S^2DCC: Secure Selective Dropping Congestion Control in hybrid wireless multimedia sensor networks

M. Tortelli[b], A. Rizzardi[a], S. Sicari[a], L.A. Grieco[b], G. Boggia[b,*], A. Coen-Porisini[a]

[a] *"DISTA, Dep. of Theoretical and Applied Science", Universita' degli Studi dell'Insubria*
v. Mazzini 5 – 21100, Varese, Italy.
[b] *"DEI, Dep. of Electrical and Information Engineering", Politecnico di Bari*
v. Orabona 4 – 70125, Bari, Italy.

Abstract

Thanks to the availability of miniaturized camera and microphones, nowadays Wireless Multimedia Sensor Networks (WMSNs) can sense and deliver audio/video signals from a target environment to remote analysis sites. Hence, new opportunity are disclosed for advanced applications in health care, surveillance, military, and traffic monitoring domains, to name a few. But, at the same time, due to the high volume of multimedia streams and the richness of information they bring, WMSNs incur critical issues in terms of congestion control, privacy, and security. These problems can be solved separately by adopting consolidated solutions conceived to address each of them. But one of the pivotal point of optimization in a Wireless Sensor Network (WSN) is the possibility of exploiting a cross layer design. To bridge this gap, an integrated solution is proposed hereby, namely Secure Selective Dropping Congestion Control (S^2DCC), based on end-to-end ciphering, in-network selective data dropping, scalable multimedia encoding, and hierarchical and hybrid network design. Moreover, an open source implementation of S^2DCC has been developed in the Castalia simulator. The main outcomes of the performance evaluation show that S^2DCC is able to

[*] Corresponding author: g.boggia@poliba.it

Email addresses: m.tortelli@poliba.it (M. Tortelli), alessandra.rizzardi@uninsubria.it (A. Rizzardi), sabrina.sicari@uninsubria.it (S. Sicari), a.grieco@poliba.it (L.A. Grieco), g.boggia@poliba.it (G. Boggia), alberto.coenporisini@uninsubria.it (A. Coen-Porisini)

meet data security and privacy requirements and to improve the quality of the received images at the sink with respect to state of the art solutions.

Keywords: WMSN, security, privacy, congestion control, image encoding.

1. Introduction

Wireless Multimedia Sensor Networks (WMSNs) are composed by devices able to acquire, process, and transmit both scalar data (i.e., temperature, humidity, pressure) and audio/video signals. They may represent an efficient solution in video surveillance, traffic and environmental monitoring, and telemedicine systems [1]. Nevertheless, their devices are usually constrained in terms of power and computational resources, thus requiring a careful cross layer design of the whole protocol stack.

A key aspect to consider is the greater traffic volume and burstiness typical of WMSNs with respect to classical WSNs. Therefore, the presence of a congestion control protocol is fundamental to limit the negative effects contributed by packet losses. Indeed, a lost packet not only worsen the quality of the reconstructed audio/video signal, but it also inflates energy consumption due to retransmissions. Several metrics can be monitored in order to detect the presence of network congestion, such as channel load [2], packet inter-arrival times [3], and local buffer occupancy [4] [5]. Different mitigation actions, as well, can be undertaken after a congestion episode is detected, including back pressure with rate control of neighboring nodes and prioritized packet dropping from transmission buffers (see Sec. 6 for a detailed review of related works). In this context, the adoption of scalable encoding techniques for multimedia contents plays a crucial role. In fact, such algorithms represent multimedia contents with different bitstreams that can be selectively added (or dropped) to increase (or decrease) data resolution. In this way, it becomes possible to deliver high quality multimedia contents in presence of a high available bandwidth, and to lower the resolution of transmitted signals as soon as a congestion happens or the channel becomes noisy. As a remarkable example, the Fully Scalable SPIHT (FS-SPIHT) image codec [6], based on the wavelet transform, has been used in [7] to design the so called Selective Dropping Congestion Control (SDCC). FS-SPIHT keeps the main features of the SPIHT, such as the Signal-to-Noise Ratio (SNR) scalability, and adds the *spatial scalability* without increasing the codec complexity. This feature allows the nodes in SDCC to selectively discard buffered packets based on their relative importance with respect to the Peak Signal-to-Noise Ratio (PSNR) of sensed images.

Another important issue in WMSNs is represented by privacy and security requirements. For example, in some applications such as video surveillance or telemedicine, the manipulation of data by malicious nodes, as well as unauthorized accesses to confidential information, must be prevented [8]. However, introducing privacy and security mechanisms able to guarantee data anonymity and integrity without compromising the performance of the network itself (e.g., energy efficiency and end-to-end delay) is a difficult task.

It is worth to remark that many approaches have been proposed in the WSN context to face congestion control, privacy and security (as detailed in Sec 6), but no solutions currently address these challenges in WMSNs. To bridge this gap, a *secure* extension of SDCC, namely S^{2}DCC is proposed hereby. To highlight the novelty of the present work, it is important to clarify the contributions of S^{2}DCC with respect to SDCC; they can be summarized as follows:

1. The use of an end-to-end ciphering model which makes the proposed WMSN architecture suitable for application domains that require sensitive information to be encrypted.
2. The adoption of a hierarchical and hybrid WMSN architecture, where sensor nodes are grouped in clusters, with one mesh router that can communicate with the sink or with other mesh routers. This architecture permits a distribution of tasks among the network nodes according to their capabilities. In this particular case, sensor nodes, assumed with limited resources, are responsible for only the sensing, ciphering and transmitting tasks, whereas mesh routers, assumed to have enough power and com-

putational resources, are responsible for executing the congestion control algorithm, as well as privacy and integrity verification operations.

3. The capability to counteract the activity of malicious nodes, which can violate the integrity and/or the anonymity of the data collected by the network.

The performance of S^{2}DCC have been thoroughly compared with respect to SDCC by extending the well known (and largely adopted in WSN simulations) Castalia simulator [9]. Results show that the proposed solution is able to improve the quality of the reconstructed images at the sink, evaluated in terms of PSNR and image size, thanks to its resilience to malicious nodes, while guaranteeing almost the same energy consumption of SDCC.

The rest of the paper is organized as follows: in Sec. 2, the characteristics of the privacy model adopted in S^{2}DCC are explained; in Sec. 3, an exhaustive background on the FS-SPIHT codec and on the SDCC algorithm is presented; while, in Sec. 4, the novelties introduced in S^{2}DCC are thoroughly described. A performance evaluation is reported in Sec. 5. After having discussed related work in Sec. 6, conclusions and future work are drawn in Sec. 7.

2. Privacy model

The privacy model used in this work extends the one presented in [10], where specific tasks are assigned to each network entity. Figure 1 reports the class diagram related to this schema. Class *Role* characterizes nodes with respect to privacy, and it is extended by three distinct classes representing the roles a node can play : *Subject*, *Processor* and *Controller*. Instead, class *Function* indicates which task is executed by a node of the network.

More specifically, in the UML schema of Figure 1 three macro elements can be distinguished: *Node*, *Data*, and *Action*. To map the different kinds of devices in the proposed architecture (see also Sec. 4), the entity *Node* includes both *Sensors* and *Mesh Routers*. The distribution of the different tasks among such nodes is a key aspect in S^{2}DCC. With reference to the class *Role*, sensor nodes

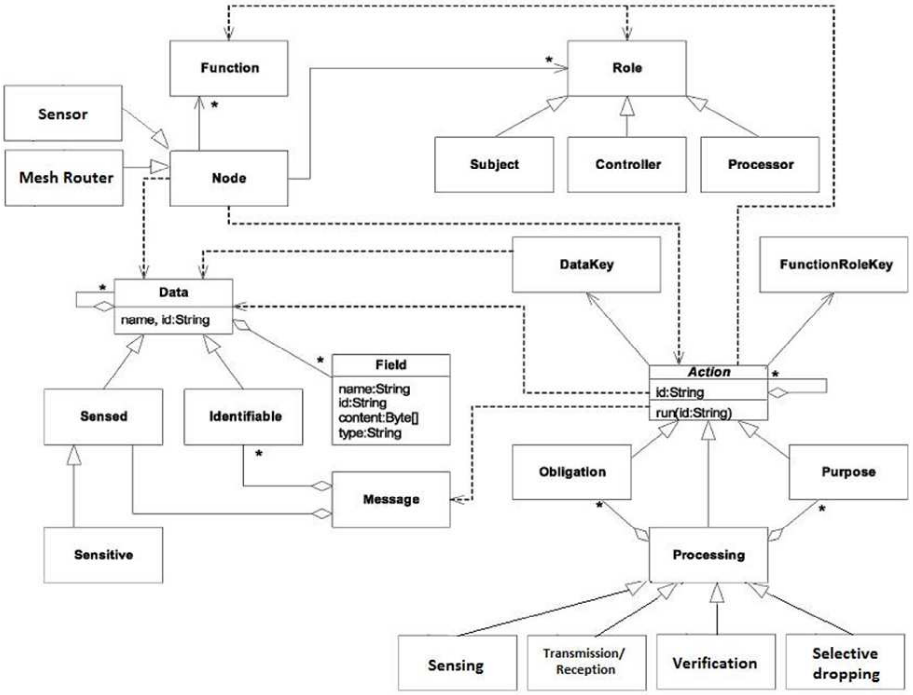

Figure 1: Diagram of network privacy classes.

act both as *Subject* and *Processor* because they acquire sensed data from the environment (so that, they take the role of *Subject*), and they also execute some actions on data, such as encryption, transmission and forwarding (so that, they take the role of *Processor*). On the other side, each mesh router can take the role of *Subject*, e.g., when it generates a new packet after the execution of the congestion control algorithm, and the role of *Processor* or *Controller*, when it executes transmission or integrity checks on the received data, respectively.

The class *Data*, instead, represents the information related to the type of element processed by *Processors*. In particular, it is extended by class *Identifiable* and class *Sensed*; the former includes the information useful to identify nodes, such as their identifier. The latter represents the image that is sensed by the nodes within the network. The instances of class *Sensed* can be also

Table 1: Node actions

Action	*Description*
Sensing	the acquisition of data by a node, concerning a specific scope
Transmission	the sending of a message containing the image sensed by the current node
Reception	the reception of a message sent by another node of the network
Forwarding	the sending of a message previously received by a node
Selective Dropping	the selective dropping performed by mesh routers in case of network congestion
Verification	the checks performed by mesh routers regarding the integrity of the data contained in a received message

instances of class *Sensitive* when the acquired information requires particular access controls (e.g., in case of health domain, an image related to the diagnosis of a patient). Furthermore, an instance of class *Data* can be a more complex structure, composed of basic information units, called *Fields*, and aggregated into instances of class *Message*. Class *Message* represents the basic communication unit exchanged by the nodes and it may contain both *Identifiable* and *Sensed* data, as it will be shown in Sec. 4.

Finally, class *Action* consists of the operations performed by a node, including sensing, transmission, reception, forwarding, selective dropping, and verification, which are the most common actions in a WMSN, as specified in Table 1. Class *Action* is extended by classes *Obligation*, *Processing* and *Purpose*. Since the processing activity is executed under a purpose and an obligation, *Processing* specifies an aggregation relationship with *Purpose* and *Obligation*. As regards obligations, it may be useful to associate *Verification* with one or more of them, in order to model the fact that, upon inconsistency detection, some countermeasures (such as sending an error/alert message to the sink) have to be undertaken.

To guarantee security and privacy of the transmitted data within the WMSN, the requirements, needed to protect communications among authorized nodes, have to be defined. Only authorized nodes can access data and/or execute actions; hence, the model presented in Figure 1 defines some encryption mechanisms. Two classes, representing encryption keys, named *DataKey* and *Func-*

Table 2: Keys schema

Key	*Use*	*Class*
Sensing-Subject *(SS)*	data encryption, ensure the data confidentiality	*DataKey, FunctionRoleKey*
Transmitter-Processor *(TP)*	node ids encryption, ensure the anonymity of communications	*FunctionRoleKey*
Hash-Subject *(HS)*	data integrity verification at mesh routers level	*DataKey, FunctionRoleKey*

tionRoleKey, are introduced. The former is used to protect the data content of messages, and therefore each node of the network owns a (possibly) different *DataKey* to encrypt the data content of its messages. The latter is used to guarantee that messages communication and data handling are executed only by authorized nodes. To fulfill these goals, each action is associated with a function-role pair and, therefore, it requires a given *FunctionRoleKey* in order to be executed. Thus, only nodes owning the corresponding *FunctionRoleKey* can execute a given action. Since a node may play different functions and roles, it may own more than one function-role key.

An end-to-end security scheme is adopted, thus only the sink is allowed to decipher the encrypted data previously acquired by sensor nodes. As a result, having the mesh routers the role of *Controller*, they are able to execute the integrity checks on received data, without having the need to decipher them. To achieve this goal it is necessary to determine a scheme of keys assigned to the network entities. In details, each sensor node and each mesh router own three types of keys characterized by the couple function-role, as shown in Table 2.

Since the present work focuses on securing the selective dropping protocol proposed in [7], the key distribution phase is not considered at this stage because it is orthogonal to the S^{2}DCC design. It should take place in an earlier phase of network configuration, by means of well-known mechanisms discussed in [11]. The same considerations apply to key rotation as well as key revocation [12]. However, it is important to note that whereas the sink owns all the SS and HS keys to decrypt the data acquired by sensor nodes, the mesh routers have only the HS keys belonging to the nodes of their cluster. Thanks to this strict distribution of tasks and keys, the privacy model presented hereby is capable of providing end-to-end security and guaranteeing not only the integrity of transmitted data, but also their anonymity, even in presence of malicious nodes, since both the nodes ids and the data are encrypted by means of different keys. The verification is executed at mesh routers level because it allows to quicken the identification of malicious behaviors. Note that mesh routers are able to apply the selective dropping algorithm without accessing the clear content because the verification is performed by means of the HS key and not the SS key. In this way, any requirement on the the trustworthiness of mesh routers can be relaxed, since they do not own the key to decrypt the data.

2.1. Threat model

As regards the threat model, we consider that sensor nodes can be deployed in unsafe environments. They are assumed to have the same communication ranges and the size of the packets exchanged among nodes is fixed.

2.1.1. Eavesdropping and masking attacks

Each node owns three keys, namely: SS key, HS key, and TP key, which are pre-distributed to sensor nodes, before the network activity starts. SS and HS keys are responsible for guaranteeing the integrity and the confidentiality of the sensed and transmitted data, while TP key is responsible for preserving data confidentiality (i.e., the node ids are not transmitted in clear). Therefore, both the eavesdropping and masking attacks are counteracted by means of a two-level encryption approach.

More in details, the transmitted data are firstly encrypted with the node SS key and, then, a hash of the obtained result (i.e., the encrypted data) is further encrypted with the node HS key. As a consequence, if a malicious entity wants to know the packet content or inject false information into the network, then it should operate a brute force attack operating on the transmitted packets. However, the CHs check, for each packet arrived from the sensor nodes, the compliance of the encrypted hash with respect to the encrypted data. Such a verification is made by re-calculating the encrypted hash on the encrypted data, contained in the arrived packet, and making a comparison between the obtained result and the encrypted hash, included in the arrived packet. If the encrypted hash or the encrypted data or both are compromised, then the verification executed by the CHs will not succeed and the packet will be discarded from the network. We remark that CHs manage only encrypted data (i.e., they do not know the clear content of packets).

2.1.2. Replay attacks

The hash calculation includes the current timestamp (i.e., the instant time of packet generation), thus preventing replay attacks. Note that the sensor nodes are not synchronized (i.e., they do not have perfect clocks and they does not generate packets at the same instant time). Such an aspect along with the different keys, owned by sensor nodes, cause that the same event (e.g., the detection of a particular condition) reported by different nodes will generate diverse encrypted hash. Hence, this situation increases the effort to be executed for successing a brute-force attack.

Summarizing, we mainly consider the tampering, eavesdropping and replay attacks from outsider nodes. S^{2}DCC does not address the detection of malicious behavior towards the routing protocol or the network resources (e.g., denial of service attack). To solve that issue, it would be better to adopt a reputation system, able to evaluate the node behavior and the networks attacks on the basis of well-defined features (e.g., analyzing the number of packets generated or forwarded by each node), as presented by the authors in [13] for WSN scenario.

3. Background on SDCC

3.1. FS-SPIHT Codec

The *spatial scalability* represents the added value of the FS-SPIHT algorithm [6] with respect to the SPIHT codec [11]. By means of the spatial correlation among the subbands of a wavelet transformed image, it is possible to logically organize the wavelet coefficients in a hierarchical tree (see Figure 2). That is, coefficients are grouped in squared groups of four, starting from the upper-level of the tree, and they can be identified as top-right, bottom-right or bottom-left (i.e., their position inside the square). Each coefficient, except for the top-left one in the uppest level, has four children belonging to four different subbands. Due to the strong correlation among the wavelet coefficients, the four children are highly likely to be zero if their parent is equal to zero.

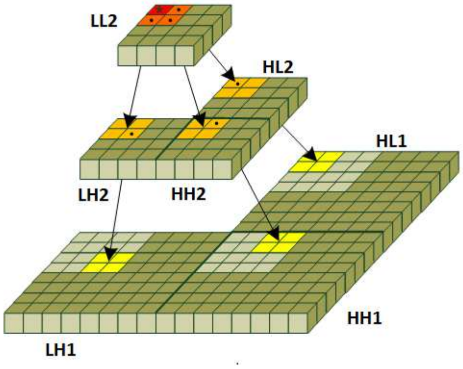

Figure 2: Father-child dependencies in a spatial orientation tree.

Theoretically, a wavelet transform with N levels can generate a maximum number of spatial resolution levels (RLs) equal to $k_{max} \equiv N + 1$. Starting from the resolution level k, it is possible to augment the resolution level of one step (i.e., to reach the resolution level $k - 1$), by simply adding the three subband at the level k. Accordingly, the RL $k = 1$ (respectively k_{max}) corresponds to the full sized (respectively smallest) image. Considering the example in Figure 2, an increment from level two to one needs the addition of subbands $LL1$, $HH1$ and $HL1$.

Combining the aforementioned *spatial scalability* with the *quality scalability*, a different weight can be assigned to each bit. In particular, the wavelet coefficients can be represented over several bitplanes (BPs), thus enabling an incremental transmission, from the most to the least significant ones, based on the network condition.

These characteristics are reflected in the bitstream structure reported in Figure 3, formed by a general header, containing information such as the original image pixel size, the parameters k_{max} and n_{max}, where n_{max} is the maximum number of *BP*s generated by the wavelet transform. The magnitude of the wavelet coefficients is used to calculate the n_{max} parameter, according to the formula:

$$n_{max} = \left\lfloor log_2 \left(\max_{\{(i=1..h, j=1..w)\}} \{|c(i,j)|\} \right) \right\rfloor , \tag{1}$$

where h and w are the height and the width of the transformed image, and $c(i, j)$ is the single transformed coefficient identified by the coordinates i and j.

Apart from the general header, the bitstream is composed by a sequence of bitplanes, ranging from n_{max} to 0, that, in turn, contains the related sequence of resolution levels.

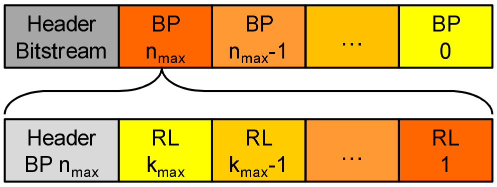

Figure 3: FS-SPIHT bitstream structure.

For example, a half-sized version of the original image can be obtained by removing the *RL*1 from each bitplane; in addition, the quality of the decoded image, and thus its PSNR, can be reduced by removing some of the less significant bitplanes.

In SDCC [7], each *BP*-*RL* pair represents a *bitstream segment*, i.e., the information unit. Therefore, considering an integer number of bitstream segments, the image distortion (in terms of PSNR, computed as in [6]) can be characterized by a two dimensional space, as reported in Fig 4. It specifically shows the PSNR of the *Lena* 512x512 grayscale image (similar results are obtained for other benchmark sequences) as a function of used *RL*s and removed *BP*s. It can be noticed that the highest PSNR value is obtained by using all the *BP*s with the smallest image (remember that $RL\ k_{max}$ corresponds to the smallest image). On the contrary, the PSNR decreases by increasing both the image size and the number of removed BPs.

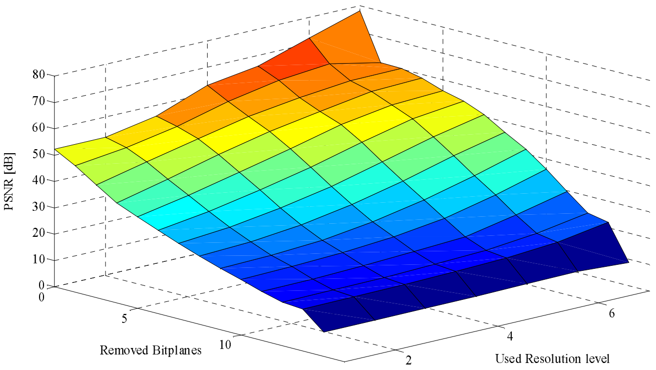

Figure 4: Image distortion at various decoding precisions.

For what concerns the specific wavelet transform, the well known *Cohen-Daubechies-Feauveau 9/7* (CDF 9,7) wavelet has been used herein, in accordance to [7]. It has been implemented through the *Lifting Scheme* [14], in order to reduce the computational complexity and to make it suitable for heavily constrained devices.

3.2. SDCC algorithm

The SDCC algorithm [7] takes advantage of the packet classification introduced by the FS-SPIHT to implement congestion control functionalities. In SDCC, each node can detect a congestion episode, based on the local transmission buffer occupancy. To avoid buffer overflows and to mitigate their negative effects, SDCC requires each node to selectively drop enqueued packets, chosen accordingly to their importance on the PSNR of the reconstructed image [7]. Accordingly, SDCC acts as a cross layer algorithm, which accounts for application layer requirements (PSNR) to face congestion episodes at the network layer.

With SDCC, the time is seen as an endless sequence of timeslots. To establish the number of packets to be selectively dropped at each slot, a simple discrete time controller is used [10] (see Figure 5). In the discrete-time domain, the dynamics of the transmission queue lenght can be modeled by the following linear model:

$$\varphi(t_{j+1}) = \varphi(t_j) + \xi(t_j) - u(t_j) + u_V(t_j) \tag{2}$$

With reference to the j-th sampling interval, $\xi(t_j) \geq 0$ represents the number of packets in the transmission buffer, while $u(t_j) \geq 0$ is the number of transmitted packets. They both affect the buffer occupancy in such a way that they cannot be evaluated in advance, i.e., they are modeled as disturbances.$\varphi(t_{j+1})$, which is the queue length associated with the transmission buffer at the time instant t_{j+1}, is also influenced by the result of the PI control action, i.e., $u_V(t_j) \leq 0$, whose absolute value represents the number of packets that it needs to be dropped, taking as a reference the target queue level φ_T.

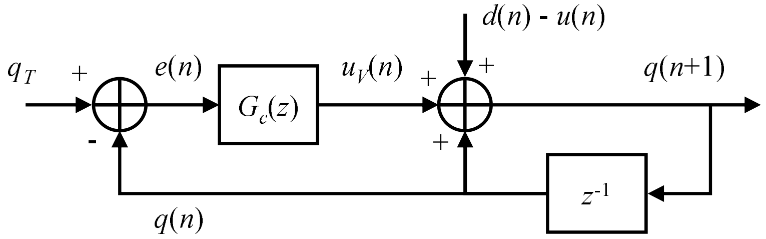

Figure 5: Block Diagram of the controlled system.

The transfer function of the PI controller[1], chosen due to its ability to filter out the continuous component of the disturbance $\xi(t_j) - u(t_j)$, is computed as the $\mathcal{Z}$-transform of its discrete-time pulse response, and it is given by:

$$G_C(z) = K \cdot \left(1 + \frac{1}{T}\frac{z}{z-1}\right), \tag{3}$$

where K and T are non negative constants.

Remember that in SDCC each bitstream segment is mapped into a single network packet, identified by a single *BP-RL* pair (more packets are created if the segment size exceeds the maximum packet size). Therefore, when a conges-

[1] The reader is referred to [7] for the tuning rules of the PI controller.

tion is detected, $|u_V(n)|$ packets can be easily dropped from the transmission buffer by following a *zig-zag-like* prioritized order, as reported in Figure 6. In this manner, packets with the minimum impact on the PSNR of the reconstructed image will be dropped at first; then, the zig-zag order will allow to select other packets in order to simultaneously reduce both the image size and its quality.

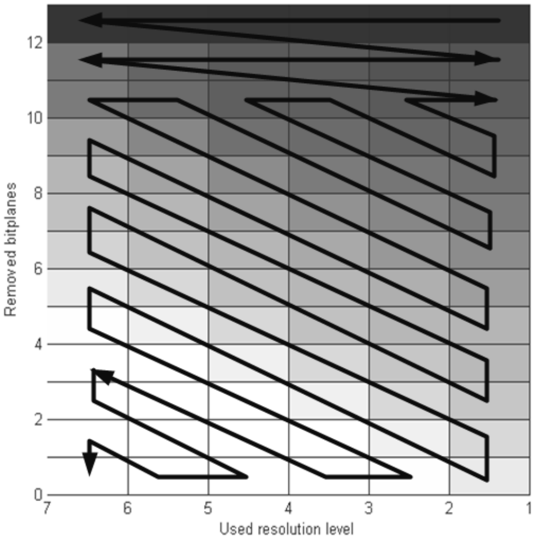

Figure 6: Zig-Zag Scanning Order.

At the sink side, received packets are logically mapped into a matrix with size n_{max} x k_{max} in order to optimally decode the respective image. An example is reported in Figure 7; it can be noticed that the FS-SPIHT decoder needs to select a “compact” (without missing segments) sub-rectangle having its bottom left corner coincident with the matrix one. So that, it is evident that a further scope of dropping packets following the zig-zag scanning order is the minimization of the probability to have holes in the middle of the matrix.

One of the criteria that can be used to implement an optimal decoding is the number of bits, because a slightly increase in the image quality is reached for every single bit that is added. The search problem of finding the biggest rectangle in terms of number of bits can be solved using a binary tree-like data

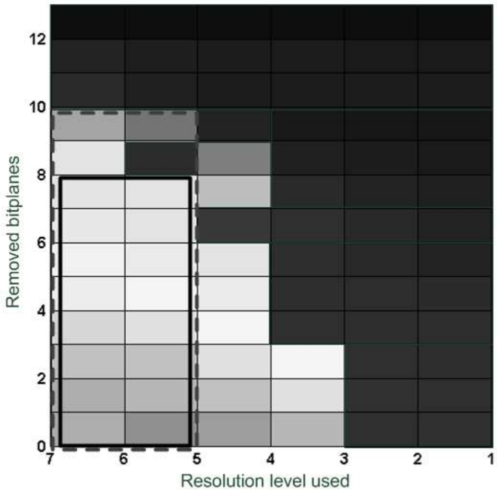

Figure 7: Ex. of packets available at the sink; the dark rectangles refer to packet not available.

structure and navigating it *breadth-first*. To summarize, a schematic overview of the SDCC approach is pictured in Figure 8.

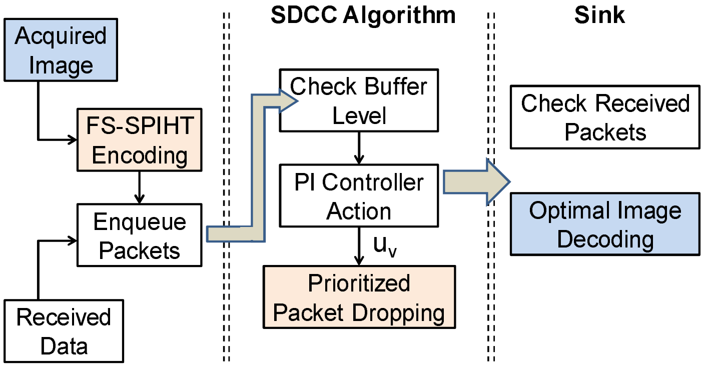

Figure 8: Overall view of the SDCC algorithm.

4. From SDCC to S^{2}DCC

After having extensively described both the scalable encoding technique and the main points of the congestion control used in SDCC, the novelties introduced by its new secure version S^{2}DCC are presented herein.

4.1. Hierarchical architecture

In S^{2}DCC a hierarchical architecture is proposed, as shown in Figure 9. Sensor nodes are grouped in clusters; each cluster is connected to a mesh router, which plays the role of *cluster head* (CH). In turn, CHs can be connected through a high speed wireless mesh backbone to other CHs or directly to the sink. It is assumed that each node is aware of its own geographical position and implements a CSMA-like MAC protocol. Moreover, the nodes implements the Multipath Rings Routing [15] algorithm, in which each sensor node, or mesh router, knows only its rank (i.e., its distance from the sink). At each transmission a node/mesh router will send the message to one of its parent nodes, selected randomly every time. Such a routing protocol mitigates the action of malicious nodes, avoiding the creation of static paths where malicious nodes could easily act, for example altering the normal forwarding of packets, thus generating additional unnecessary traffic. Furthermore, since at each hop a packet is sent only to one parent node, the network overhead and congestion are remarkably reduced with respect to other existing routing algorithms [15]. The hierarchical architecture allows to differentiate tasks executed by nodes according to their capabilities. That is, sensor nodes in S^{2}DCC are only responsible of acquiring images, encrypting, and sending them to the respective CH. Otherwise, CHs can execute the packet dropping algorithm, in addition to all the operation needed to verify the integrity of the received data. Indeed, they are also responsible of informing the sink about possible violation that may occur within the network. Hence, S^{2}DCC introduces a new complete protocol for data integrity verification and privacy preservation. In the following sections, the various parts of the proposed protocol are described in detail.

4.2. Message structure

In S^{2}DCC, a message is the basic application unit exchanged by the nodes within the network. Each message refers to a single transmission hop between adjacent nodes. To satisfy the end-to-end security issue, some information contained in the packets are encrypted by using Message Digest MD5, algorithm

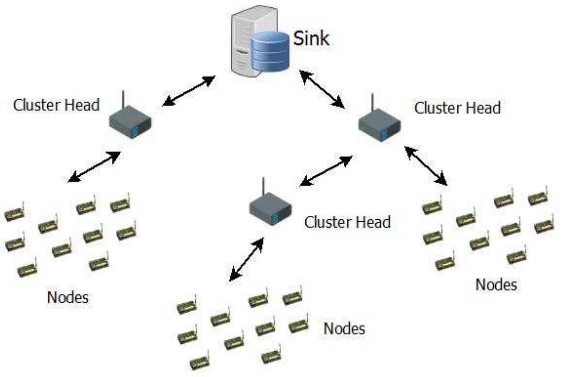

Figure 9: Network model in S^{2}DCC.

with proper keys. MD5 has been adopted, which has been proven to be suitable for WSN in terms of memory usage and resource power consuming [16], for example with respect to SHA and AES, which require more computational effort. A message $m_{n,q}$ is defined as a tuple in which all fields, unless otherwise specified, are ciphered.

$m_{n,q}$= ($s_{n,q}$, id_I,D, H_c, BP, RL, S_N, S_T, d_m, L_n, e, $\chi_{n,q}$), where:

- $s_{n,q}$ is the couple (n_s, q_s) or (r_i, q_s). n_s, or r_i, identifies the *Subject* sensor node, or the CH, respectively, which either sensed or transmitted the image after the congestion control; q_s identifies such a message among those transmitted by n_s or r_i. In case of error notification, r_i identifies the CH that discovered and generated the error. Note that this field is kept unchanged among transmissions.
- id_I is the unique identifier for the image transmitted over the network.
- D is the sensed or transmitted image.
- H_c is the hash of the field D used by a CH to verify the integrity of the received image (see Section 2.1).
- BP is the bitplane associated to the bitstream segment of the current image D.

- RL is the resolution level associated to the bitstream segment of the current image D.

- S_N is a sequence number which identifies the current segment in case of fragmentation.

- S_T is the total number of segments in case of fragmentation.

- d_m is the size of the image in bytes, without considering the headers of the message.

- L_n is a list of the nodes which forwarded the image towards the sink, and it is updated each time a node forwards a packet. Such a field is used by the sink and the CHs, respectively, to know which keys to use to decrypt the content of the packet or to do the security checks.

- e is set to 1 if an error is detected, otherwise it is equal to 0.

- $\chi_{n,q}$ is the couple $(n_s,\, q_s)$ that, in case of error notification, identifies the node that either sensed or transmitted the correct image and the identifier of the message transmitted by such a node, which correspond to $s_{n,q}$.

Note that fields e and $\chi_{n,q}$ are set only in case of error notification, while the encryption of the fields related to nodes, such as $s_{n,q}$ and L_n, helps to guarantee data anonymity. According to the model defined in Sec. 2, the fields D, H_c, id_I, BP, RL, S_N and S_T are instances of class *Sensed*; $s_{n,q}$, $\chi_{n,q}$, and L_n are instances of class *Identifiable*; d_m and e are instances of class *Data*.

In relation to the message structure, the following protocols will be detailed in the next sections:

1. *Sensing*, which defines the actions carried out by nodes when acquiring images from the environment.
2. *Integrity Verification*, which describes the actions carried out by CHs when receiving a message, in order to check the integrity of the received data and to reveal malicious behavior.

3. *Selective Dropping Algorithm*, which defines the process carried out by CHs in case of congestion detection.

4.3. Sensing

Let n be a node sensing an image d, uniquely identified by the identifier id, from the environment where it is located. According to the function-role classification, when sensing d, the node acts as a *Sensing-Subject* and, therefore, it encrypts the image d using the corresponding key $k_{n,SS}$. The node calculates the hash of the encrypted field D and encrypts the obtained result with its own hash key $k_{n,HS}$. This key is also used to encrypt other relevant information regarding the images, such as the bitplane BP, the resolution level RL, the sequence number S_N, and the total S_T. Thus, the message $m_{n,q+1}$ is arranged according to the structure discussed in the previous section and is queued in the node's transmission buffer. When preparing the message, the node acts as a *Transmitter-Processor* and, therefore, all the cyphered fields that are not related to the acquired image are encrypted using the key $k_{n,TP}$. ϵ represents an empty field, instead $E^*_c\{\cdot\}$ is an encrypting function [17]. The values assigned to the fields of the message are:

$$
\begin{aligned}
s_{n,q+1} &= (E^*_c\{n, k_{n,TP}\}, E^*_c\{q+1, k_{n,TP}\}) \\
id_I &= E^*_c\{id, k_{n,SS}\} \\
D &= E^*_c\{d, k_{n,SS}\} \qquad H_c = E^*_c\{\text{hash}(D,\ k_{n,HS},\ t_n)\} \\
BP &= E^*_c\{BP, k_{n,HS}\} \qquad RL = E^*_c\{RL, k_{n,HS}\} \\
S_N &= E^*_c\{S_N, k_{n,HS}\} \qquad S_T = E^*_c\{S_T, k_{n,HS}\} \\
d_m &= \text{Image size in Bytes} \qquad e = 0; \qquad \chi_{n,q} = \epsilon; \\
L_n &= (E^*_c\{n, k_{n,TP}\}, E^*_c\{q+1, k_{n,TP}\})
\end{aligned}
$$

4.4. Integrity Verification

Let $m_{n,q}$ be a message received by a CH. The message is analyzed to find out whether an integrity violation has been performed. More specifically, the CH runs the following steps:

1. It calculates the hash of the field D of the received message.
2. Then, it encrypts the output with the hash key, $k_{n,HS}$ of the nodes that generated the message.
3. If the obtained result matches the field H_c of the received message, a security violation has not occurred; the image can be encapsuled in a new packet $s_{r,q}$ and can be submitted to the selective dropping algorithm in case of traffic congestion.
4. Otherwise, if there is not a positive matching, the received message should be considered as corrupted, and, therefore, the CH has to transmit over the network an error notification message structured as follows.

$$
\begin{aligned}
s_{r,q+1} &= (E_c^*\{r, k_{n,TP}\}, E_c^*\{q+1, k_{n,TP}\}) \\
id_I &= E_c^*\{id, k_{n,SS}\} \\
D &= \epsilon \qquad H_c = \epsilon \\
BP &= \epsilon \qquad RL = \epsilon \\
S_N &= \epsilon \qquad S_T = \epsilon \\
d_m &= \epsilon \qquad e = 1 \qquad \chi_{r,q+1} = m_{n,q}.s \\
L_n &= m_{n,q}.L_n \cup (E_c^*\{r, k_{n,TP}\}, E_c^*\{q+1, k_{n,TP}\})
\end{aligned}
$$

where $s_{r,q+1}$ is set to the mesh router r which detected the image violation, while $q+1$ represents the identifier of the message among those sent from the mesh router itself. $m_{n,q}$. Note that field $s_{r,q}$ identifies the CH that found the error (i.e., node r_i); field $\chi_{r,q}$ equals the content of field $s_{n,q}$ of the original received message; e is set to 1 to indicate that the current message is an error notification. The fields D, H_c, BP, RL, S_N, S_T and d_m are left empty, unlike

Pseudocode 1 Selective Dropping Algorithm

```
Input: $SamplingTime(s_t),\ q_T,\ PIControllerConstants(K, T)$
for all $s_t$ do
    $q(n) \leftarrow CurrentBufferSize$
    $e(n) \leftarrow q_T - q(n)$
    $Calculate\ u_V$
    if $u_V >= 0$ then
        $u_V \leftarrow 0$
    else
        $Retrieve\ necessary\ hash\ keys\ k_{n,HS}$
        $Exclude\ packets\ with\ e = 1\ from\ u_V$
        $Map\ packets\ according\ to\ their\ importance$
        $Selective\ Drop\ (u_V)$
    end if
end for
```

the field id_I which is preserved. Finally, the new message is queued in the transmission buffer.

When the sink receives the error notifications sent by the cluster heads, it may execute some analysis relating to the L_n field, in order to identify the malicious nodes.

4.5. Selective Dropping Algorithm

After having verified the integrity of the received message, a CH can proceed by inserting $m_{n,q}$ in its transmission buffer. Depending on traffic situation, the CH may execute the selective dropping algorithm only on no violated messages, since error notifications, characterized by the field e set to 1, must be always sent to the sink. Starting from the description of the SDCC algorithm presented in section 3.2, the entire procedure of the selective dropping in S^{2}DCC is reported in the following Pseudocode 1.

At each iteration, the dropping phase is started only if $u_V < 0$, i.e., more packets than the target threshold are inside the transmission queue. Then the

CH needs to access to the BP and RL information of the packets present in its queue, in order to logically place them inside the matrix BP-RL (according to their impact on the PSNR of the reconstructed image). To achieve such information, the CH retrieves the hash key $k_{n,HS}$ of the nodes that generated packets in the transmission queue and it computes $bp_n = D_c^*\{bp_n, k_{n,HS}\}$ and $rl_n = D_c^*\{RL_n, k_{n,HS}\}$, with D_c^* representing a decrypting function. At this point, it can start the selective dropping of the $|u_V|$ packets, following the zig-zag-like scanning, and starting from the least significant BP-RL pair, i.e., the one on the top-right in Figure 6. Note that, because of the fragmentation, more than one packet can belong to the same BP-RL pair; the total number of packets to be dropped should always be $|u_V|$.

5. Performance Evaluation

To evaluate the performance of S^{2}DCC algorithm, the Castalia simulator based on Omnet++ [9] has been extended with new features that allow the simulation of all the facets of S^{2}DCC itself. The performance of S^{2}DCC have been also compared with respect to other two protocols. The first is SDCC, which adopts a flat architecture where each sensor node executes the selective dropping, and no security or privacy issues are addressed. The second, named No-SDCC, adopts a flat architecture, but neither the selective dropping nor security countermeasures are taken.

The parameters used for the simulated scenarios are summarized in Table 3. First of all, the behavior of S^{2}DCC, SDCC, and No-SDCC is evaluated using two topologies, where the number of sensor nodes varies in order to reproduce different network loads. The data rate of the sensor nodes also varies; in particular, each node is supposed to implement a CSMA/CA like MAC protocol, with a data rate equal to 21 or 56 Mbps, and a maximum packet size of 4096 bytes. The aforementioned values are in accordance with the IEEE 802.15.3 standard. In our simulation scenarios, the sensor nodes are supposed to have the same communication range and almost the same initial battery supplies. For all the simulation results presented, an average of 30 simulation runs have been executed. As regards our attack scenario, the number of no-malicious nodes is fixed during simulations, whereas the number of malicious nodes is increased with a percentage of 10% to 40% with respect to the well behaving ones. Furthermore, to simulate a video-surveillance scenario, each sensor node is supposed to acquire images at one frame per second. The K and T constants in the PI controller are set to 1 and 90, respectively, according to the analysis presented in [7]. In fact, to grant system stability, the K parameter is constrained in the range (0, 2), while the parameter T can vary over a wider set of values. With respect to the clustering technique, the sensor nodes are distributed in a uniform way among the three clusters in order to balance the load of the network.

The comparison among S^{2}DCC, SDCC and No-SDCC has been done monitoring the following performance indices:

- *Packet Loss Ratio*: two types of losses are analyzed. First, simple lost packets, when packets are lost due to collisions, congested links, and so on. Second, selectively dropped packets, when a packet is dropped from the transmission queue of a node because of the action of the selective dropping algorithm, which is executed by the cluster heads in S^{2}DCC and by each sensor node in SDCC. In the No-SDCC scenario, only simple lost packets can be monitored.

- *Received Messages*: it expresses the received messages at the sink level in S^{2}DCC. In particular, a packet received by the sink can be classified as violated (i.e., in case of an error notification due to integrity checks) or not violated. The number of violated packets with respect to the not violated ones varies in relation to the percentage of malicious nodes in the network.

- *Delay*: it is the time elapsed between the packet generation at a sensor node and its reception at the sink.

- *Quality of Received Images*: the quality of the images received by the sink

Table 3: Simulation parameters

Param.	*Description*	*Topology* 1	*Topology* 2
N	Number of nodes	25	50
C	Cluster Number	3	3
D_c	Depth of connections	3	5
M	Percentage of malicious nodes	0% 10% 20% 30% 40%	0% 10% 20% 30% 40%
P	Interval time of data generation	1s	1s
P_{cKmax}	Max Packet size	4096 bytes	4096 bytes
br	Bit rate	21 Mbps, 56 Mbps	21 Mbps, 56 Mbps
K	Constant of PI controller	1	1
T	Constant of PI controller	10:100	10:100
Q_m	CH buffer size	5000 messages	5000 messages
S_m	CH percentage of buffer size emptying (S^{2}DCC)	95%	95%
Q_n	Node buffer size	2000 messages	2000 messages
S_n	Node percentage of buffer size emptying (SDCC)	95%	95%
t_S	Duration of simulation	200 s	200 s

is expressed by both their PSNR and by their size (i.e., number of pixels).

- *Power Consumption*: it represents the power consumption of the sensor nodes. It is quantified by using the *Resource Manager* offered by the Castalia simulator itself, which considers both the power consumption of a node in several states (i.e., transmission, reception and idle) as well as the power consumption due to the computational load of encryption and selective dropping operations. Note that the power consumption during sensing has not been measured because it is negligible compared with the total consumption of the sensor nodes.

With respect to the protocol overhead, the following contributions have been considered:

- The overhead generated by the error notifications sent by the CHs to the sink in case of violation attempts; such a kind of information is reported, later on, in Figures 14 and 15.

- The overhead in the packet size of S^{2}DCC with respect to the original

SDCC protocol; it includes the nodes list L_n (whose length depends on the number of hops a message goes through to reach the sink. The size of this field can be easily upper bounded based on the maximum distance (in number of hops) from a sensor to the sink, namely k. Accordingly, it is less than k x 8 bits. In addition to the L_n there is the field e which points out if a message is violated or not (8 bits), the field $\chi_{n,q}$ used in case of error notification (8 bits), and the hash (128 bits). Summarizing, the total overhead is less than (144 + k x 8) bits.

- The overhead of the memory storage of sensor nodes; only three 128-bits length keys are required in order to run the S^{2}DCC algorithm (i.e., sensing-subject, transmitter-processor, hash-subject). Moreover, in terms of power consumption, note that sensor nodes execute encryption operations only, whereas CHs perform both encryption and decryption, thus reducing the overall consumption of sensor nodes. In fact, as will be shown in Figure 16, S^{2}DCC introduces only a slight increase energy consumption with respect to the original SDCC algorithm.

5.1. Simulation results

5.1.1. Packet loss ratio

The Packet Loss Ratio is reported in Figures 10 and 11. A strict correlation between the increase of the malicious nodes and the increase of the selectively dropped packets is evident, in both topologies and for both data rates in S^{2}DCC and SDCC scenarios. What firstly emerges is that, looking at the selectively dropped packets, they have a smaller percentage in S^{2}DCC with respect to SDCC. The reason is many-fold. First of all, in S^{2}DCC, the selective dropping algorithm is executed only by the CHs, which are characterized by a larger buffer size with respect to the sensor nodes. Otherwise, in SDCC the selective dropping is executed by each sensor node in the network. In this way, with S^{2}DCC it is expected to obtain images with a better quality at the sink level (as illustrated in Figures 12 and 13) with respect to SDCC. To summarize, there are two motivations of the improved image quality contributed by S^{2}DCC: (i) the network infrastructure has been potentiated by putting near the sink (where there is more traffic) the more powerful CHs (note that, in a tree-structure network, the chance of traffic congestion increases for decreasing ranks); (ii) S^{2}DCC is able to detect the corrupted packets and consequently drop them, thus reducing unnecessary traffic on the network. These two aspects counterbalance the slightly higher percentage of packets lost with respect to SDCC.

5.1.2. Image quality

The rationale of the selective dropping algorithm is to smartly drop packets in order to fit the bandwidth constraints of WMSNs; in this way, the sink is able to reconstruct more images, with smaller delays, but with smaller sizes, as it can be noticed from Figure 12. When no selective dropping is executed (i.e., No-SDCC scenario), fewer and bigger images are reconstructed at the sink. The quality of a reconstructed image at the sink is also given by its PSNR, reported in Figure 13.

What is remarkable is that images delivered by S^{2}DCC are characterized by higher values of PSNR with respect to SDCC. Such a result can be due to the resiliency of S^{2}DCC against the presence of malicious nodes. In fact, malicious nodes can re-route packets or introduce a considerable number of new packets into the network, thus causing potential situation of congestion and loss of fundamental messages for the reconstruction of images at the sink. Since both SDCC and No-SDCC do not recognize violated messages, the quality of the reconstructed images can be degraded. In S^{2}DCC, instead, if a packet is identified by the cluster head as corrupted, it is discarded and an error notification is sent to the sink. Finally, from Figure 13, it is possible to note that the worst results regarding the PSNR of reconstructed images are associated with No-SDCC, where no selective dropping is executed. In that case, the reconstructed images are often hard to interpret with respect to the original ones.

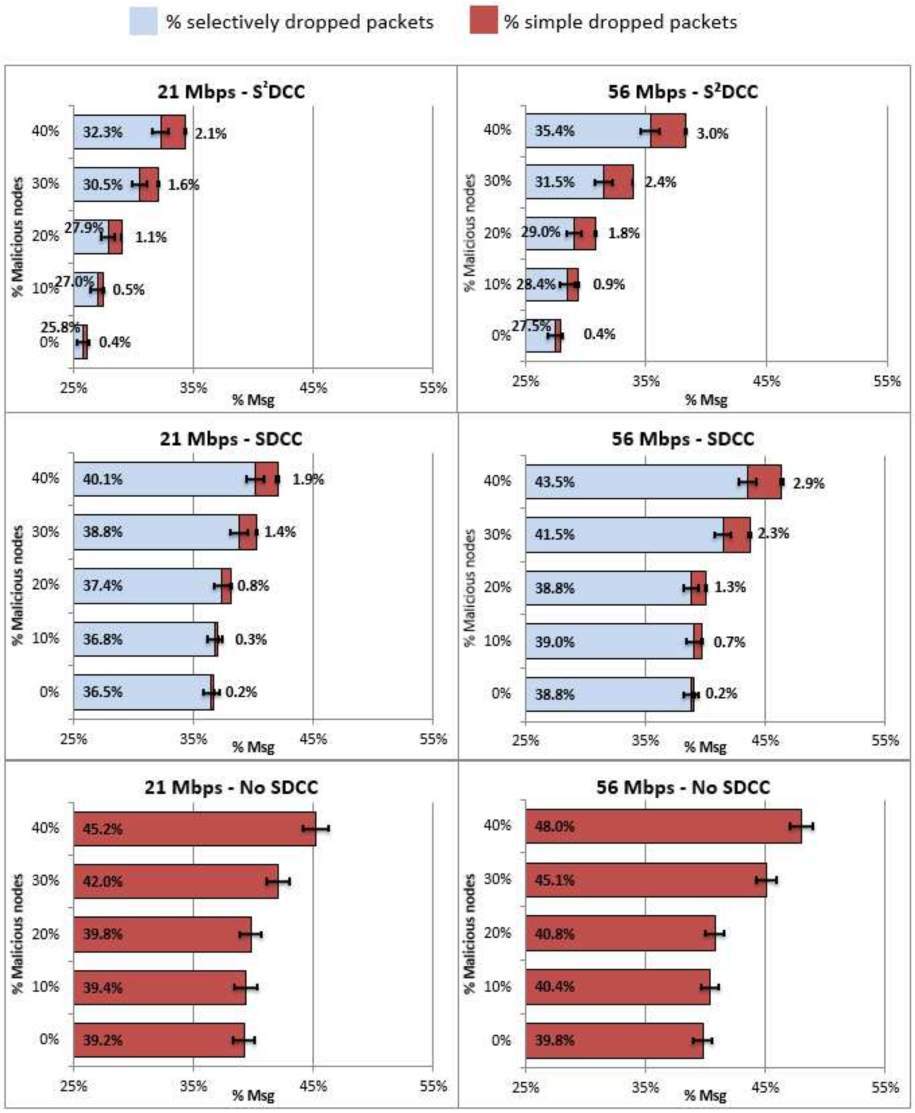

Figure 10: Packet Loss Ratio - Topology 1.

5.1.3. *Error notifications*

With respect to the evaluation of the efficiency of the error notification mechanism introduced in S^{2}DCC, the proportion between the no violated and the violated messages marked at the sink is reported in Figures 14 and 15. As

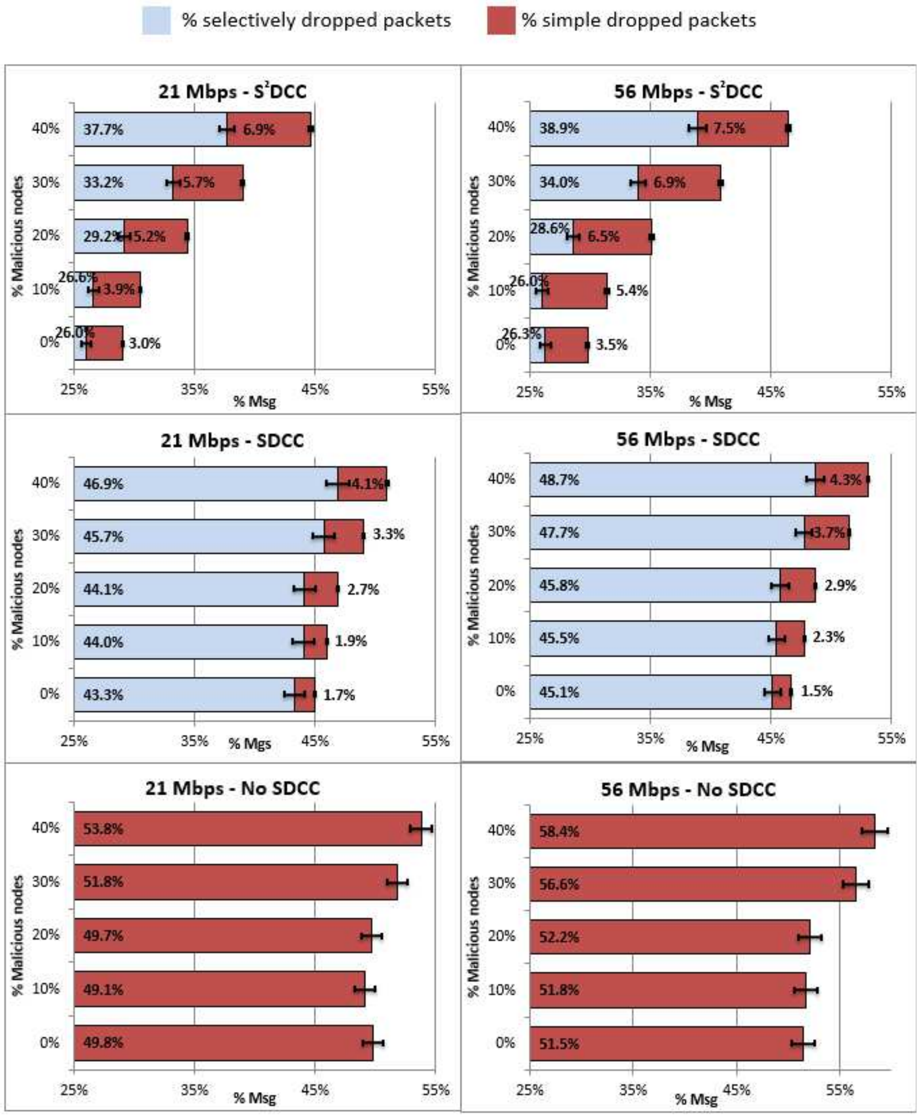

Figure 11: Packet Loss Ratio - Topology 2.

observed for the packet loss ratio, the effectiveness of the integrity checks of the S^{2}DCC is confirmed by the correlation between the increase in the percentage of malicious nodes and the percentage of error notifications.

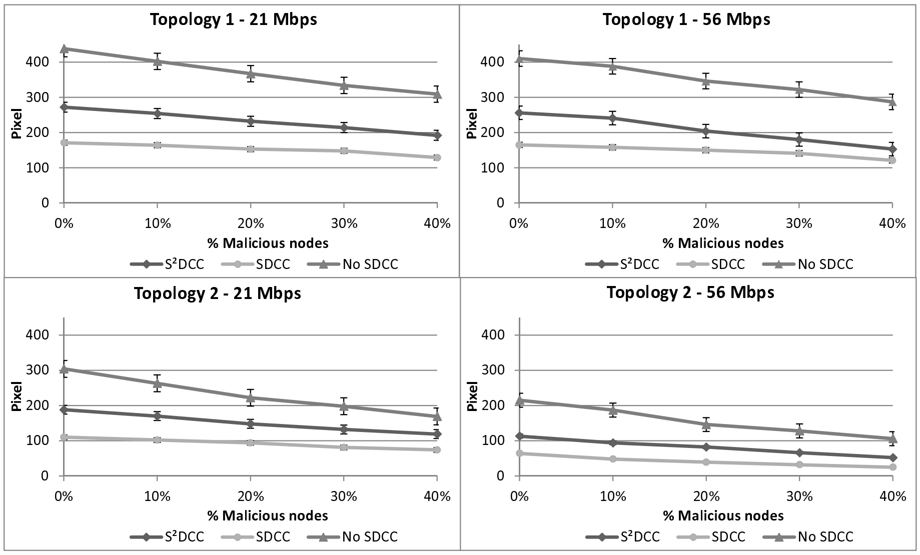

Figure 12: Mean Image Size.

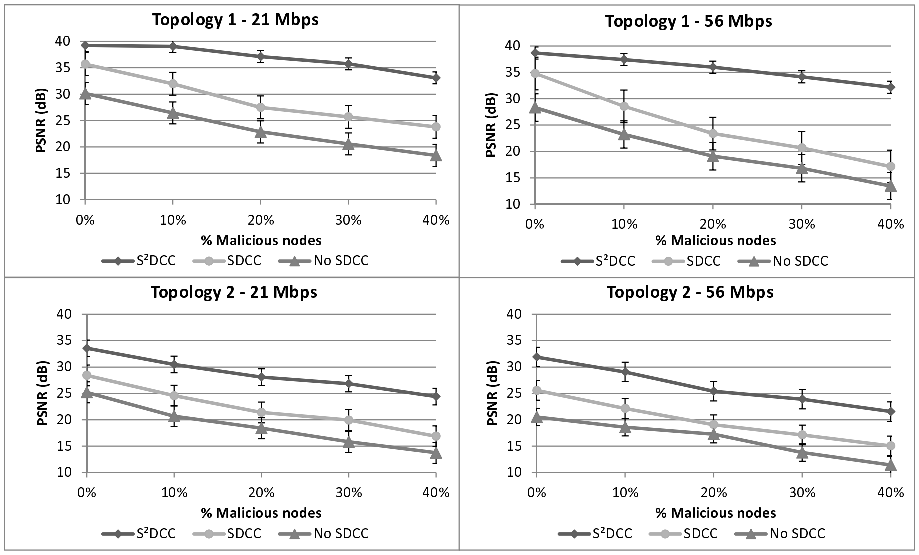

Figure 13: Mean PSNR of Received Images.

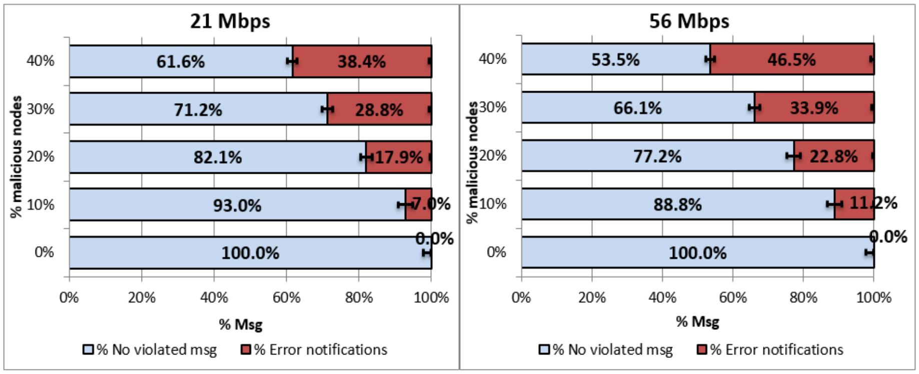

Figure 14: Messages received by the sink in S^{2}DCC - Topology 1.

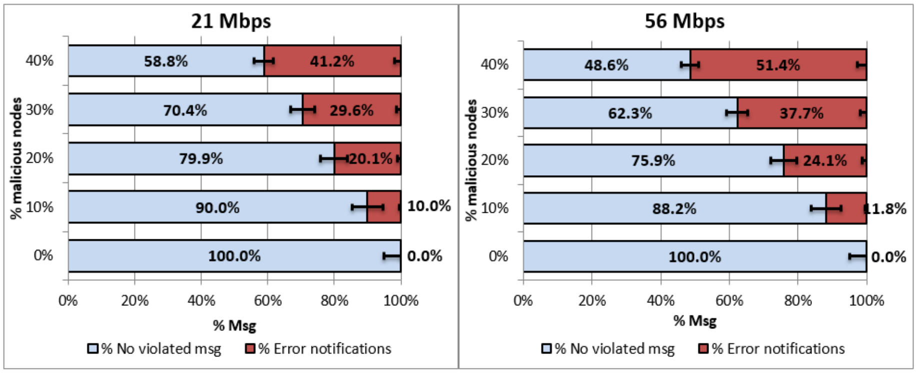

Figure 15: Messages received by the sink in S^{2}DCC - Topology 2.

5.1.4. Signalling overhead and power consumption

The use of signaling messages in S^{2}DCC may lead to additional traffic and higher power consumption. Nevertheless, as it can be seen from Figure 16[2], which reports the mean power consumption of sensor nodes for the three different protocols in Topology 1[3], the new messages introduced in S^{2}DCC do not

[2] The mean power consumption is reported as a mean of the overall consumption of sensor nodes acting in the network.

[3] Similar results have been obtained for Topology 2.

significantly compromise the overall power consumption, which is only slightly higher with respect to SDCC. To fully understand this result is necessary to consider the different contributions to energy consumption arising from S^{2}DCC and SDCC. Comparing the two algorithms, it is possible to found that with S^{2}DCC a higher number of messages is handled by sensor nodes (because the aggregation is executed at CHs only) and that encryption procedures inflate the energy needs of S^{2}DCC. On the other side, with SDCC the aggregation algorithm is executed at each node, which incurs not negligible energy consumption. These different contributions counterbalance each other so that the global energy budget spent by S^{2}DCC and SDCC is almost the same.

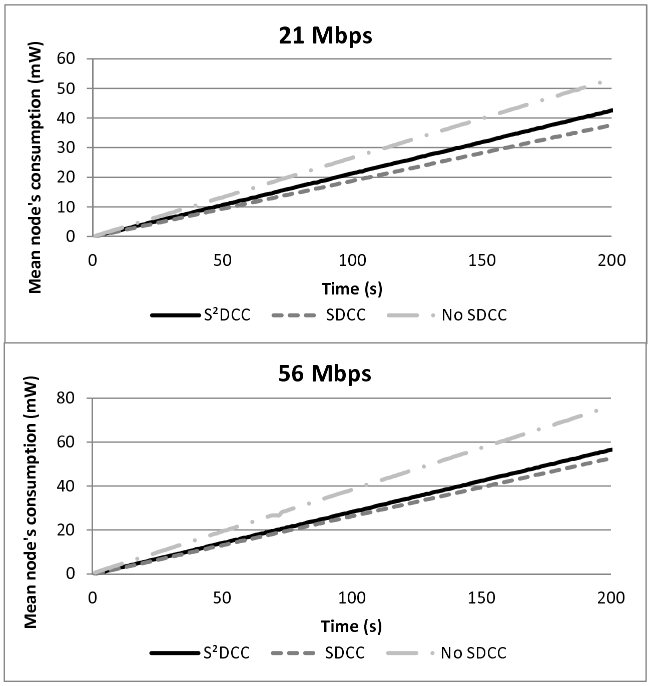

Figure 16: Mean Power Consumption of sensor nodes - Topology 1.

5.1.5. Packet delay

Both SDCC and S^{2}DCC are able to lower packet delays with respect to the No-SDCC scenario, as it can be seen from Figure 17, which reports the

Cumulative Distribution Function (CDF) of the packet arrival time at the sink. In fact, they adopt a congestion control algorithm that enforces bounded queue levels and hence reduces delays. Of course, the degree of improvement is higher with SDCC since its control action is executed at every node instead of at the cluster heads only, as in S^{2}DCC.

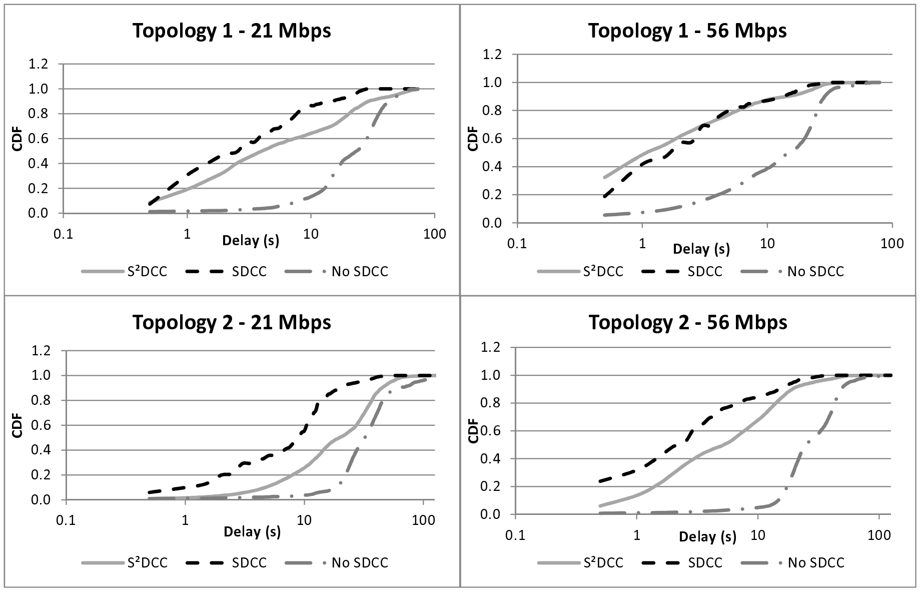

Figure 17: CDF of Packet Arrival Delay.

6. Related Work

The key feature of S^{2}DCC is the capability to encompass in an integrated solution many different technical challenges of WMSNs, including: (i) the security of the data transmitted over the network; (ii) the privacy and the key distribution management; and (iii) the network traffic congestion control. Accordingly, to ease the analysis of related work, this section has been split in three different subsections, each one addressing a different face of S^{2}DCC.

6.1. Security issues in WMSN

The wireless nature of the communication channel in WMSN exacerbates the risk of attacks that can lead to violations of the integrity and the confidentiality of the transmitted images. *Eavesdropping* and *masking* represent two noticeable examples of threats that may occur: the former means that a malicious user could easily discover the communication content by listening to the communication occurring among nodes; the latter happens when a malicious node may mask its real nature behind the identity of a node that is authorized to take part to the communication in order to misroute messages [18].

In the field of WSN, the literature reports many solutions addressing at the same time secure aggregation and various security aspects such as confidentiality, integrity, anonymity, authentication and availability [19]. Unfortunately, in the context of WMSN, security issues are harder than in simple WSN, due to the multimedia nature of the data.

Some existing solutions combine image compression and encryption techniques, in order to face such security threats (an exhaustive and comprehensive view of this topic can be found in [20]). In such a context, thanks to the flexibility of standard codings such as JPEG2000 , in [21], Wang et al. describe an image as composed of different and mutual integrable qualitative levels and encrypt only the data belonging to the basic level, by making useless any attack oriented to the theft of no encrypted transmitted data. Also [22] uses the JPEG compressed domain in order to protect multimedia informaton during transmissions; the proposed content security scheme integrates both encryption and digital fingerprinting.

Other solutions adopt some mathematical and physical properties in order to encrypt the images, such as: Fourier transform [23]; chaotic maps [24] and a more effective cryptosystem for image encryption and decryption based on Brahmagupta-Bhaskara equation and chaos [25]; Mellin transform [26]; DTC transform [27]; SPIHT codec [28]; Haar Wavelet transform [29].

Note that the encryption of images is a highly power consuming task ; as a consequence, the most innovative solutions adopt selective encryption schemes for the multimedia contents In particular, in order to limit the energy consumption, MAES [27], AES [30] and ECC [31] are the most effective encryption techniques emerged in the last years in the image security area.

Moreover, all the proposed approaches can be also classified into two big families depending on whether the hop-by-hop or end-to-end cryptography is used. Hop-by-hop encryption requires each node to decrypt every message it receives to allow in-network processing. To do this, each node has to share the keys with all its neighbors, thus causing a confidentiality breach. Furthermore, applying several consecutive encryption/decryption operations can negatively impair latencies and costs. Whereas end-to-end approaches need not to distribute the decryption keys to all the network nodes, but use other mechanisms (e.g., hashing techniques) to perform checks on data violations, as in [10] [32] for WSN and in our S^{2}DCC developed for WMSN.

6.2. Privacy preservation and key management

The literature reports three different types of solutions that aim at hindering image privacy attacks: anonymity mechanisms based on data cloaking [33], privacy-aware mechanisms based on securing the communication channels [34] and privacy policy based approaches (e.g., access control mechanisms) [35]. In particular, as regards the anonymity mechanisms, no sensor nodes have a complete vision of the data, but only partial information, since the original data is divided into some shares. The weakness of such an approach is the lack of a complete solution able to guarantee at the same time the anonymity and the security of the transmitted data. Moreover, the presented solutions lack to adopt a unique privacy model. In two previous works of the authors, two protocols, named DyDAP [10] and SETA [32], apply a role-based privacy model to a WSN and a cluster-based WSN, respectively, along with an efficient algorithm for congestion control. These works handle scalar data, whereas in WMSN context exists no more solutions yet which cope with all these aspects at the same time, as S^{2}DCC does.

Another important issue regards the key management in the wireless sensor networks, since the key distributon system adopted may affect both the energy consumed by the sensor devices and the security of the overall network. Many solutions have been proposed, taking into account key revocation as well as computation and communication constraints [12]. S^{2}DCC, besides adopting none of them yet, is suitable to different key management systems.

6.3. Traffic congestion issues

Due to the bursty nature of the traffic in a WMSN, congestion is more likely to happen than in a traditional WSN [1]. As a consequence, a congestion control mechanism represents a fundamental component of a WMSN. In general, a congestion control protocol should be able to i) detect a congestion event, ii) notify the involved nodes, and iii) take some countermeasures to mitigate the negative effects. Among the several protocols proposed in the literature of the WMSN [8], some of them, [4] [5], share the concept of dropping selected packets from the transmission buffer queue in order to mitigate the negative effects of a congestion, while, at the same time, preserving the video quality perceived at the receiver as much as possible. [36] employs a Source Congestion Avoidance Protocol (SCAP) in the source nodes, and a Receiver Congestion Control Protocol (RCCP) in the intermediate nodes in order to adjust the sending rate of source nodes and the distribution of the departing packets from the source nodes, by monitoring the queue length of the intermediate nodes. Note that none of this work deals with the security of transmissions.

In [4], different metrics are taken into account in the congestion detection process, such as the buffer occupancy of the parent sensor node, the ratio of incoming to outgoing packets, and the number of contenders. Once a congestion is detected, the multi-layered structure of the progressive JPEG is used to select specific packets to be dropped; in particular, a weight is assigned to each packet, calculated considering the current hop count of the packet, its average delay and its frame index. When the buffer is full, the packet with the minimum score is dropped. In [5], instead, both path scheduling and packet scheduling are combined; that is, the multipath selection algorithm is responsible of finding the sets of paths that support the highest overall end-to-end transmission bandwidth, while the packet scheduling employs a recursive distortion prediction model to select and drop the packets that are predicted to have a smaller impact on the video distortion at the receiver.

Other contributions available in literature are explicitly based on the use of resource intensive codecs, like JPEG2000 [37] and H.264 [38] , or lightweight codecs like SPIHT [6] but with no support to spatial scalability. In [7], the authors propose SDCC, where the local buffer occupancy is used as a reference metric to detect congestion events, while the concepts of linear discrete time control theory are used to grant a limited computational complexity.

Summarizing, in this paper a step further beyond SDCC has been move by proposing its secure version, i.e., S^{2}DCC, which is able to guarantee security and privacy requirements in a WMSN. Differently from SDCC, it adopts the hierarchical topology of the network, thus moving the selective dropping task from each node to the cluster head ones. This allowed to insert the security mechanisms inside the network without compromising the overall power consumption, and, at the same time, still preserving the quality of the transmitted images.

7. Conclusions and Future Research

In this paper the congestion control, the security and the privacy requirements in wireless sensor networks dealing with multimedia data (i.e., images) have been addressed.

As regards congestion control, the FS-SPIHT codec has been adopted to encode the sensed images in a flexible manner, due to its spatial and quality scalability, and to ease the execution of the selective dropping algorithm; in fact, in case a congestion is detected, packets can be dropped from the transmission queue of the sensor nodes following a specific priority order which limits, as much as possible, the degradation of the images reconstructed at the sink. The remarkable novelty of S^{2}DCC regards security and privacy introduced to let the network being resilient against the presence of malicious nodes, and to guarantee the secrecy of the sensed data, or any other sensitive information transmitted towards the sink. Due to the limited power resources, the satisfaction of privacy and secutity is achieved thanks to the adoption of a hierarchical architecure. More in details, sensor nodes have been grouped in clusters, thus only the cluster heads communicate with the sink. The idea is to reduce the power consumption by means of a strict distribution of tasks among the several nodes according to their function and role into the network.

The performance of S^{2}DCC have been evaluated in a video surveillance scenario thanks to an extended version of the Castalia simulator, using different topologies and network loads. The behavior of S^{2}DCC has been compared with SDCC and No-SDCC algorithms using different metrics: the packet loss ratio, the packet arrival delay, PSNR and size of the images received at the sink, and the mean power consumption of sensor nodes. The obtained results have shown that, despite the introduction of new requirements regarding privacy and security of sensed data, the overall performance of the WMSN is not compromised with respect to the previous SDCC.

In the next future we are planning to define a score reputation mechanism, which supports the sink in the malicious nodes identification with a well defined probability, in case an error notification message is received. Hence, S^{2}DCC is under investigation in order to define the method for an integration in a more complex system, as the one defined in Internet of Things applications [39].

References

[1] I. Akyildiz, T. Melodia, K. Chowdhury, A survey on wireless multimedia sensor networks, Comput. Netw. 51 (4) (2007) 921–960.

[2] C. Wan, S. Eisenman, A. Campbell, CODA: Congestion Detection and Avoidance in Sensor Networks, in: Proc. of ACM SenSys, Los Angeles, California, USA, 2003.

[3] D. Patil, S. Dhage, Priority-based Congestion Control Protocol (PCCP) for controlling upstream congestion in Wireless Sensor Network, in: Proc. of IEEE ICCICT, Mumbai, India, 2012.

[4] C. Sonmez, S. Isik, M. Donmez, O. Incel, C. Ersoy, SUIT: A Cross Layer Image Transport Protocol with Fuzzy Logic Based Congestion Control for Wireless Multimedia Sensor Networks, in: Proc. of NTMS, Istanbul, Turkey, 2012.

[5] I. Politis, M. Tsagkaropoulos, T. Dagiuklas, S. Kotsopoulos, Power efficient video multipath transmission over wireless multimedia sensor networks, Mob. Netw. Appl. 13 (3-4) (2008) 274–284.

[6] H. Danyali, A. Mertins, Fully spatial and snr scalable, SPIHT-based image coding for transmission over heterogenous networks, Journal of Telecommunications and Information Technol. 2 (2003) 92–98.

[7] A. Martelli, L. A. Grieco, M. Bacco, G. Boggia, P. Camarda, Selective dropping congestion control for wireless multimedia sensor networks, in: Proc. of IEEE Symp. on Computers and Commun., ISCC, Kerkira, Corfu, Greece, 2011.

[8] S. Misra, M. Reisslein, X. Guoliang, A survey of multimedia streaming in wireless sensor networks, Communications Surveys Tutorials, IEEE 10 (4) (2008) 18–39.

[9] Castalia simulator - official website, accessed: 2013-04-14 (2013).
URL `http://castalia.research.nicta.com.au/index.php/en/`

[10] S. Sicari, L. A. Grieco, G. Boggia, A. Coen-Porsini, Dydap: A dynamic data aggregation scheme for privacy aware wireless sensor networks, Elsevier Journal of Systems and Software 88 (1) (2012) 152–166.

[11] A. Said, W. A. Pearlman, A new, fast, and efficient image codec based on set partitioning in hierarchical trees, IEEE Trans. on Circuits and Systems for Video Technology 6 (3) (1996) 243–250.

[12] A. Selcuk Uluagac, R. Beyah, J. Copeland, Secure source-based loose synchronization (sobas) for wireless sensor networks, Parallel and Distributed Systems, IEEE Transactions on 24 (4) (2013) 803–813.

[13] S. Sicari, A. Rizzardi, L. A. Grieco, A. Coen-Porisini, Gone: dealing with node behavior, in: 5th IEEE International Conference on Consumer Electronics, IEEE 2015 ICCE-Berlin, 2015.

[14] I. Daubechies, W. Sweldens, Factoring wavelet transforms into lifting steps, Journal of Fourier Analysis and Applications 4 (3) (1998) 247–269.

[15] S. Nath, P. Gibbons, S. Seshan, Z. Anderson, Synopsis diffusion for robust aggregation in sensor networks, ACM Trans. Sen. Netw. 4 (2) (2008) 7:1–7:40.

[16] M. Passing, F. Dressler, Experimental performance evaluation of cryptographic algorithms on sensor nodes, in: Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on, IEEE, 2006, pp. 882–887.

[17] C. Castelluccia, E. Mykletun, G. Tsudik, Efficient aggregation of encrypted data in wireless sensor networks, in: Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005. The Second Annual International Conference on, 2005, pp. 109–117.

[18] H. Chan, A. Perrig, Security and privacy in sensor networks, IEEE Computer Magazine 36 (10) (2003) 103–105.

[19] S. Ozdemir, Y. Xiao, Secure data aggregation in wireless sensor networks: a comprehensive overview, Computer Networks 53 (12) (2009) 2022–2037.

[20] L. A. Grieco, G. Boggia, S. Sicari, P. Colombo, Secure wireless multimedia sensor networks: a survey, in: Proc. of UBICOMM, Sliema, Malta, 2009.

[21] W. Wang, D. Peng, H. Wang, H. Sharif, A cross layer resource allocation scheme for secure image delivery in wireless sensor networks, in: Proc. of ACM IWCMC, Honolulu, Hawaii, USA, 2007.

[22] Y. Xu, L. Xiong, Z. Xu, S. Pan, A content security protection scheme in {JPEG} compressed domain, Journal of Visual Communication and Image Representation 25 (5) (2014) 805 – 813.

[23] J. Lang, Image encryption based on the reality preserving multiple parameter fractional fourier transform and chaos permutation, Optics and Lasers in Engineering 50 (7) (2012) 929–937.

[24] E. A. Naeem, M. M. A. Elnaby, N. F. Soliman, A. M. Abbas, O. S. Faragallah, N. Semary, M. M. Hadhoud, S. A. Alshebeili, F. E. A. El-Samie, Efficient implementation of chaotic image encryption in transform domains, Journal of Systems and Software 97 (2014) 118 – 127.

[25] K. Rao, K. Kumar, P. M. Krishna, A new and secure cryptosystem for image encryption and decryption, IETE Journal of Research 57 (2) (2011) 165–171.

[26] N. Zhoua, Y. Wang, L. Gong, X. Chen, Y. Yang, Novel color image encryption algorithm based on the reality preserving fractional mellin transform, Optics & Laser Technology 44 (7) (2012) 2270–2281.

[27] P. Telagarapu, B. Biswal, V. Guntuku, Security of image in multimedia applications, in: Proc. of International Conference on Energy, Automation, and Signal (ICEAS), Bhubaneswar, Odisha, 2011.

[28] X. Zhang, X. Wang, Chaos-based partial encryption of SPIHT coded color images, Signal Processing 93 (9) (2013) 2422–2431.

[29] S. Tedmori, N. Al-Najdawi, Image cryptographic algorithm based on the haar wavelet transform, Information Sciences 269 (2014) 21 – 34.

[30] F. Riaz, S. Hameed, I. Shafi, R. Kausar, A. Ahmed, Enhanced image encryption techniques using modified advanced encryption standard, in: Proc of 2nd International Multi Topic Conference, IMTIC, Jamshoro, Pakistan, 2012.

[31] L. D. Singh, K. M. Singh, Image encryption using elliptic curve cryptography, Procedia Computer Science 54 (2015) 472 – 481.

[32] S. Sicari, L. Grieco, A. Rizzardi, G. Boggia, A. Coen-Porisini, Seta: A secure sharing of tasks in clustered wireless sensor networks, in: Proc. of IEEE WiMob, Lyon, France, 2013.

[33] S. Saghaiannejadesfahani, Y. Luo, S.-C. Cheung, Privacy protected image denoising with secret shares, in: Proc. of IEEE ICIP, Lake Buena Vista, FL, United States, 2012.

[34] D. A. Fidaleo, H. Nguyen, M. Trivedi, The networked sensor tapestry (NeST): A privacy enhanced software architecture for interactive analysis of data in video-sensor networks, in: Proc. of 2nd ACM International Workshop on Video Surveillaince & Sensor Network, New York, USA, 2004.

[35] K. Yi, M. Han, J. Park, Privacy protection and access control of image information processing devices, Journal of Internet Technology 12 (5) (2011) 711–716.

[36] S. M. Aghdam, M. Khansari, H. R. Rabiee, M. Salehi, Wccp: A congestion control protocol for wireless multimedia communication in sensor networks, Ad Hoc Networks 13, Part B (2014) 516 – 534.

[37] W. Y. Sahinoglu, Z. Vetro, A. Sahinoglu, Energy efficient jpeg 2000 image transmission over wireless sensor networks, in: Proc. of IEEE GLOBECOM, Dallas, Texas, USA, 2004.

[38] D. Kandris, M. Tsagkaropoulos, I. Politis, A. Tzes, S. Kotsopoulos, A hybrid scheme for video transmission over wireless multimedia sensor networks, in: Proc. of 17th Mediterranean Conf. on Control and Automation, Thessaloniki, Grece, 2009.

[39] S. Sicari, A. Rizzardi, L. Grieco, A. Coen-Porisini, Security, privacy and trust in internet of things: The road ahead, Computer Networks 76 (0) (2015) 146 – 164.