and Communications, PerCom'08, 2008, pp. 185–194.

[2] R. Di Pietro, L.V. Mancini, C. Soriente, A. Spognardi, G. Tsudik, Data security in unattended wireless sensor networks, IEEE Transactions on Computers 58 (11) (2009) 1500–1511.

[3] D. Ma, C. Soriente, G. Tsudik, New adversary and new threats: security in unattended sensor networks, IEEE Network 23 (2) (2009) 43–48.

[4] R. Di Pietro, L.V. Mancini, C. Soriente, A. Spognardi, G. Tsudik, Playing hide-and-seek with a focused mobile adversary in unattended wireless sensor networks, Ad Hoc Networks 7 (8) (2009) 1463–1475.

[5] D. Ma, G. Tsudik, DISH: distributed self-healing (in unattended sensor networks), in: 10th International Symposium on Stabilization, Safety, and Security of Distributed Systems, SSS'08, 2008, pp. 47–62.

[6] R. Di Pietro, D. Ma, C. Soriente, G. Tsudik, Posh: Proactive co-operative self-healing in unattended wireless sensor networks, in: 27th IEEE Symposium on Reliable Distributed Systems, SRDS'08, 2008, pp. 185–194.

[7] R. Di Pietro, C. Soriente, A. Spognardi, G. Tsudik, Collaborative authentication in unattended WSNs, in: 2nd ACM Conference on Wireless Network Security, WISEC'09, 2009, pp. 237–244.

[8] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, D.E. Culler, Spins: security protocols for sensor networks, Wireless Networks 8 (5) (2002) 521–534.

[9] D. Liu, P. Ning, S. Zhu, S. Jajodia, Practical broadcast authentication in sensor networks, in: 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, MOBIQUITOUS'05, 2005, pp. 118–132.

[10] S. Zhu, S. Setia, S. Jajodia, P. Ning, Interleaved hop-by-hop authentication against false data injection attacks in sensor networks, ACM Transactions on Sensor Networks 3 (3) (2007) 14. http://doi.acm.org/10.1145/1267060.1267062.

[11] F. Ye, H. Luo, S. Lu, L. Zhang, Statistical en-route filtering of injected false data in sensor networks, IEEE Journal on Selected Areas in Communications 23 (4) (2005) 839–850.

[12] W. Zhang, G. Cao, Group rekeying for filtering false data in sensor networks: a predistribution and local collaboration-based approach, in: 24th IEEE International Conference on Computer Communications, INFOCOM'05, 2005, pp. 503–514.

[13] Y. Zhang, J. Yang, H.T. Vu, The interleaved authentication for filtering false reports in multipath routing based sensor networks, in: 20th International Parallel and Distributed Processing Symposium, IPDPS'06, 2006, pp. 1–10.

[14] H. Yang, F. Ye, Y. Yuan, S. Lu, W. Arbaugh, Toward resilient security in wireless sensor networks, in: 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc'05, 2005, pp. 34–45.

[15] M. Bohge, W. Trappe, An authentication framework for hierarchical ad hoc sensor networks, in: Proceedings of the 2nd ACM Workshop on Wireless Security, WiSe'03, 2003, pp. 79–87.

[16] D. Ma, G. Tsudik, Extended abstract: forward-secure sequential aggregate authentication, in: IEEE Symposium on Security and Privacy, S&P'07, 2007, pp. 86–91.

[17] A.A. Yavuz, P. Ning, Hash-based sequential aggregate and forward secure signature for unattended wireless sensor networks, in: 6th Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, MOBIQUITOUS'09, 2009, pp. 1–10.

[18] W. Zhang, H. Song, S. Zhu, G. Cao, Least privilege and privilege deprivation: towards tolerating mobile sink compromises in wireless sensor networks, in: 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc'05, 2005, pp. 378–389.

[19] L. Zhou, J. Ni, C.V. Ravishankar, Supporting secure communication and data collection in mobile sensor networks, in: 25th IEEE International Conference on Computer Communications, INFOCOM'06, 2006, pp. 1–12.

[20] D. Liu, Efficient and distributed access control for sensor networks, in: 3rd IEEE International Conference on Distributed Computing in Sensor Systems, DCOSS'07, 2007, pp. 21–35.

[21] M. Shao, S. Zhu, W. Zhang, G. Cao, Y. Yang, pDCS: security and privacy support for data-centric sensor networks, IEEE Transactions on Mobile Computing 8 (8) (2009) 1023–1038.

[22] J. Luo, P. Papadimitratos, J. Hubaux, Gossicrypt: wireless sensor network data confidentiality against parasitic adversaries, in: 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON'08, 2008, pp. 441–450.

[23] M. Bellare, R. Canetti, H. Krawczyk, Keying hash functions for message authentication, in: 16th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO'96, 1996, pp. 1–15.

[24] National Institute of Standards and Technology (NIST), Secure hash standard, fips pub 180-2, 2002. http://www.itl.nist.gov/fipspubs/fip180-2.htm.

[25] Y. Dodis, J. Katz, S. Xu, M. Yung, Key-insulated public key cryptosystems, in: International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT'02, 2002, pp. 65–82.

[26] Y. Dodis, J. Katz, S. Xu, M. Yung, Strong key-insulated signature schemes, in: 6th International Workshop on Theory and Practice in Public Key Cryptography, PKC'03, 2003, pp. 130–144.

[27] Y. Dodis, M.K. Franklin, J. Katz, A. Miyaji, M. Yung, A generic construction for intrusion-resilient public-key encryption, in: The Cryptographers' Track at the RSA Conference, CT-RSA'04, 2004, pp. 81–98.

[28] G. Itkis, L. Reyzin, Sibir: Signer-base intrusion-resilient signatures, in: 22nd Annual International Cryptology Conference on Advances in Cryptology, CRYPTO'02, Springer-Verlag, London, UK, 2002, pp. 499–514.

[29] A. Varga, The OMNeT++ discrete event simulation system, in: 15th European Simulation Multiconference, ESM'01, 2001.

[30] A. Varga, Using the OMNeT++ discrete event simulation system in education, Education, IEEE Transactions on 42 (4) (1999) 11 pp.

[31] K. Akkaya, M.F. Younis, A survey on routing protocols for wireless sensor networks, Ad Hoc Networks 3 (3) (2005) 325–349.

[32] J.N. Al-Karaki, A.E. Kamal, Routing techniques in wireless sensor networks: a survey, IEEE Wireless Communications 11 (6) (2004) 6–28.

[33] Y.W. Law, J. Doumen, P. Hartel, Survey and benchmark of block ciphers for wireless sensor networks, ACM Transactions on Sensor Networks 2 (1) (2006) 65–93.

[34] A.S. Wander, N. Gura, H. Eberle, V. Gupta, S.C. Shantz, Energy analysis of public-key cryptography for wireless sensor networks, in: 3rd IEEE International Conference on Pervasive Computing and Communications, PERCOM'05, 2005, pp. 324–328.

[35] C. Schurgers, M. Srivastava, Energy efficient routing in wireless sensor networks, 2001, pp. 357–361.