Collaborative Authentication In Unattended WSNs

Roberto Di Pietro
Università di Roma Tre
dipietro@mat.uniroma3.it

Claudio Soriente
University of California, Irvine
csorient@ics.uci.edu

Angelo Spognardi
INRIA Rhône-Alpes
spognard@inrialpes.fr

Gene Tsudik
University of California, Irvine
gts@ics.uci.edu

ABSTRACT

An unattended wireless sensor network (UWSN) might collect valuable data representing an attractive target for the adversary. Since a sink visits the network infrequently, unattended sensors cannot immediately off-load data to some safe external entity. With sufficient time between sink visits, a powerful mobile adversary can easily compromise sensor-collected data.

In this paper, we propose two schemes (CoMAC and ExCo) that leverage sensor co-operation to achieve data authentication. These schemes use standard (and inexpensive) symmetric cryptographic primitives coupled with key evolution and few messages exchange. We provide security analysis for proposed schemes and assess their effectiveness via simulations. We show that proposed schemes cope well with real WSN issues, such as message loss and sensor failure. We also compare the two schemes with respect to robustness and overhead, which allows network designers to carefully select the right scheme and tune appropriate system parameters.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.4 [**Computer Systems Organization**]: computer communication networks—*Distributed Systems*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms

Performance, Security

Keywords

Wireless Sensor Networks, Authentication, Mobile Adversary

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WiSec'09, March 16–18, 2009, Zurich, Switzerland.
Copyright 2009 ACM 978-1-60558-460-7/09/03 ...$5.00.

1. INTRODUCTION

Unattended Wireless Sensor Networks (UWSN-s) are an emerging class of WSN-s where the sink visits the network to collect sensed data at infrequent intervals. UWSN nodes store sensed data locally until the next sink visit. In contrast, much of prior research in WSN security assumes that the network is constantly attended by a trusted entity (collector or sink) to which data can be transferred at will.

The inability to off-load sensed data in real time makes UWSN-s susceptible to a wide range of attacks. Within the time between sink visits, an adversary ($\mathcal{ADV}$ from now on) can compromise sensors, manipulate sensed data, and leave the network without leaving a trace. The type of adversary that roams the network gradually compromising nodes is often called a *mobile adversary*. Such an adversary has two salient characteristics: (1) it can simultaneously compromise up to k-out-of-n nodes (where $k < n$ and n is the network size); and, (2) it can migrate between sets of compromised nodes. In the last year, several consecutive research results [6, 16, 5] defined the UWSN mobile adversary, identified several sub-types varying in goals and behavior, and suggested some mitigating techniques. Such techniques are necessary, since prior solutions from WSN security literature are unsuitable for coping with a mobile adversary in UWSN-s.

In this paper, we focus on the type of the mobile adversary that aims to introduce fraudulent data into the UWSN. Such an adversary might attempt to do so by replacing authentic data obtained by sensors or by simply injecting fraudulent data. Within this framework, we make several **contributions**. First, we overview current solutions for the authentication problem in UWSN-s and show that they only offer weak protection against a mobile adversary. Second, we construct two techniques – based on sensor cooperation – that achieve much higher security (compared to prior results) with small increase in overhead. Both techniques rely only on symmetric cryptography. We explore the effectiveness of proposed techniques both analytically and via simulations. Furthermore, we show how they cope with real network issues, e.g., message loss and sensor failure. Analytical and simulation results indicate that our schemes are resilient as well as efficient and allow network designers to easily find the desired balance between overhead and robustness.

The rest of the paper is organized as follows: next section surveys related work. Section 3 introduces the system model, in particular the network assumptions and the adversarial model. Section 4 shows that traditional authentication techniques do not provide sufficient security in UWSN-s. Proposed cooperative schemes are introduced in Section 5

and security analysis is provided in Section 6. Simulation results and discussion are presented in Section 7. Section 8 provides concluding remarks.

2. RELATED WORK

WSN security is a mature research area. A number of important topics were considered in the prior literature, such as: secure initialization, key management, secure/resilient routing, clone detection, broadcast authentication and secure localization. Data authentication has been the subject of several research efforts, however, all of them relied on the constant presence of the sink. Several schemes, e.g., [17, 14], have been proposed for efficient sink-to-sensors broadcast authentication. In contrast, we focus on sensor-to-sink data authentication, in the envisaged UWSN setting, in the presence of a mobile adversary.

In some prior work, sensors are assumed capable of detecting false data, via collaborative mechanisms for value endorsement. For example, [22] describes an interleaved authentication scheme for filtering false data. It is based on the assumption that data with a list of MAC-s are routed towards the sink and recursively verified along the route. Sensors are grouped into clusters of size at least $t+1$, where t is a collusion threshold: if t or fewer sensors are compromised, the network detects false data before it traverses $t+1$ hops towards the sink.

A similar technique is Statistical En-route Filtering (SEF) [19]. In it, a data packet is authenticated by each cluster sensor with an individual MAC. Assuming a suitable key pre-distribution scheme, one MAC is verified with a certain probability at each hop along the route. When a packet reaches the sink, all enclosed MAC-s are verified. An extension to SEF can be found in [20], where the effectiveness of filtering is enhanced by using a family of pre-distribution and local collaborative cluster re-keying schemes.

Other schemes using en-route filtering include [18] and [21]: the former incorporates localization information for additional data protection, while the latter provides extra security via multi-path routing.

An authentication framework for hierarchical sensor networks is presented in [4]. It leverages network heterogeneity to relegate expensive crypto-operations to the most powerful sensors.

While many prior techniques provide bottom-up (sensors-to-sink) authentication, we focus on Unattended WSN-s, without sink's constant presence. As mentioned earlier, some very recent results considered UWSN-s but focused on different problems and mobile adversary sub-types, such as data survival [6] and key healing [16, 5] for intrusion-tolerant secrecy.

We note that data authentication in UWSN-s was first investigated in [15] which constructed forward-secure aggregate authentication techniques that optimize storage requirements. However, this is effective only against a relatively weak adversary, referred to as *reactive* in [7]. (Moreover, no communication among sensors is assumed in [15]).

Techniques proposed in this paper share some features with cooperative key-healing approaches in [16] and [5]. In both [16] and [5], sensors regain security after compromise using randomness provided by non-compromised peers. Secrets derived from combining contributions by peers with prior state, are used either as keys (with symmetric encryption) or as randomizers (with public key encryption). However, the symmetric key case is very fragile since it assumes no lost messages and no node failures. The public key case is thus the only robust approach. Unfortunately, it does not apply to data authentication, for the following reason. A secret value can be used as a randomizer in public key encryption (under the sink's public key) such that, given a ciphertext, learning the corresponding plaintext requires guessing this randomizer, which is computationally unfeasible. The same secret value cannot be used as a private key in computing digital signatures, since signature verification requires the verifier (sink) to obtain the corresponding public key; but the public key (being dependent on the private key) is not always computable by the sink, due to lost messages and node failures.

Solutions explored in this paper rely on purely symmetric techniques and offer robustness in spite of imperfect communication and potential node failures. Also, our techniques achieve higher security than [16, 5] with the same bandwidth overhead.

3. SYSTEM MODEL

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

$\mathcal{ADV}$	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	$\mathcal{ADV}$'s compromise power – maximum size of C_r
$s_1 ... s_n$	sensors
s_i	$i \in [1..n]$ node that collected target data
$\bar{r}$	target data collection round
d_j^r	data collected by s_j in round r
K_j^r	key used by s_j in round r
z_j^r	MAC computed by s_j at round r using K_j^r
t	numbers of co-authenticators

Table 1: Notation

3.1 Network assumptions

We assume a homogeneous UWSN composed by a set of sensors $\{s_1, s_2, \ldots, s_n\}$ uniformly distributed over a certain geographical area. Notable details and assumptions are as follows (similar to those in [5]):

- **Data collection:** Time is divided in collection rounds and each sensor collects a single data item per round. Round synchronization can be obtained via any well-known technique, such as [11].
- **Unattended operation:** An itinerant sink periodically visits the UWSN to collect data. There is a system-wide parameter – v – denoting the maximum number of data collection rounds between successive sink visits.
- **Communication:** The UWSN is always connected and any two sensors can communicate either directly or through peers, according to the underlying routing protocol. However, messages can be lost and sensors can fail.
- **Storage:** Each sensor has storage to accommodate $O(v)$ data items.

- **Cryptographic Capabilities:** Each sensor can perform cryptographic hashing and has a Pseudo-Random Number Generator (PRNG) initialized with a unique secret seed, shared with the sink.

- **Re-initialization:** At each visit, the sink securely re-initializes secret seed values for all sensors and resets the round counter.

3.2 Adversarial model

The envisaged adversary has the following features:

- **Goal:** $\mathcal{ADV}$ aims to substitute a value obtained by s_i at round $\bar{r}$ with an arbitrary value of its choice. We assume that any sensor and any round are equally likely to be chosen as $\mathcal{ADV}$'s targets. $\mathcal{ADV}$ learns its objective, namely s_i and $\bar{r}$, at the end of round $\bar{r}$ and from that moment it has $v - \bar{r}$ rounds to accomplish its goal. $\mathcal{ADV}$ is successful if, at the end of round v, the sink visits the network to collect sensed data, and validates the value injected by the adversary, as a measurement of s_i at round $\bar{r}$. We stress the fact that the network is unaware of s_i and $\bar{r}$; thus, each collected data item must be equally protected.

- **Compromising capabilities:** $\mathcal{ADV}$ can compromise a constant number k ($k < n$) of sensors at any round between 1 and v. If a sensor is compromised at round r, $\mathcal{ADV}$ acquires keys and status, reads all storage/memory, and can eavesdrop on all incoming and outgoing communications of the compromised sensor. We further distinguish two types of $\mathcal{ADV}$:
  Reactive: $\mathcal{ADV}$ starts compromising sensors at round $\bar{r} + 1$, that is after identifying its target.
  Proactive: $\mathcal{ADV}$ starts compromising sensors at round 1, before receiving any information about the target sensor and the target data collection round.

- **Periodic Operation:** Time is divided into equal and fixed compromise rounds. At the end of each compromise round, $\mathcal{ADV}$ picks a subset of k sensors to be compromised in the following round — this subset can have non empty intersection with former one. At the start of each round, the adversary releases in an atomic way the subset of sensors compromised in the previous round and compromises the new subset.

- **Compromise Round & Collection Round:** For ease of exposition and without loss of generality, we assume that the compromise and collection rounds have the same duration and are synchronized, i.e., both types of rounds start and end at the same time.

- **Topology Knowledge:** $\mathcal{ADV}$ knows the entire topology of the UWSN.

- **Minimal Disruption:** $\mathcal{ADV}$ does not interfere with sensor behavior, mainly in order to remain undetected. In particular, it does not delete or delay messages and does not introduce spurious messages. $\mathcal{ADV}$'s goal is to find the target data and replace it.

- **Defense Awareness:** $\mathcal{ADV}$ is fully aware of any scheme or algorithm that the UWSN uses to protect itself.

Note that a proactive $\mathcal{ADV}$ starts compromising sensors before learning its target; thus, possibly $s_i \in C_{\bar{r}}$. This case is not interesting as $\mathcal{ADV}$ wins by dumb luck. We focus on a more interesting and likely scenario where $s_i \notin C_{\bar{r}}$ and possibly $s_i \in C_r,\ r < \bar{r}$. That is, $\mathcal{ADV}$ does not control s_i while target data is being collected, but it might have compromised (and then released) that sensor at some earlier round.

4. BASIC TECHNIQUES

Since $\mathcal{ADV}$ is focused on replacing (modifying or manufacturing) data, authentication is a natural defence. That can be achieved via either message authentication codes (MAC-s) or public key signatures. Although it is normal to expect the latter to offer better security, we show that this is not true in UWSN-s.

With MAC-s, each sensor s_j has a unique secret key K_j (shared with the sink) used for computing MAC-s. At each round r, s_j collects d_j^r and computes $z_j^r = MAC(K_j, d_j^r)$, where $MAC(\circ)$ can be the standard HMAC. During its next visit, the sink collects all pairs $\{d_j^r, z_j^r\}_{1 \le r \le v}$ and proceeds to verify their authenticity.

With public key signatures, each sensor s_j can be 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. When the sink collects all pairs $\{d_j^r, \sigma_j^r\}_{1 \le r \le v}$, it can verify each signature with s_j's public key.

Unfortunately, neither approach is effective against our mobile adversary, not even against the reactive version. Indeed, once the reactive $\mathcal{ADV}$ learns $\{s_i, \bar{r}\}$, it compromises s_i during round $\bar{r} + 1$ and learns K_i (in case of MAC) or SK_i (in case of public key signatures); knowledge of this key allows $\mathcal{ADV}$ to produce MAC-s or signatures.

Note that off-loading data to random peers, as investigated in [6], does not help much since, after a few rounds, $\mathcal{ADV}$ can locate and replace the target data.

A natural defense against a reactive $\mathcal{ADV}$ is the use of *forward-secure* cryptographic techniques. Such techniques rely on per-round key evolution that makes computationally infeasible for $\mathcal{ADV}$ to compute a prior key from any of its descendants. Forward-secure MAC [3] and forward-secure signature schemes [2] can be found in the literature. With a forward-secure scheme, even if the reactive $\mathcal{ADV}$ compromises s_i at round $\bar{r}+1$ and learns s_i's current key, it cannot compute the key used by s_i in round $\bar{r}$. Once $\mathcal{ADV}$ compromises a sensor, it does not matter whether forward-secure MAC-s or signatures are used; in either case, $\mathcal{ADV}$ can only evolve the current key into the future.

Although forward security is effective against a reactive $\mathcal{ADV}$, it does not protect against its proactive counterpart. Recall that: (1) if $\mathcal{ADV}$ is proactive we could have $s_i \in C_r,\ r < \bar{r}$, and (2) with a forward-secure authentication scheme, $\mathcal{ADV}$ can compute keys that will be used by compromised sensors in any round after compromise. Thus, if $s_i \in C_r,\ r < \bar{r}$, $\mathcal{ADV}$ can compute $K_i^{\bar{r}}$ and replace both target data and its authentication tag. Finally, if $s_i \notin C_r,\ r < \bar{r}$, the situation is essentially the same as with the reactive $\mathcal{ADV}$.

To protect against a proactive $\mathcal{ADV}$, we need another property, called *backward security*. Informally, an authentication scheme is backward secure if knowledge of the secret

for the current round cannot lead to learning the next round secret. An authentication scheme that is both forward and backward secure is called *key-insulated*. Unfortunately, current key insulated schemes [9, 10, 8, 12, 13] are unsuitable for our setting. The reason is that, in a key-insulated scheme, at least one of the inputs for the key evolution function must come from a separate entity, typically, 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 third party is realistic.

In summary, we state that no current authentication scheme can achieve the desired level of security (both forward and backward security) for UWSN-s, to cope with a proactive adversary.

5. COOPERATIVE MECHANISMS

The main shortcoming of aforementioned basic 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 thus be easily replaced. Involving several sensors in the authentication process of a single data item can lead to more secure schemes. In the following we construct and evaluate two schemes that enhance UWSN security using so-called *co-authenticators*. We say that s_j is a *co-authenticator* of s_l if s_l requires s_j's cooperation to authenticate its data. By *security* we mean the probability that target data cannot be replaced (until the next visit of the sink). Since, as discussed above, public key signatures do not offer any particular advantage, we consider schemes that only rely on symmetric cryptography.

5.1 Co-MAC

The main idea in this scheme is that each sensor asks a set of randomly chosen peers to authenticate its data and to keep the resulting tags.

At round 1, each s_j has an initial key K_j^1 shared with the sink. At the end of each round, a sensor evolves its current key via a one-way collision-resistant function $F(\circ)$ to compute the next round's key. At any round, s_j runs two separate processes, *MAIN* and *RECEIVE*. The former computes MAC of sensed data and commits it to storage. It also sends the data to t randomly chosen co-authenticators. The *RECEIVE* process receives data from peers, computes MAC-s, commits them to storage and computes the next round key.

Process 1 CoMAC: *MAIN*

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

The function $SELECT_DISTINCT(t, n, j)$ returns an array of t distinct elements randomly chosen from the set: $\{1, \ldots, n\} \setminus \{j\}$. Its output is based on the results of the PRNG that is initialized with a secret seed given by the sink, and then queried at every round. This way, the sink knows S_j^r for any s_j and any r.

Process 2 CoMAC: *RECEIVE*

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

During its next visit, the sink acquires collected data along with corresponding MAC-s. 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 $t + 1$ MAC-s are verified successfully, the sink considers d_j^r to be authentic, i.e., actually collected by s_j at round r. Note that computation of any K_j^r requires knowledge of K_j^{r-1} as well as knowledge of the sensors that s_j received a message from, at round r (namely R_j^r). The sink can compute any K_j^r as it knows the initial key of s_j and can read the sets $\{R_j^{r'}\}_{1 \le r' \le r-1}$ from s_j' storage. At any round, each sensor sends t messages and stores, on average, $t + 1$ MAC-s.

In the present scheme, each MAC authenticates a single data item and is stored by the sensor that produced it. As we discuss below, better security can be achieved by bundling batches of MAC-s, computed over different data items from multiple sensors.

5.2 Extensive Cooperation (ExCo)

In this scheme, each sensor sends the MAC computed over its own data to t randomly-chosen co-authenticators. Moreover, it bundles all received MAC-s into a single authentication tag. At round r, s_j runs the following processes:

Process 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 p=1..t **do**
  Send z_j^r to $s_{S_j^r[p]}$
end for

Process 4 ExCo: *RECEIVE*

Set $R_j^r = \emptyset$
Set $\Lambda_j^r = \emptyset$
p=1
while round is not over **do**
  Receive z_q^r
  Set $R_j^r[p] = q$
  Set $\Lambda_j^r[p] = z_q^r$
  Set $p = p + 1$
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 $\{d_j^r, H_j^r, R_j^r\}$

Process *MAIN* computes the MAC of current data and sends it to t randomly chosen peers. Process *RECEIVE*

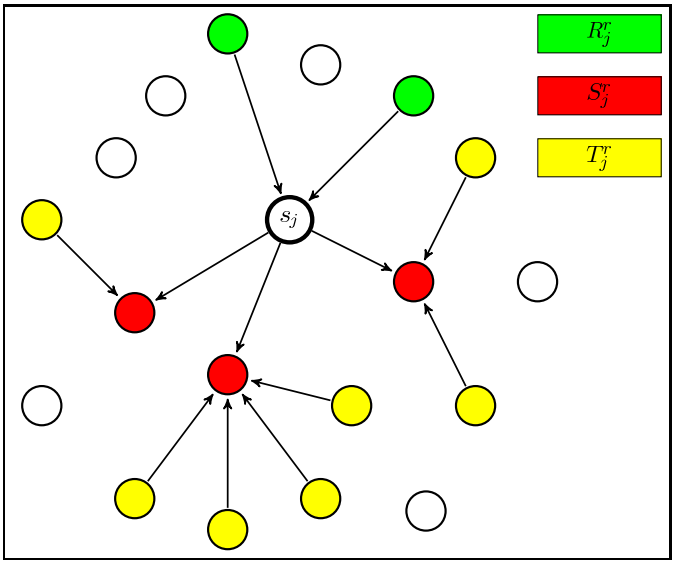


Figure 1: In ExCo, at round r, R_j^r contains the sensors which s_j received a MAC from, S_j^r contains the sensors s_j sent its MAC to, and T_j^r contains the sensors that sent the MAC to any sensor in S_j^r.

receives MAC-s from peers, aggregates them in a single authentication tag, commits it to storage and computes the next round key.

Data verification in ExCo is similar to that in CoMAC. Once the sink acquires d_j^r, it verifies all authentication tags involving that value. In particular, to verify d_j^r, the sink must compute:

- H_j^r using K_j^r and $\{K_q^r, d_q^r\}_{s_q \in R_j^r}$
- $\{H_q^r\}_{s_q \in S_j^r}$ using $\{K_q^r, d_q^r\}$ and $\{K_p^r, d_p^r\}_{s_p \in T_j^r}$ where $T_j^r = \{\bigcup_{s_l \in S_j^r} R_l^r\}$

By T_j^r we denote the set of sensors that have at least one co-authenticator in common with s_j at round r. As an example, Figure 1 shows a network where the colored sensors are the ones involved in the authentication of a value sensed by s_j at round r. An arrow between s_p and s_q means that $s_q \in S_p^r$. In order to verify d_j^r, the sink must verify all the authentication tags produced by $s_j \cup S_j^r$. That requires K_j^r as well as key and sensed data at round r from each sensor in $S_j^r \cup R_j^r \cup T_j^r$.

As before, the sink can compute the key used by any sensor at any round, knowing the initial key and using information found on that sensor. At any round, each sensor sends t messages and stores two authentication tags.

6. SECURITY ANALYSIS

In this section we analyze $\mathcal{ADV}$'s ability to replace target data and its authentication tag(s). We assume that $\mathcal{ADV}$ is proactive and roams the network before learning $\bar{r}$, hoping to discover as many keys as possible. To maximize the total number of collected keys (by round $\bar{r}$), $\mathcal{ADV}$ compromises sensors in a round-robin fashion, choosing C_r such that $C_r \cap \{C_1 \cup \ldots \cup C_{r-1}\} = \emptyset$. With this strategy, $\mathcal{ADV}$ 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.

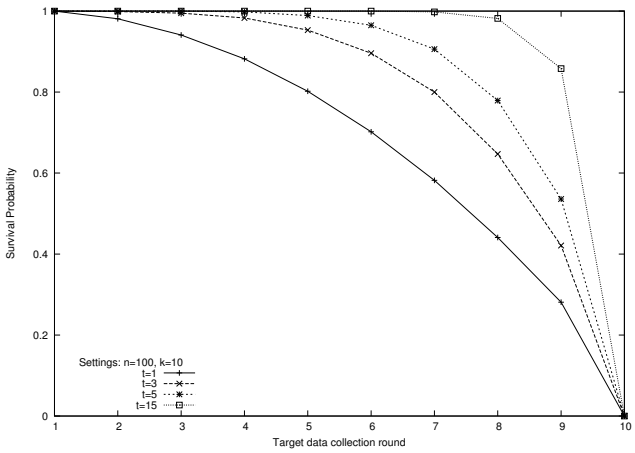


Figure 2: Survival probability of the target data, with CoMAC.

6.1 CoMAC

To "win" against CoMAC, $\mathcal{ADV}$ must replace authentication tags generated by s_i and $S_i^{\bar{r}}$. For this to happen:

$$s_i \cup S_i^{\bar{r}} \in \{C_1 \cup \ldots \cup C_{\bar{r}-1}\} \quad (1)$$

The probability of this event can be expressed as:

$$P_{\mathcal{ADV}}(n,k,\bar{r},t) = \begin{cases} \frac{k(\bar{r}-1)}{n} \cdot \prod_{i=1}^{t} \frac{k\bar{r}-i}{n-i} & \text{if } k\bar{r} \leq n \\ 1 & \text{otherwise} \end{cases}$$

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 probability that the target data remains unforged is:

$$P_{CoMAC}(n,k,\bar{r},t) = 1 - P_{\mathcal{ADV}}(n,k,\bar{r},t) \quad (2)$$

Figure 2 plots this probability in a UWSN 100 sensors, where $\mathcal{ADV}$ can compromise $k = 10$ sensors per round. It shows that, even if target data collection occurs after $\mathcal{ADV}$ compromises a substantial portion of the network, $\mathcal{ADV}$'s success probability is rather low, even for small values of t.

Using the set R_j^r to update s_j key at round r prevents $\mathcal{ADV}$ from deleting any MAC computed by s_j at that round. In fact, if $\mathcal{ADV}$ does not know the key of any sensor $s_j \in S_i^{\bar{r}}$, it might delete $MAC(K_j^{\bar{r}}, d_i^{\bar{r}})$ from s_j's storage, as if s_i's request to MAC its data, never reached s_j. With the adopted key evolution, if s_i is removed from $R_j^{\bar{r}}$, the sink will not be able to correctly compute $K_j^{\bar{r}+1}$. In particular, s_j would compute $K_j^{\bar{r}+1} = F(K_j^{\bar{r}} || R_j^{\bar{r}})$ while the sink would compute it as $K_j^{\bar{r}+1} = F(K_j^{\bar{r}} || R_j^{\bar{r}} \setminus \{s_i\})$. As a result, verification of any MAC computed by s_j from round $\bar{r}+1$ on, will fail and that would expose $\mathcal{ADV}$'s presence to the sink.

6.2 ExCo

With ExCo, $\mathcal{ADV}$ must replace (that is, must be able to recompute) all the authentication tags generated using $d_i^{\bar{r}}$. In particular, $\mathcal{ADV}$ 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, $\mathcal{ADV}$ must know $K_i^{\bar{r}}$. Computation of $H_i^{\bar{r}}$ requires also $(K_j^{\bar{r}}, d_j^{\bar{r}})$ for each

$s_j \in R_i^{\bar{r}}$. 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 $\{(K_q^{\bar{r}}, d_q^{\bar{r}})\}_{s_q \in S_j^{\bar{r}}}$.

To summarize, $\mathcal{ADV}$ must know all the following keys: $K_i^{\bar{r}}$, $\{K_j^{\bar{r}}, d_j^{\bar{r}}\}_{s_j \in R_i^{\bar{r}}}$, $\{K_j^{\bar{r}}, d_j^{\bar{r}}\}_{s_j \in S_i^{\bar{r}}}$ and $\{K_j^{\bar{r}}, d_j^{\bar{r}}\}_{s_j \in T_i^{\bar{r}}}$. That is, $\mathcal{ADV}$ 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 $r \leq \bar{r}$.

Note that since $\mathcal{ADV}$ can eavesdrop on traffic routed through compromised sensors, it does not require to know the pairs $\{K_j^{\bar{r}}, d_j^{\bar{r}}\}$ if it has intercepted the corresponding $z_j^{\bar{r}}$. As shown in Figure 1, to replace target data, $\mathcal{ADV}$ must know $K_i^{\bar{r}}$, as well as keys and sensed values at round r for every sensor in $S_i^{\bar{r}} \cup R_i^{\bar{r}} \cup T_i^{\bar{r}}$. To assess the effort $\mathcal{ADV}$ has to make to succeed, we estimate the probability of a sensor to be 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 the set $R_i^{\bar{r}}$ ($S_i^{\bar{r}}, T_i^{\bar{r}}$) 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 can write:

$$P[\mathcal{R}] = P[\mathcal{S}] = \frac{t}{n-1}$$

since a sensor always sends its value to t distinct sensors randomly chosen excluding itself and a sensor is selected – on average – by $\frac{t}{n-1}$ sensors as a co-authenticator. We can also write that

$$P[\overline{\mathcal{T}} | \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] = \frac{P[\overline{\mathcal{T}} \wedge \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}]}{P[\overline{\mathcal{R}} \wedge \overline{\mathcal{S}}]}$$

and then

$$P[\overline{\mathcal{T}} \wedge \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] = P[\overline{\mathcal{T}} | \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] \cdot P[\overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] \quad (3)$$

It is worth noting that $P[\overline{\mathcal{T}} | \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}] = P[\overline{\mathcal{T}} | \overline{\mathcal{R}}]$, since belonging to the set $T_i^{\bar{r}}$ is completely independent from belonging to set $S_i^{\bar{r}}$: i.e. sensor $s_j \in S_i^{\bar{r}}$ because has been chosen by s_i, but $s_j \in T_i^{\bar{r}}$ because itself chose as a co-authenticator another sensor in $S_i^{\bar{r}}$. To evaluate Equation 3 we can write

$$P[\overline{\mathcal{T}} | \overline{\mathcal{R}}] = \left(1 - \frac{t}{n-2}\right) \dots \left(1 - \frac{t}{n-2-(t-1)}\right) = \\ = \prod_{j=0}^{t-1} \left(1 - \frac{t}{n-2-j}\right) \geq \left(1 - \frac{t}{n-2}\right)^t \quad (4)$$

The last relation holds since sensor $s_j \notin T_i^{\bar{r}}$ only if s_j sent all its t MAC-s 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 $s_j \notin R_i^{\bar{r}}$). Since 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 $S_i^{\bar{r}}$.

Thus, we can estimate $\nu(n,t)$, the number of sensors involved in the computation of the authentication tags of generic data d_j^r, as

$$\nu(n,t) = n\left(1 - P[\overline{\mathcal{T}} \wedge \overline{\mathcal{R}} \wedge \overline{\mathcal{S}}]\right) \\ \geq n\left(1 - \left(\frac{t}{n-1}\right)^2 \left(1 - \frac{t}{n-2}\right)^t\right) \quad (5)$$

Figure 3 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 n from 100 to 1000: it reports the ratio between the number of times the set resulted of 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 Equation 5 produces 14.42 and 14.94, respectively.

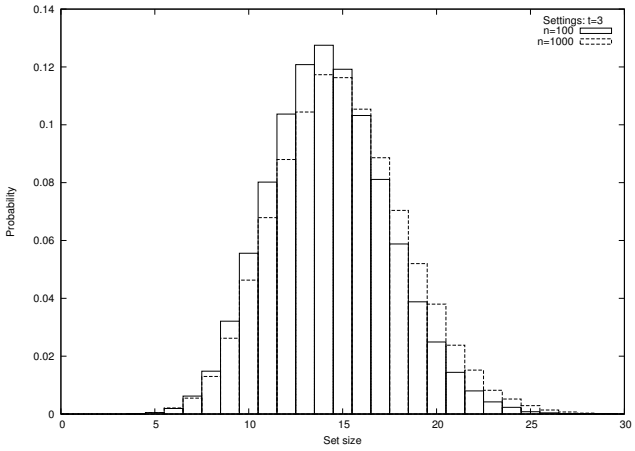


Figure 3: Distribution of the size of $R \cup S \cup T$.

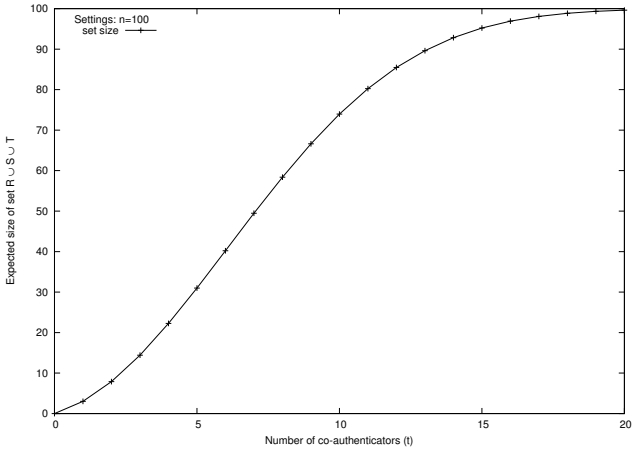


Figure 4: Expected size of $R \cup S \cup T$.

Figure 4 reports the expected size of the set $R_j^r \cup S_j^r \cup T_j^r$, obtained via Equation 5, fixing n to 100 and varying the number of co-authenticators t. It is clear that, fixed 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 $t+1$). In next section we presents the results of our simulations that confirm the above analysis (see Figure 5).

As in CoMAC, here as well key evolution is performed using the IDs of communicating peers so that $\mathcal{ADV}$ can not delete traces of the communication between sensors, without revealing its presence to the sink (cf. Section 6.1).

7. SIMULATION AND DISCUSSION

In this section we present the results of our simulations and discuss the robustness of presented schemes.

7.1 Simulation Results

To support the analysis of previous sections, we report the results of our simulations. Experiments were run for a network with 100 sensors with $k = 10$: for each scenario, $\mathcal{ADV}$ roams the network for $\bar{r}$ rounds, corrupting k different sensors at each round and collecting the keys that the sensors will use to generate authentication tags. The sensors, on the

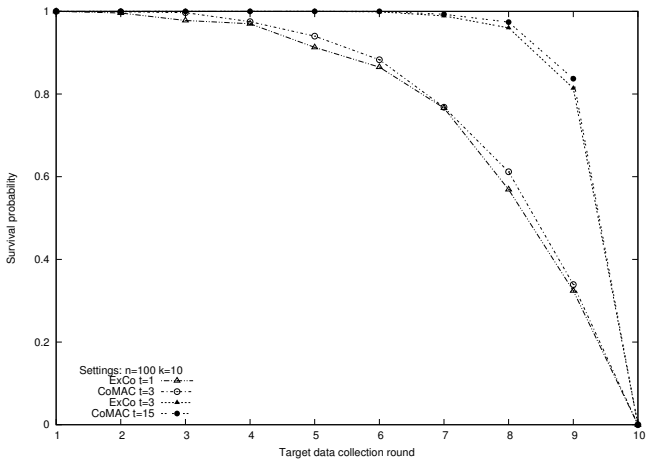


Figure 5: Comparison between CoMAC and ExCo.

other side, sense a data item per round and authenticate it according to the selected scheme. After $\bar{r}$ rounds, we randomly select target data among all data item collected during the current round and verify if $\mathcal{ADV}$ has all the required keys used to produce the authentication tags. Experiments were repeated 1000 times for each configuration of t and $\bar{r}$; the reported values are the resulting averages.

If target data was sensed at round 1, $\mathcal{ADV}$ has no way to succeed, since, by assumption, target data is never sensed by a sensor that is currently compromised. Similarly, if target data was sensed at round $\geq \lceil \frac{n}{k} \rceil$, there is no way to avoid $\mathcal{ADV}$'s success, since enough rounds have passed to have the adversary compromising the whole network.

Figure 5 and Figure 6 compare CoMAC and ExCo security, showing the survival probability of target data, as introduced in Section 6.

Figure 5 highlights the correctness of Equation 5. The x axis is the number of compromise rounds before the target data collection one, while the y axis is the expected survival probability. Fixed n, security performances of ExCo with t co-authenticators can be compared with CoMAC that involves $\nu(n,t)$ co-authenticators: in figure ExCo with $t = 1$ and $t = 3$ is comparable with CoMAC with $t = 3$ and $t = 15$, respectively.

Figure 6 compares the schemes showing 3d plots of the expected survival probability of target data: the x axis represents the number of co-authenticators (t), while the y axis is the number of compromise rounds ($\bar{r}$) before the target data collection one. The survival probability of ExCo is always greater than the one obtained by CoMAC, for the same configuration of t and $\bar{r}$. For example, given $t = 3$, survival probability of target data with ExCo is 0.8, even if $\mathcal{ADV}$ has compromised 90% of the sensors. Given the same ratio of compromised sensors, CoMAC only reaches 0.3 survival probability.
Even if $\mathcal{ADV}$ has compromised 90% of the network before target data is collected, ExCo with $t \geq 4$, provides a survival probability greater than 0.92, while CoMAC requires $t = 20$ to reach such high probability.

7.2 Robustness

Analysis and simulations of previous sections are valid for an *ideal* networks where messages between sensors are always delivered and sensors do not stop working for events such as malfunctioning or battery depletion. In this section, we try to analyze the proposed protocols in a *real* network where message delivery can fail with probability β and sensor can fail with probability γ.

Message failure. In an ideal network, 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 that would authenticate d_j^r, would be dropped by a factor of β, in both schemes. In the worst case, all messages sent by s_j at round r are not delivered, and d_j^r is only authenticated by s_j. In this scenario, for $\mathcal{ADV}$ to win, it only requires K_j^r to reach its goal with CoMAC, while with ExCo it requires K_j^r as well as any key K_q^r such that $s_q \in R_j^r$.

Sensor failure. In case of sensor failure, sensed data and MAC-s computed by the failed sensor are lost. In particular, assume that s_l fails at round r.

With CoMAC, for each round $r' < r$, each $d_j^{r'}$ such that $s_l \in S_j^{r'}$ looses one MAC. Also, the MAC-s computed on d_l^r by any $s_j \in S_l^r$ can be discarded by the sink, as data on which those MAC-s are computed will not be available.

With ExCo, at each round $r' < r$, one authentication tag of the value $d_q^{r'}$ sensed by $s_q \in \{S_j^{r'} \cup R_j^{r'} \cup T_j^{r'}\}$ will not be verifiable. Comparing ExCo and CoMAC, the higher security achieved by ExCo leveraging on the secret material of a larger set of peers, comes at a cost of lower robustness.

In realistic settings, if messages are not always delivered and sensors can fail, the sink might find MAC-s that lack of the corresponding data they where computed on, or collected data with $t' \leq t + 1$ MAC-s.

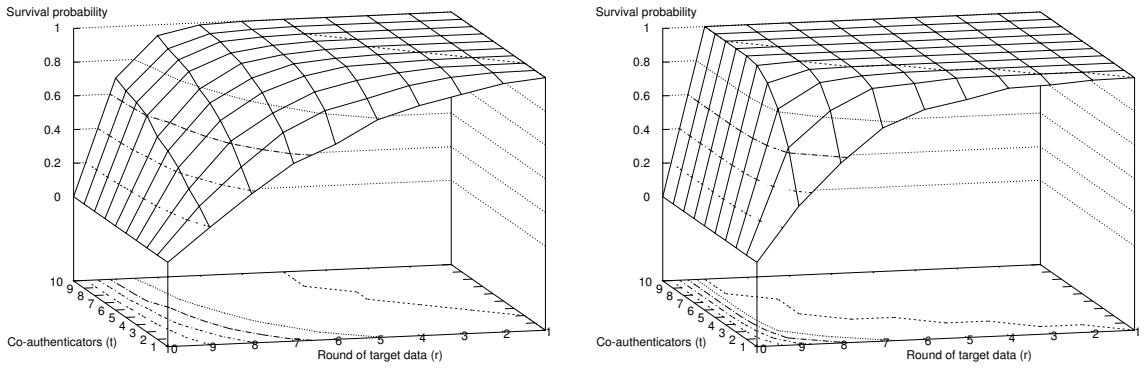
In the first case, MAC-s can be discarded. Indeed, if sensors become aware that any peer has failed (this mechanism is out of the scope of this paper, however this information could be inferred from the routing protocol [1]), they can delete from their storage all MAC-s computed on data collected by the failed sensor. In the second case, data authentication is still viable, but weakened. Indeed, we can consider that the number of MAC-s provides a natural degree of confidence on the authenticity of the data: the more the number of MAC-s, the more the effort the adversary needs to make to forge an authenticated value.

8. CONCLUSION

In this paper, we focused on a mobile adversary attempting to replace authentic data in UWSN-s. We proposed two techniques based on sensor cooperation that achieve much higher security than prior results, with little overhead. Both techniques rely only on symmetric cryptography. 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.

9. ACKNOWLEDGMENTS

Roberto Di Pietro is also with the UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, DEIM — Tarragona, Spain. Authors are the solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization.
This work was partly supported by The Spanish Ministry of Science and Education through projects TSI2007-65406-



(a) CoMAC (b) ExCo

Figure 6: Comparison between CoMAC and ExCo.

C03-01 "E-AEGIS" and CONSOLIDER CSD2007- 00004 "ARES", and by the Government of Catalonia under grant 2005 SGR 00446; the IST project SENSEI (Integrating the Physical with the Digital World of the Network of the Future, Grant Agreement Number: 215923). It was also supported in part by an award from the US Army Research Office (ARO) contract W911NF0410280 as well as a grant from the US Fulbright Foundation.

10. REFERENCES

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

[2] M. Bellare and S. Miner. A forward-secure digital signature scheme. In *CRYPTO*, 1999.

[3] M. Bellare and B. Yee. Forward Integrity for Secure Audit Logs. *UCSD CSE Dept. Technical Report 23*, 1997.

[4] M. Bohge and W. Trappe. An authentication framework for hierarchical ad hoc sensor networks. In *ACM Workshop on Wireless Security*, 2003.

[5] R. Di Pietro, D. Ma, C. Soriente, and G. Tsudik. Posh: Proactive co-operative self-healing in unattended wireless sensor networks. In *IEEE SRDS*, 2008.

[6] R. Di Pietro, L. Mancini, C. Soriente, A. Spognardi, and G. Tsudik. Catch me (if you can): Data survival in unattended sensor networks. In *IEEE PerCom*, 2008.

[7] R. Di Pietro, L. Mancini, C. Soriente, A. Spognardi, and G. Tsudik. Playing hide-and-seek with a focused mobile adversary. Cryptology ePrint Archive, Report 2008/293, 2008.

[8] Y. Dodis, M. Franklin, J. Katz, A. Miyaji, and M. Yung. A generic construction for intrusion-resilient public key encryption. In *CT-RSA*, 2004.

[9] Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-insulated public key cryptosystems. In *EUROCRYPT*, 2002.

[10] Y. Dodis, J. Katz, S. Xu, and M. Yung. Strong key-insulated signature schemes. In *PKC*, 2003.

[11] S. Ganeriwal, S. Capkun, C. Han, and M. Srivastava. Secure time synchronization service for sensor networks. In *WiSe*, 2005.

[12] G. Itkis. Intrusion-resilient signatures: Generic constructions. In *SCN*, 2002.

[13] G. Itkis and L. Reyzin. SiBIR: Signer-base intrusion-resilient signatures. In *CRYPTO*, 2002.

[14] D. Liu, P. Ning, S. Zhu, and S. Jajodia. Practical broadcast authentication in sensor networks. In *IEEE MobiQuitous*, 2005.

[15] D. Ma and G. Tsudik. Extended abstract: Forward-secure sequential aggregate authentication. In *IEEE S&P*, 2007.

[16] D. Ma and G. Tsudik. DISH: Distributed Self-Healing (in Unattended Sensor Networks). In *SSS*, 2008.

[17] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar. Spins: security protocols for sensor netowrks. In *MOBICOM*, 2001.

[18] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. A. Arbaugh. Toward resilient security in wireless sensor networks. In *ACM MobiHoc*, 2005.

[19] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical en-route filtering of injected false data in sensor networks. In *IEEE INFOCOM*, 2004.

[20] W. Zhang and G. Cao. Group rekeying for filtering false data in sensor networks: a predistribution and local collaboration-based approach. In *IEEE INFOCOM*, 2005.

[21] Y. Zhang, J. Yang, and H. T. Vu. The interleaved authentication for filtering false reports in multipath routing based sensor networks. In *IPDPS*, 2006.

[22] S. Zhu, S. Setia, S. Jajodia, and P. Ning. Interleaved hop-by-hop authentication against false data injection attacks in sensor networks. *TOSN*, 3(3), 2007.