Evaluating and Improving the Scalability of RPL Security in the Internet of Things

Antonio Arena, Pericle Perazzo, Carlo Vallati, Gianluca Dini, Giuseppe Anastasi

Dept. of Information Engineering, University of Pisa, Via Girolamo Caruso 16, Pisa, Italy

Abstract

Wireless Sensor and Actuator Networks (WSANs) will represent a key building block for the future Internet of Things, as a cheap and easily-deployable technology to connect smart devices on a large scale. In WSAN the Routing Protocol for Low-Power and Lossy Networks (RPL) has a crucial role as the standard IPv6-based routing protocol. RPL specifications define a basic set of security features, without which it would be open to disruptive routing attacks. However, the impact of these features on the WSAN performance has not been thoroughly investigated yet. The contribution of this paper is two-fold. First, we extensively evaluate the impact of security mechanisms on the scalability of WSANs by means of both simulations and real experiments. We show that the protection against eavesdropping and forgery has a modest impact on the performance, whereas the protection against replay has a more considerable impact, especially on the network formation time which increases noticeably. Despite this, we show that protecting against replay reduces the number of control messages exchanged and improves routes optimality. For these reasons, we recommend to always use the security mechanisms. Finally, we propose a standard-compliant optimization for defending against replay that reduces the impact on the overall performance.

Keywords: Internet of Things, Embedded Systems, RPL, Secure Routing,

Email addresses: antonio.arena@ing.unipi.it (Antonio Arena), pericle.perazzo@iet.unipi.it (Pericle Perazzo), carlo.vallati@iet.unipi.it (Carlo Vallati), gianluca.dini@iet.unipi.it (Gianluca Dini), giuseppe.anastasi@iet.unipi.it (Giuseppe Anastasi)

1. Introduction

Recent technology advancements are rapidly making the Internet of Things (IoT) a reality [1]. According to the IoT paradigm, objects will be empowered with communication capabilities to enable seamless integration with information systems. In this future, such smart objects will penetrate the physical world around us, in some cases by implementing remote monitoring and control capabilities, in others by offering enhanced features that exploit automation and self-coordination. IoT applications are expected to cover a wide range of domains such as smart home, smart city, e-health, industrial automation, logistics, and so on.

Wireless Sensor and Actuator Networks (WSANs) will be a key enabler for all these IoT applications, because they allow for rapid and cost-effective installation of smart objects over large areas. The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [2], standardized in 2012 by the IETF ROLL working group, is considered at the present the most mature option to connect IPv6-enabled devices and form WSANs over lossy links with minimal overhead [3]. Considering the importance of the delivered services, protecting the routing functionalities from attacks will be crucial to prevent malicious attempts to disrupt IoT network operations [4]. A basic set of cryptographic security mechanisms to guarantee routing resilience to external attackers has been introduced by design in RPL specifications [2]. However, the impact, in terms of such mechanisms on the performance, has not been investigated yet.

In this paper, we analyze the impact of the RPL security mechanisms on the performance of large-scale network topologies. First, we implemented the RPL security features on Contiki OS, a popular operating system for sensor nodes. To this aim, we analyzed and tackled all the aspects left unspecified by the standard in order to develop, to the best of our knowledge, the first implementation publicly available. Then, we carried out an extensive performance evaluation of the RPL security features by means of both simulations and real experiments. Our analysis shows that the RPL security mechanisms have a negligible impact on the performance at the network bootstrap, considering a configuration that does not protect against replay-based attacks. When instead a complete replay protection is adopted, the impact on the performance is more relevant, i.e., the network bootstrap time increases noticeably. In order to reduce the impact of the replay protection mechanism, a standard-compliant optimization that reduces the length of the bootstrap phase and still defends against replay-based attacks is proposed and evaluated. The results allowed us to conclude that the adoption of the RPL security features, including also the replay protection mechanism, have a limited cost and their adoption is desirable.

The present paper significantly extends the previously published conference version[5] by introducing an evaluation of the RPL security mechanisms to cover also downward routes formation. We evaluated the impact of the security features with both storing and non-storing modes enabled on upward and downward routes formation. The performance evaluation has been significantly extended considering larger simulation topologies and also real-world experiments. Finally, we propose a standard-compliant optimization of the RPL security features in order to moderately lower the impact of security on all routing procedures.

The remainder of the paper is organized as follows. In Section 7 we review related work. In Section 2 we offer a short introduction to the RPL specifications, including its security mechanisms. In Section 3 we describe our standard-compliant implementation of the security mechanisms of RPL, focusing in particular on the aspects left unspecified by the standard. In Section 4 we evaluate the impact of RPL security on performance by means of simulations. In Section 5 we describe our standard-compliant optimization and we evaluate its performance by means of simulations. In Section 6 we validate the results by means of experiments on a real testbed. Finally, Section 8 concludes the paper.

2. RPL Protocol

The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [3, 2] is an IPv6 distance-vector routing protocol designed for resource-constrained devices and lossy wireless environments. RPL assumes that the majority of the traffic is upward, i.e., directed towards a single node acting as a border router. Downward traffic, i.e., generated by the border router and directed towards other nodes, is considered to be sporadic, and node-to-node traffic to be rare. For this reason, RPL builds and maintains a logical topology for upstream data delivery, whereas downstream routes are established only when required. Specifically, the topology is a *Destination Oriented Directed Acyclic Graph* (DODAG), in which every node has a set of neighbors

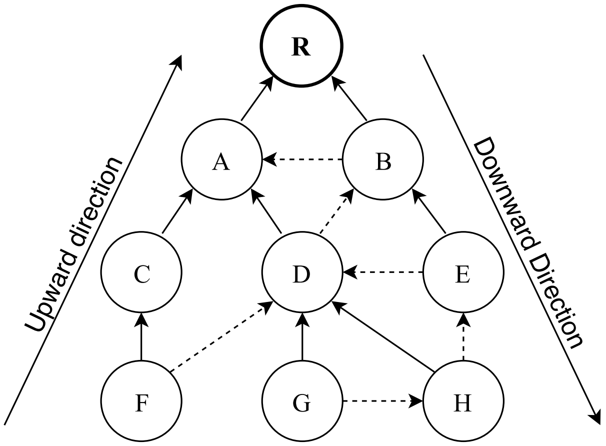
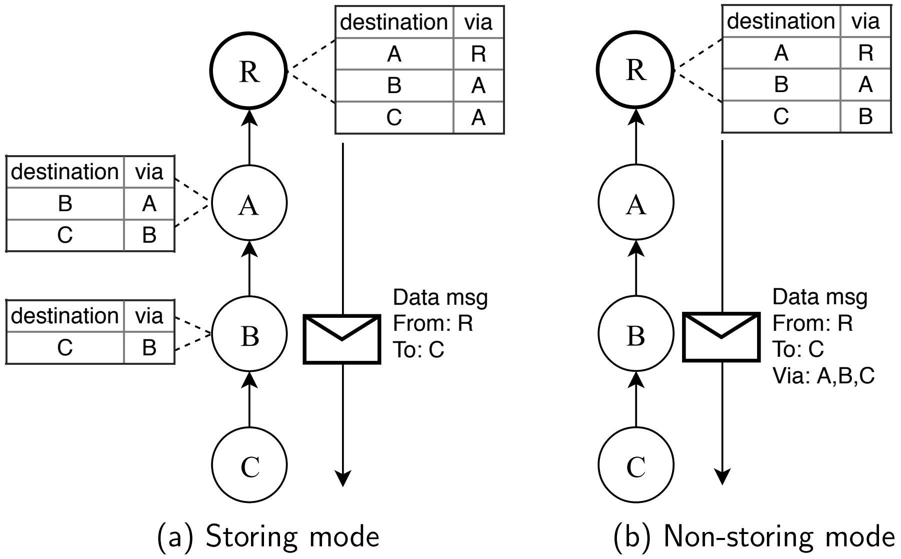


Figure 1: RPL DODAG topology. Node R is the DODAG root. The solid arrows represent the preferred parent relationships. The dotted arrows represent alternative parents relationships.

(*parent set*), which are candidates for upstream data delivery. Among the nodes in the parent set, one node is selected as the *preferred parent*. The preferred parent is the node exploited for upstream data forwarding, whereas the other parents are kept as failover. We call *children* of a given node all those nodes that selected the given node as preferred parent. The DODAG is rooted in a single node, called *DODAG root*, to which all upstream data is directed. An example of RPL DODAG topology is shown in Figure 1. The DODAG root is usually implemented by the border router. It is responsible for triggering the network formation through the emission of *DODAG Information Object* (DIO) messages. Initially, every non-root node listens for DIO messages. When a DIO is received, the node joins the network using the information included in the message. Right after joining the network, the node starts emitting DIOs to advertise its presence and its distance to the root. During regular operations, the emission of DIO messages is regulated by the *Trickle* algorithm [6], which aims at reducing the power consumption of the nodes by minimizing redundant messages and by adapting dynamically transmission rates over time. The asynchronous emission of DIOs can be requested through *DODAG Information Solicitation* (DIS) messages, e.g., to accelerate the join process of a node during network formation or to recover from errors during regular network operations.

In order to enable downward traffic, every non-root node emits *Destina-*



destination	via
A	R
B	A
C	A

destination	via
B	A
C	B

destination	via
C	B

(a) Storing mode

destination	via
A	R
B	A
C	B

(b) Non-storing mode

Figure 2: Downward routing tables in storing and non-storing modes. An example of data message routing is shown with both modes.

tion Advertisement Object (DAO) messages to propagate destination information upward along the DODAG. RPL specifies two modes of operation to establish and maintain downward routes: *storing mode* and *non-storing mode*. In the storing mode, every node stores the routes advertised by all nodes in its subtree. With the storing mode, a DAO message is sent in unicast by a child node to its preferred parent, which stores the advertised routes received before forwarding the received DAO messages to its preferred parent. As a consequence, every node is able to route messages to its subtree with the storing mode. In Figure 2a the downward routing tables using the storing mode are shown. Every node maintains in its routing table reachability information for its subtree. Note that the root node does not need to insert the list of hops for routing a message to Node C.

In non-storing mode, intermediate parents do not store downward routes. Only the DODAG root stores the routes advertised by all the nodes in the network. Every node sends in unicast the DAO message to the DODAG root. In particular, every node advertises its own global IPv6 address and its preferred parent IPv6 address. In non-storing mode, only the DODAG root is able to route messages in the downward direction, so every packet directed to a non-root node has to transit through the root node in order to be correctly delivered [7]. In Figure 2b the downward routing table using the non-storing mode is shown. Non-root nodes do not have a routing table for downward routes. On the other hand, the DODAG root maintains in its routing table the reachability information for all nodes of the network. In order to route a message to a node, the root node has to insert the list of hops needed to reach the destination node, such as Node C.

In both storing and non-storing mode, the standard provides for an acknowledgment mechanism for DAO messages. This mechanism ensures that DAO messages are correctly delivered to its preferred parent. Whenever a DAO message is sent, the recipient sends back in unicast a *Destination Advertisement Object Acknowledgment* (DAO-ACK) message. If a node does not receive a DAO-ACK message, it retransmits the same DAO message to its preferred parent. After a maximum number of DAO retransmissions, the DAO sender selects another preferred parent.

2.1. Security Mechanisms

This section gives an overview of the RPL security mechanisms defined by the standard. RPL specifies three modes of operation to enforce security in the routing mechanism: *unsecured mode*, *preinstalled mode*, and *authenticated mode*. In the unsecured mode, RPL messages are sent in the clear and without any security protection. In the preinstalled mode, RPL messages are protected by cryptography using keys assumed to be already present in each node at boot time. Finally, in the authenticated mode, RPL messages are protected in the same way, but the nodes receive keys from some key authority after undergoing an authentication process. The preinstalled and the authenticated modes differ only in the way the keys are distributed to nodes, while they have in common all the other security mechanisms. Throughout, we will refer to both preinstalled and authenticated modes as “secured modes”.

RPL specifications define the following security services: (a) *data confidentiality*, (b) *data authenticity*, (c) *replay protection*. Data confidentiality assures that routing control messages are read from legitimate destinations only. Data authenticity assures that routing control messages come from legitimate sources. Replay protection assures that malicious duplicates of routing control messages are discarded.

If the DODAG operates in a secured mode, all RPL messages are *secured*. A secured RPL message follows the general format shown in Figure 3, in which the message body is preceded by a Security Section. The Code field

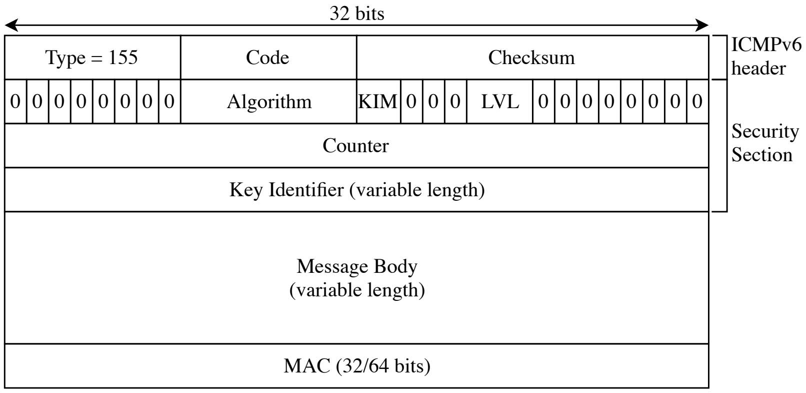


Figure 3: Secured RPL message format.

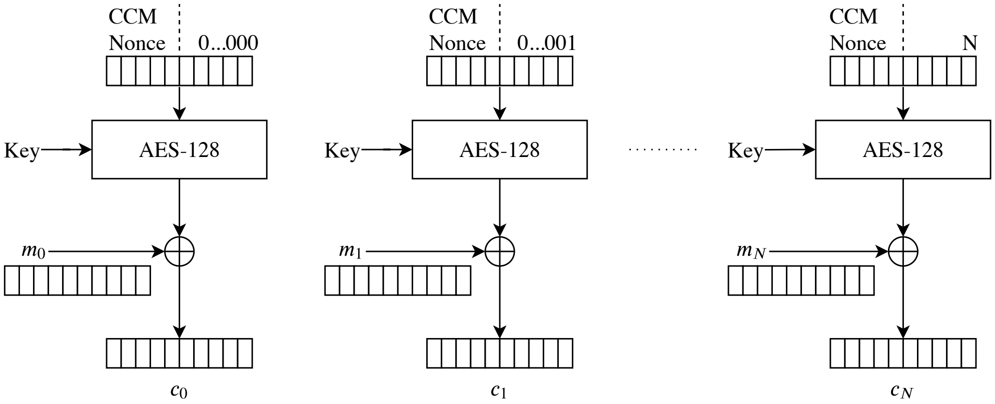


Figure 4: CCM with CTR encryption mode with AES-128.

of the ICMPv6 header determines the type of the RPL message: secured DIO, secured DIS, etc. The Algorithm field of the Security Section specifies the algorithm suite employed to authenticate and encrypt the message. With the current version of the specifications, only CCM (CTR with CBC-MAC) with AES-128 is supported [8]. CCM is a mode of operation for 128-bit block ciphers which can provide for both confidentiality and authenticity by combining the CTR (CounTeR) encryption mode of operation, shown in Figure 4, and CBC-MAC (Cipher Block Chaining Message Authentication Code), shown in Figure 5. The MAC length can be 32 or 64 bit. The LVL bits (Security Level) specify whether the message is only authenticated or both authenticated and encrypted, and the length of the MAC field. The

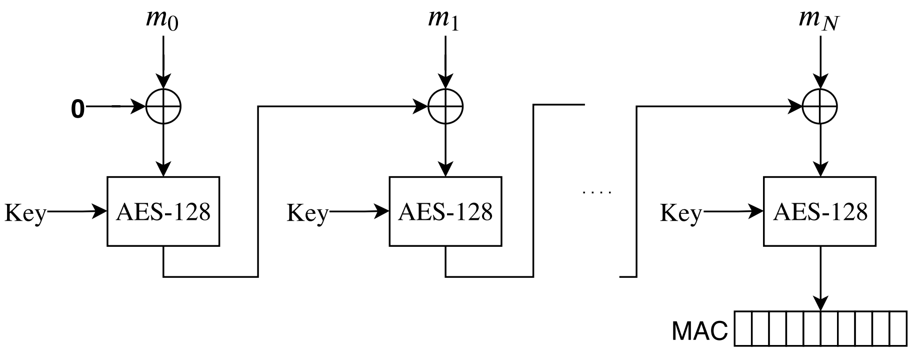


Figure 5: CCM CBC-MAC mode with AES-128.

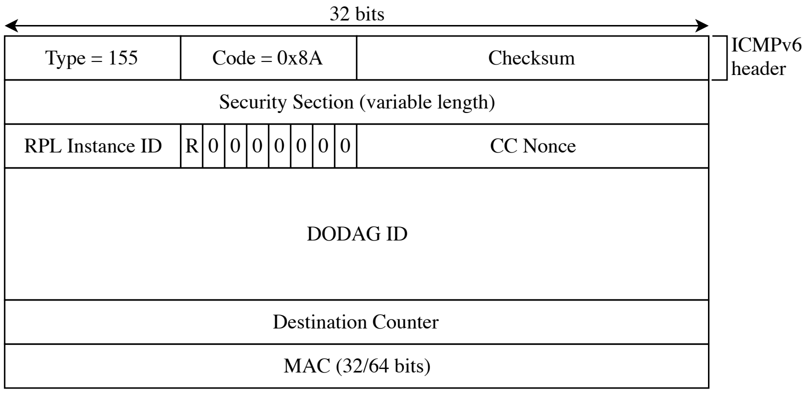


Figure 6: Consistency Check (CC) message format.

Key Identifier field, whose format is specified by the KIM bits (Key Identifier Mode), identifies the employed cryptographic key. Every secured RPL message carries a 32-bit Counter field, whose value is incremented at each sent RPL message.

In order to implement the replay protection mechanism, RPL specifications introduce an additional RPL message, the *Consistency Check* (CC) message. The format of such a message is shown in Figure 6. The CC message is used to issue authenticated challenge-response handshakes, and to inform a destination about its last valid Counter value using the Destination Counter field. The CC message carries the 16-bit CC Nonce field, used as a proof of *freshness* within challenge-response handshakes. The CC message includes the R bit, which specifies whether the message is a challenge (*CC request*, $R = 0$) or a response (*CC response*, $R = 1$).

3. Secure RPL

Many aspects of the RPL security features are left unspecified by the standard. Since these aspects are extremely important for a real implementation of the protocol, in this section we introduce our design choices for our implementation of the RPL security mechanisms in the Contiki operating system. Our implementation extends the standard module ContikiRPL [9], which provides only for the unsecured mode. We made the implementation available from a public repository[1].

3.1. Security Configurations

In our implementation we propose two possible configurations:

- a *light-security configuration* which implements data confidentiality, data authenticity and an incomplete but lightweight replay protection;

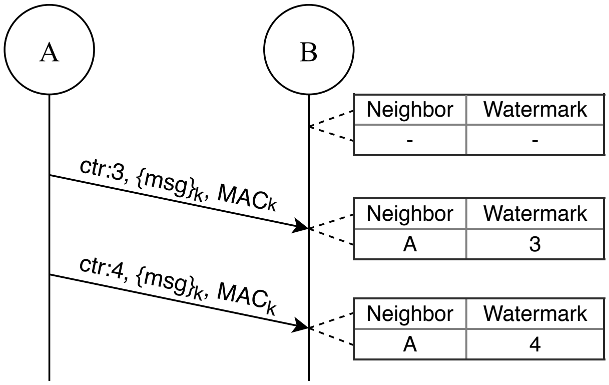
- a *full-security configuration* which implements a complete replay protection, in addition to data confidentiality and data authenticity.

These two configurations are not provided by the standard. We propose them in this paper, basing on two different ways in which the replay protection mechanism can be implemented. Both of them comply with the standard. We assume that a network-wide cryptographic key is already present in each node at boot time with both configurations. This realizes the pre-installed security mode foreseen by the RPL specifications [2], which is, of course, vulnerable to key stealing through node compromise. However, the presented implementation is also compliant with the authenticated security mode, in which the nodes receive keys from some key authority.

3.2. Threat Model

The light-security configuration defends against an adversary which tries to eavesdrop legitimate RPL messages to infer the topology, or forge malicious RPL messages to modify the DODAG topology, or become part of the network. This models a wide range of simple, yet disruptive, routing attacks [10, 11, 12, 13, 14, 15]. Also, the light-security configuration defends against *local replay* attacks. With such an attack, an adversary replays legitimate

[1] Secure RPL implementation available at https://unipisec.github.io



Neighbor	Watermark
-	-

Neighbor	Watermark
A	3

Neighbor	Watermark
A	4

Figure 7: Example of watermark initialization in light-security configuration.

RPL messages to nodes which are within the transmission range of the originator node. For example, using local replay, it is possible for an adversary to mount a DIO Suppression attack [16], that can cause some nodes to remain hidden and some routes to remain undiscovered.

The full-security configuration defends also against *remote replay* attacks. With remote replay attacks, an adversary replays legitimate RPL messages to nodes out of the transmission range of the originator node. For example, she replays a legitimate DIO message originally sent by the root in a zone of the network where the root is not directly reachable. The victim nodes receiving such replayed DIO are led to believe that they have a direct link with the root and could forward their upstream data along such link. Since the link does not actually exist, all the upstream communication of the victim nodes will be broken.

3.3. Light-Security Configuration

The light-security configuration simply includes a security section on each RPL message, which provides for integrity with a Message Authentication Code (MAC), confidentiality with encryption, and a lightweight replay protection with a Counter field. Every time a secured RPL message is received, its Security Section is processed. Such a processing consists on checking the MAC validity, and then decrypting the message itself. With light-security configuration unsecured messages are silently discarded. Every node maintains a *watermark* for each neighbor node, containing the highest Counter field received from that neighbor. At bootstrap time, every node does not have any watermark. Upon receiving a secured RPL message, if the wa-

Table 1: Notations

Symbol	Meaning
$\{...\}_k$	Data encrypted with the key k
$\mathrm{MAC_k}$	MAC computed over the entire packet with the key k
msg	A RPL message, which can be DIS, DIO, DAO, DAO-ACK and CC messages
crt:x	Counter field in the Security Section set to x
$\mathrm{n_i}$	CCNonce field randomly chosen by Node i
$\mathrm{n_i}$:x	CCNonce field set to value x by Node i

termark for the sender node does not exist, it means that a new neighbor node has been discovered. A node receiving the first secured RPL message from this new neighbor node, e.g., a Secured DIO message, creates a new watermark for this node and initializes it with the received Counter field. Upon receiving a new message from that neighbor node, if its Counter field is greater than the watermark, then the message is accepted and the watermark is updated. Otherwise, the message will be discarded. This means that the receiving node always considers the first secured RPL message from a new neighbor node as non-replayed. The subsequent messages will be processed only if the Counter field is greater than the watermark value.

An example of watermark initialization with light-security configuration is shown in Figure 7. For reference, Table 1 summarizes our notations. We assume that Node B does not have any watermark. Upon receiving a new message from Node A with Counter value equal to 3, Node B initializes the watermark for Node A to the received Counter value. Then, Node B receives a new message from Node A with Counter value equal to 4. In this case the Counter value is greater than the watermark, so the watermark is updated to the Counter value.

3.4. Full-Security Configuration

The full-security configuration provides for integrity and confidentiality, in the same way as the light-security configuration. In addition, it provides for a complete replay protection service. In particular, when a new neighbor node is discovered, the watermark for this new node is not immediately initialized as in the light-security configuration. Instead, the watermark is initialized if the new neighbor node is able to successfully terminate an authenticated challenge-response handshake.

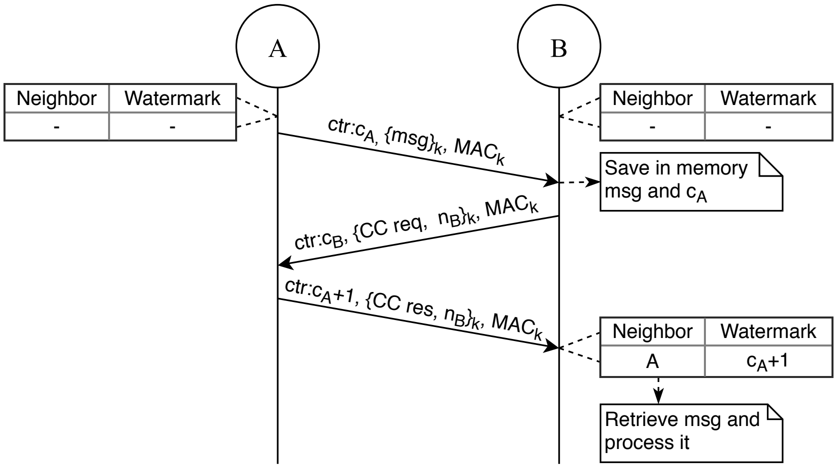

A
B
Neighbor
Watermark
-
-
Neighbor
Watermark
-
-
ctr:c$_A$, {msg}$_k$, MAC$_k$
Save in memory msg and c$_A$
ctr:c$_B$, {CC req, n$_B$}$_k$, MAC$_k$
ctr:c$_A$+1, {CC res, n$_B$}$_k$, MAC$_k$
Neighbor
Watermark
A
c$_A$+1
Retrieve msg and process it

Figure 8: CC handshake in full-security configuration.

Upon receiving a new message from the neighbor node, if its Counter field is greater than the relative watermark, the receiving node updates the watermark. Otherwise, the message will be discarded, as in the light-security configuration. With the full-security configuration, a node receiving the first secured RPL message from a new neighbor node, e.g., a Secured DIO message, initiates a *CC handshake* with this new neighbor node. Therefore, a secured RPL message from a new neighbor always triggers a CC handshake. In Figure 8 the CC handshake is presented. We assume that Node B does not have any watermark. In such handshake, Node B sends a CC request to the new neighbor, Node A, carrying a random CC Nonce n_B. Upon receiving the CC request, Node A answers with a CC response carrying the same CC Nonce n_B. Note that the CC response is authenticated. Upon receiving the CC response, Node B checks that the CC Nonce is the same as that transmitted in the CC request. If this is the case, Node B initializes the watermark for Node A to the value of the CC response Counter field $c_A + 1$. Note that the message which triggered the CC handshake is saved in the memory of Node B and it is processed only if the handshake successfully terminates.

With the full-security configuration, the random number generator (RNG) initialization takes a crucial role in the CC handshake. This is particularly true after a node reboot. In this situation, the rebooted node loses all the stored watermarks. So, it must run again all the CC handshakes with all its neighbors. If the RNG is not secure, it could generate the same CC Nonce

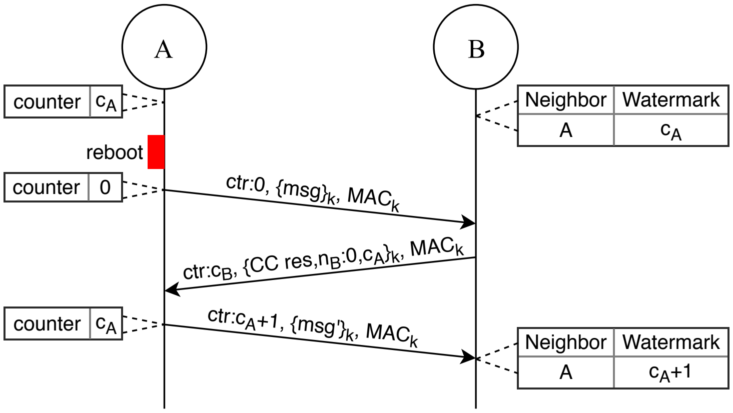

A
B
counter c_A
Neighbor Watermark
A c_A
reboot
counter 0
ctr:0, $\{msg\}_k$, MAC_k
ctr:c_B, $\{CC\ res, n_B{:}0, c_A\}_k$, MAC_k
counter c_A
ctr:c_A+1, $\{msg'\}_k$, MAC_k
Neighbor Watermark
A c_A+1

Figure 9: Resynchronization procedure in full-security configuration.

sequence used before the reboot. In this situation, an adversary can replay an old CC response to the rebooted node and successfully complete a CC handshake. At this point, the rebooted node has a watermark for the adversary and it will accept all the messages replayed by the adversary. In order to realize a good RNG on a constrained device, a viable solution could be to use the bit error rate of the wireless radio channel, which guarantees an acceptable source of entropy [17].

3.4.1. Resynchronization Mechanism

In a low-power wireless network it may occur that a node reboots, as result of battery shortage. In this case, the node resets its current Counter value, and its transmitted messages will start again from a zero Counter field. As a consequence, the messages will be discarded by the neighbors as possible replays. To recover from this situation, we implement a *resynchronization* mechanism. It is important to highlight that in our implementation we suppose to use nodes which could not have a persistent memory. With this situation, a node loses its current Counter value if it accidentally reboots and so it needs the resynchronization to retrieve its Counter values from its neighbor nodes.

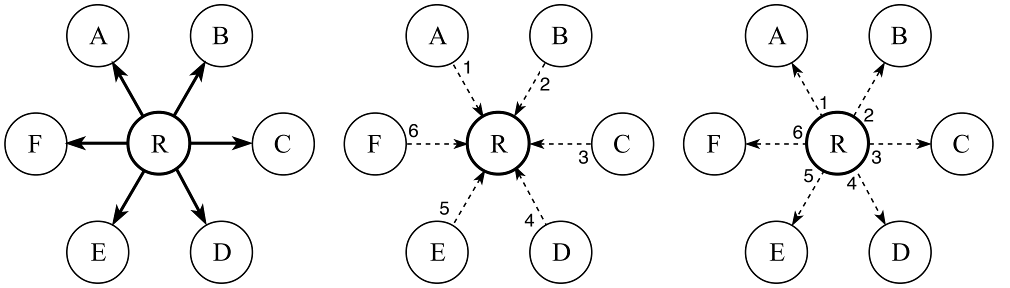
Figure 9 shows the resynchronization mechanism. After rebooting, Node A starts sending secured RPL messages, e.g. multicast Secured DIS messages, with a Counter field equal to zero. If some neighbor, e.g. Node B, is storing a watermark for Node A and receives this secured message, then Node B sends a CC response to the rebooted node carrying a Destination Counter field set to the watermark value c_A and the CC Nonce n_B set to zero. Upon receiving a CC response with CC Nonce set to zero, Node A checks whether its counter is less than the Destination Counter field. If this is the case, Node A updates its own counter to c_A, otherwise the counter remains unchanged.

Note that it is possible for an external adversary to mount a selective denial of service attack during the resynchronization process, due to the fact that the rebooted node does not have a proof of the freshness of the received CC response, i.e. it cannot determine whether the CC response sent in the resynchronization process is a replay or not. An external adversary could thus replay the CC response of resynchronization to force a victim node to install a counter value less than the watermarks scattered in the network. This attack leads to a selective denial of service because all the messages sent by the victim node are discarded by the honest nodes, as their watermarks are greater than the counter values used by the victim node. Note that the adversary must also block somehow, for example by jamming them, all the CC responses of resynchronization sent by the honest neighbors of the victim. Otherwise, the victim would resynchronize with the correct counter value. A possible solution to thwart this attack is to simultaneously perform a challenge-response handshake during the resynchronization procedure. In particular, a rebooted node includes a nonce value in every secured DIS message. The other nodes consequently would include the same nonce value in their CC responses, so that the rebooted nodes would be ensured of the freshness of the CC response. Of course, this solution is out of standard, as the DIS message format does not provide for a nonce field.

3.4.2. Critical Procedures

In the full-security configuration some critical routing procedures are particularly influenced by CC handshakes. We identified as critical the *network formation* and *downward routes formation* procedures, which sensibly change with the full-security configuration.

Network Formation Procedure. The *network formation* procedure is intended as the procedure in which every node in the network starts from scratch. The bootstrap is considered terminated when every node *joins* the network, i.e., every node obtains a route towards the DODAG root. In the full-security configuration, the complexity of the network formation significantly increases due to the CC handshake. A typical scenario of the network formation is shown in Figure 10. At the beginning no node has watermarks. Therefore,



(a) First phase: multicast Secured DIO emission

(b) Second phase: multiple unicast CC requests

(c) Third phase: multiple unicast CC responses

Figure 10: Example of network formation with 6 non-root nodes and full-security configuration. The solid arrows represent a multicast message. The dotted arrows represent unicast messages. The numeric labels temporally sort the message transmissions.

they must run the CC handshake for each discovered neighbor. The root node multicasts a Secured DIO, which is received by multiple nodes, which have just booted (Figure 10a). All nodes save the Secured DIO message in memory without processing it, waiting for assessing whether it is a replay or not. To do this, they immediately initiate a CC handshake with the root node, as in Figure 10b. Upon receiving multiple CC requests, the root node answers to all CC requests with the appropriate CC response to all nodes, as shown in Figure 10c. Finally, every node checks whether the CC response Counter field is greater than the Counter field of the stored Secured DIO message. If this is the case, they can assess that the Secured DIO was not replayed, so the new nodes can choose the root node as their preferred parent. Note that, from now on, all nodes have a watermark for the root node, but the root node does not have a watermark for any of them. It is important to notice that the network formation procedure with the light-security configuration consists only in the first phase (Figure 10a), i.e., all nodes immediately join the DODAG upon receiving a Secured DIO.

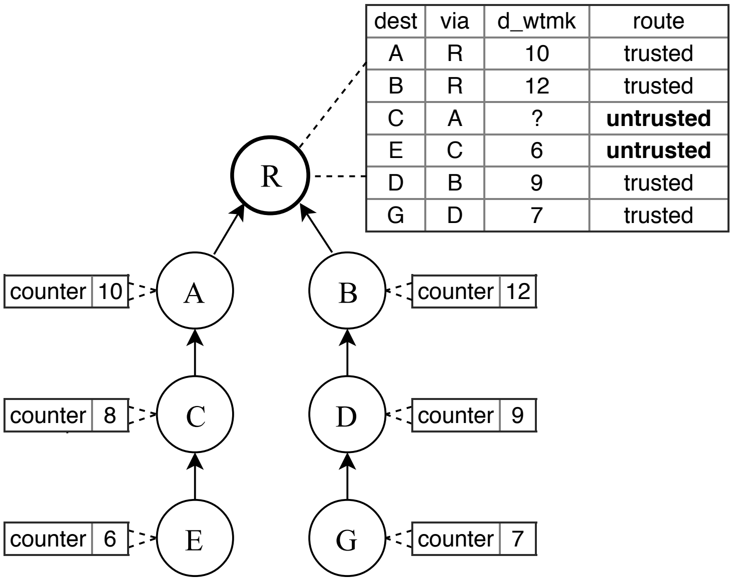
Note that a node can be added to the network after the network formation procedure is complete. In this scenario the new node will follow the same steps described above for initializing the watermarks, and for joining the network.

Downward Routes Formation Procedure. In the case the network needs downward traffic support, DAO messages are used for building downward routing tables. Every node sends a unicast DAO message in order to propagate downward routes up to the DODAG root. As mentioned before, RPL specifies two modes of operation, storing mode and non-storing mode.

In the storing mode, every node sends a unicast Secured DAO message to its preferred parent, which processes it, i.e. it stores the route towards its child node in the routing table, and forwards a Secured DAO message to its preferred parent adverting the same received route. This process reiterates until the Secured DAO message is forwarded to the root node, which eventually answers with a Secured DAO-ACK back to the DAO originator. Note that in the forwarding chain of these Secured DAO messages, it may happen that a parent node does not have a watermark for its child. In this case, the parent node must run a CC handshake in order to assess if the received Secured DAO message is replayed or not. The Secured DAO-ACK message, instead, does not trigger a CC handshake because it is forwarded from a parent node to its child node, which always has a watermark for its preferred parent due to network formation procedure.

In the non-storing mode, every node sends unicast Secured DAO messages to the DODAG root, which in general does not have a watermark for any DAO sender. DAO messages are sent end-to-end, i.e., intermediate nodes do not process this message, they simply forward it to their preferred parent. In the non-storing mode, every node advertises in the DAO message its own IPv6 global address and its preferred parent IPv6 global address. Upon receiving a Secured DAO message, the root node may not have a watermark for the DAO sender, then it may be unable to assess whether such a Secured DAO was replayed or not. In this case, the root node and the DAO sender must run a CC handshake, which in turn requires a downward route towards the DAO sender. The root node obtains such a route from the Secured DAO information, and it flags the route as *untrusted*. An untrusted downward route is a temporary route, used only to send CC requests. After the CC handshake has been successfully completed, the root node creates a watermark for the destination node. Only when the root node has a watermark for the destination node and for all the intermediate nodes, then the downward route can be flagged as *trusted*, and it can be used to route any type of message. In this way, the root is sure that all the routes stored in its downward routing table have been built from non-replayed DAO messages.

In Figure 11 an example of downward routing table stored by the root



dest	via	d_wtmk	route
A	R	10	trusted
B	R	12	trusted
C	A	?	**untrusted**
E	C	6	**untrusted**
D	B	9	trusted
G	D	7	trusted

Figure 11: Downward routing table stored by the root node with non-storing mode and full-security configuration. The root node does not have a watermark for Node C because the CC handshake between them did not complete successfully. As a consequence, routes for Node C and Node E are untrusted.

node is shown. We assume that the root node already has a watermark for Nodes A, B, D, E and G. Node C, instead, did not complete the CC handshake with the root node, so the root node does not have a watermark for Node C yet. In this situation, the routes for Node C and for any node in Node C subtree, e.g. Node E, are flagged as untrusted. Therefore, the root node can send only CC requests to these nodes until it completes a CC handshake with Node C.

4. Impact of Security

In order to evaluate the overhead introduced by the RPL security mechanisms, we run a performance evaluation based on simulations. To this aim, we considered three different RPL modes: the unsecured mode, which corresponds to the “vanilla” Contiki RPL implementation [9]; the preinstalled mode with light-security configuration and the preinstalled mode with full-security configuration.

Simulations have been run exploiting COOJA [18], a network emulator which is available as part of the Contiki distribution [19]. COOJA emulator

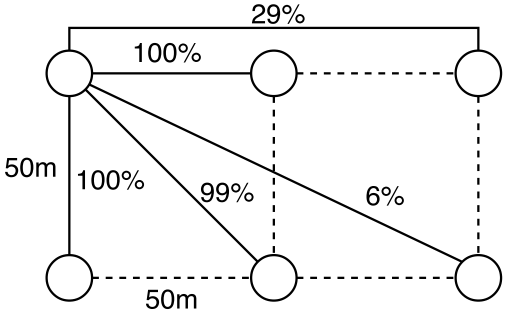


Figure 12: Packet Delivery Ratio (PDR) at different distances. Greater distances than those displayed have PDR=0%.

provides a realistic simulation environment in which wireless nodes are emulated allowing to run the same binary image that would be executed on real nodes. In our simulations, COOJA has been configured to simulate a network of *Cooja motes*, i.e., a generic sensor hardware equipped with an IEEE 802.15.4 radio interface. Although COOJA allows the emulation of real hardware, Cooja motes have been adopted to overcome the limitations in terms of memory and computation capabilities. Wireless channel has been simulated using the Multi-path Ray-tracer Medium (MRM) model, which implements a realistic radio propagation model [20]. In order to assess the performance of the RPL protocol in networks of different sizes, a regular grid topology with an increasing number of nodes has been considered. Specifically, 8x8, 11x11 and 14x14 grids have been considered. The distance between the nodes has been fixed to 50m in all the topologies. MRM channel parameters have been set in order to result in a Packet Delivery Ratio (PDR) of 100% at such distance. The PDR at other distances is reported in Figure 12. The node at the top-left corner has been configured to behave as DODAG root. Simulations have been run for 30 minutes. In order to obtain statistically sound results, we used the independent replication method, with a confidence level of 95%. Specifically, for each experiment we ran 32 independent replications. All the RPL settings have been configured according to the Contiki default parameters. The radio duty cycle algorithm adopted by each node is ContikiMAC [21]. The cryptographic algorithm has been set to CCM with the AES-128 software implementation provided by Contiki OS.

4.1. Evaluation Criteria

The impact of the RPL security mechanisms on the network operations is assessed through the following metrics.

- *Network formation time*, defined as the time between the beginning of the experiment and the time at which the last node joins the DODAG. This metric measures the time required by the network to become fully operational.

- *Routes construction time*, defined as the time between the beginning of the experiment and the time at which the root node has a route for all non-root nodes. This metric measures the time required by the network to guarantee both upward and downward communications.

- *Number of RPL messages exchanged*, defined as the overall number of RPL control messages sent in 30 minutes of simulations by all the nodes in the network. This metric is adopted to assess the overhead, in terms of additional messages introduced by the security mechanisms.

Unfortunately, it is not possible to measure the node power consumption with COOJA emulator in a sufficiently realistic manner with Cooja motes. In place of that, in Section 6 we report measurements of node power consumption in a real IoT testbed.

4.2. Simulation Results

In Figure 13 the *Cumulative Distribution Function* (CDF) of the network formation time for an 11x11 grid topology in the storing mode is reported[2]. Table 2 compares the average values, with 95%-confidence intervals, of the network formation time with different network sizes and the different security modes.

As it can be seen in Figure 13 and in Table 2, in unsecured mode and in light-security configuration the network formation time does not change. In other words, just the usage of encrypted and authenticated RPL messages does not influence the network formation time. Instead, the introduction of the replay protection mechanism (full-security configuration) sensibly shifts

[2] In the non-storing mode, the CDF does not change noticeably so we omit it in the paper for the sake of brevity.

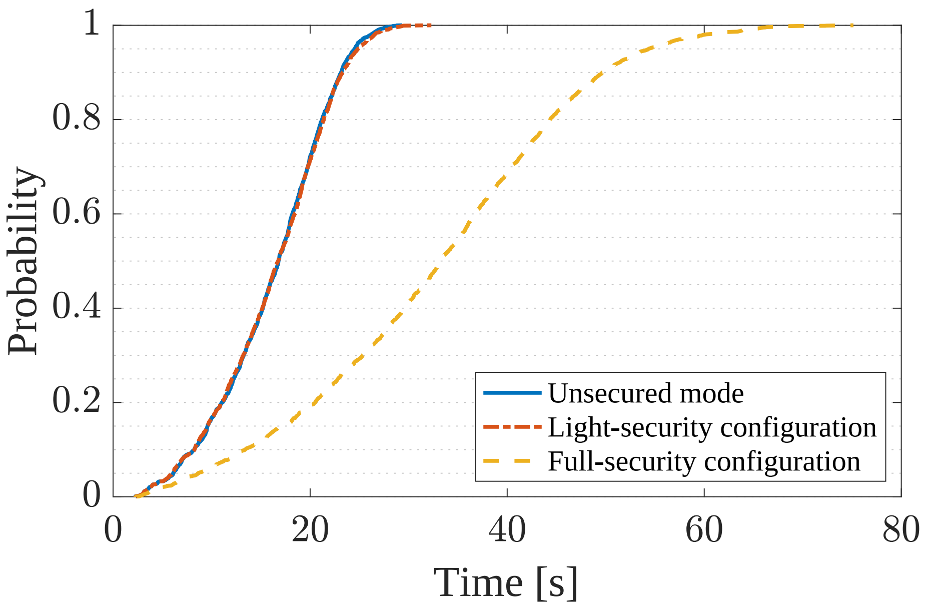


Figure 13: CDF of network formation time in storing mode, with 11x11 grid topology.

Table 2: Average network formation time, with 95%-confidence intervals.

Network formation time		
Topology	Security mode	Values [s]
8x8 Grid	Unsecured Mode	19.9 ± 0.5
	Light-security Configuration	19.8 ± 0.5
	Full-security Configuration	43.0 ± 1.6
11x11 Grid	Unsecured Mode	27.0 ± 0.6
	Light-security Configuration	27.7 ± 0.6
	Full-security Configuration	61.5 ± 2.5
14x14 Grid	Unsecured Mode	35.0 ± 0.8
	Light-security Configuration	34.9 ± 0.7
	Full-security Configuration	76.0 ± 2.4

the CDF to the right, which means that the network formation time noticeably increases. In both unsecured mode and light-security configuration the required time for the network to be fully operational in the 11x11 grid topology is around 27 seconds. With the full-security configuration instead, the required time increases to 62 seconds. This can be explained considering that every node needs to complete at least one CC handshake before joining

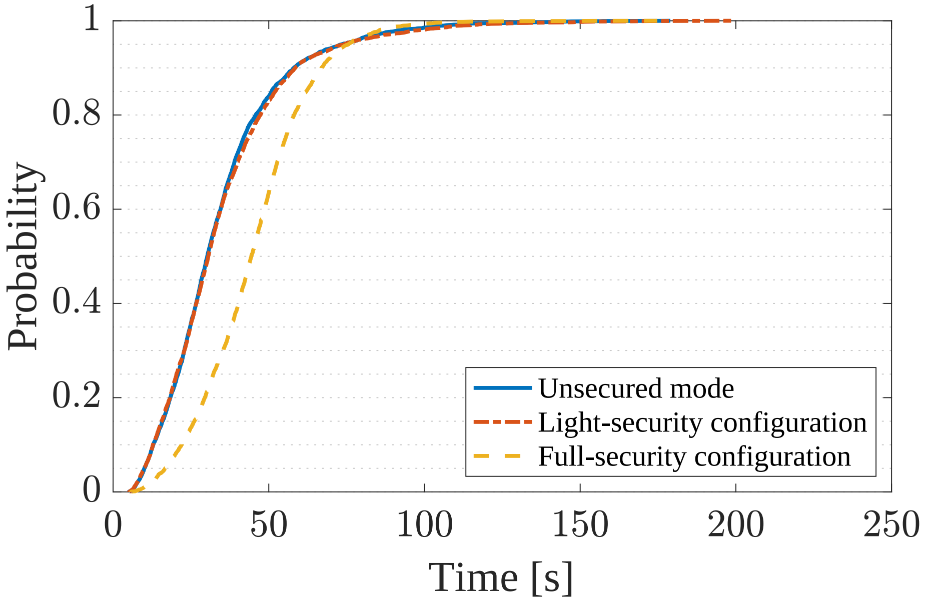


Figure 14: CDF of routes construction time in storing mode, with 11x11 grid topology.

Table 3: Average routes construction time with both storing mode and non-storing mode.

Routes construction time			
Topology	Security Mode	Storing mode [s]	Non-storing mode [s]
8x8 Grid	Unsecured Mode	59.4 ± 3.9	72.7 ± 5.2
	Light-security Configuration	59.9 ± 4.5	72.4 ± 4.8
	Full-security Configuration	59.5 ± 5.6	74.3 ± 3.3
11x11 Grid	Unsecured Mode	124.2 ± 9.1	169.1 ± 10.6
	Light-security Configuration	132.4 ± 9.8	194.3 ± 13.8
	Full-security Configuration	103.6 ± 7.6	191.5 ± 12.1
14x14 Grid	Unsecured Mode	248.3 ± 13.5	343.4 ± 14.9
	Light-security Configuration	275.5 ± 17.9	482.0 ± 53.2
	Full-security Configuration	204.2 ± 10.7	421.8 ± 33.0

the network.

In Figure 14 the CDF of the routes construction time in the storing mode for an 11x11 grid topology is reported. Table 3 compares on the first columns the average values of the routes construction time in the storing mode with different network sizes and the different security modes.

As expected, the usage of encrypted and authenticated RPL messages

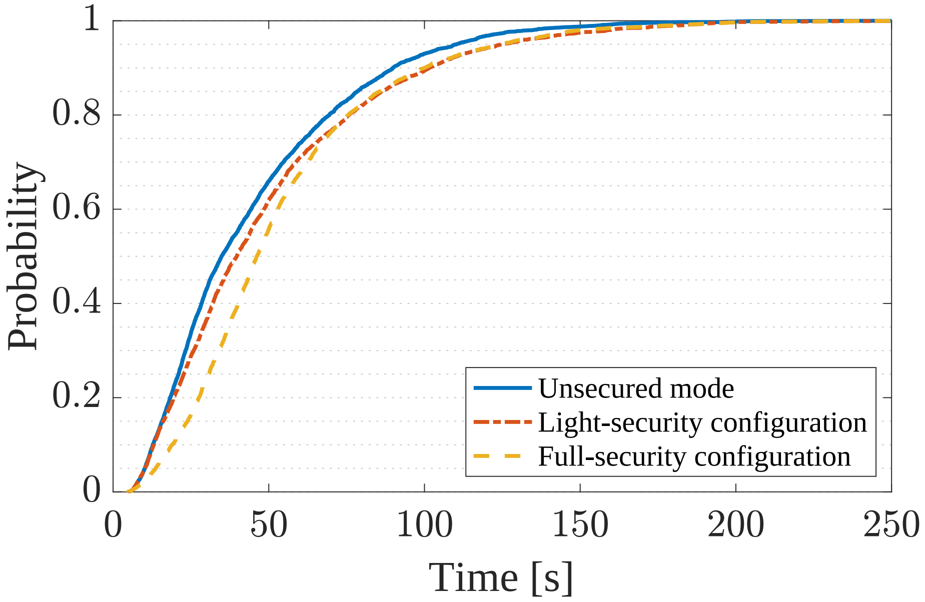


Figure 15: CDF of routes construction time in non-storing mode, with 11x11 grid topology.

only does not influence the routes construction time, as in the network formation time. With the full-security configuration, the downward routes formation procedure undergo a slight delay at the bootstrap. This can be explained considering that a node starts advertising its own global IPv6 address only when it joins the network.

In Figure 15 the CDF of the routes construction time in the non-storing mode for an 11x11 grid topology is reported. Table 3 compares on the second column the average values of the routes construction time in the non-storing mode with different network sizes and the different security modes. With respect to Figure 15, in the non-storing mode, the usage of the light-security configuration slightly increases the routes construction time. The reason for this is due to the 6LoWPAN fragmentation mechanism [22]. With the light-security configuration and non-storing mode, the Secured DAO message length is 131 bytes, which exceeds the 802.15.4 maximum transmission unit (MTU) of 127 bytes [23], so the 6LoWPAN packet fragmentation mechanism is needed. In a 6LoWPAN multi-hop scenario, a fragmented packet is reassembled at every hop and it is fragmented again when it is forwarded to the next hop. The Secured DAO message delivery to the DODAG root is

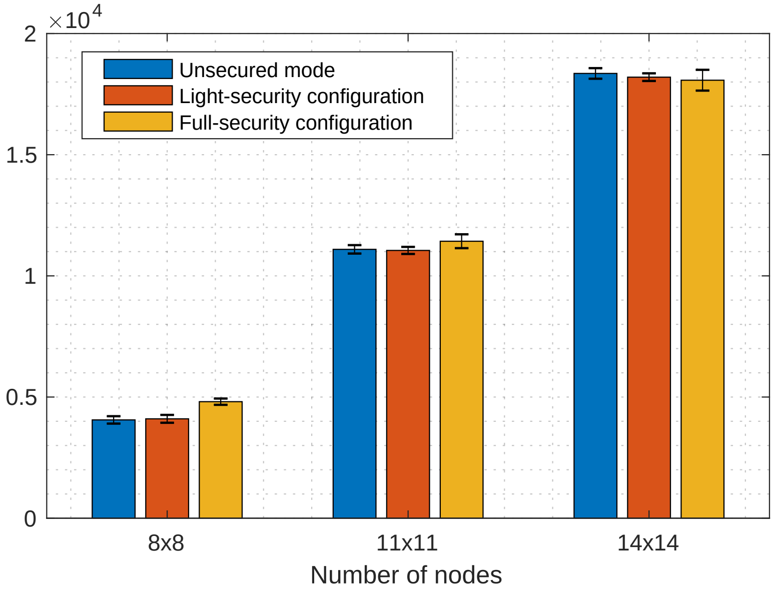


Figure 16: Average total number of RPL control messages at time T=90s with storing mode. 95%-confidence intervals are displayed in error bars.

delayed because of this reassembling process, which is not performed in the unsecured mode. With the full-security configuration, the routes construction time initially increases due to the long network formation time. However, the full-security configuration presents the same trend of the light-security one after 70 seconds, i.e., when all nodes have joined the network. Therefore, once the network formation procedure terminates, only the 6LoWPAN fragmentation mechanism influences the routes construction time.

Figure 16 illustrates the average number of RPL messages exchanged until time T=90s. The reported value includes DIO, DIS, DAO and DAO-ACK messages, and also the CC messages exchanged in the full-security configuration. As expected both unsecured mode and light-security configuration have similar number of RPL control messages exchanged. On the other hand, in the full-security configuration the number of RPL messages exchanged is quite similar to the other two configurations, although the count includes CC messages. This is due to the RPL approach for link quality estimation[24, 25]. In particular, RPL adopts a passive link monitoring approach, i.e., existing unicast data traffic is exploited to measure the link quality of wireless links. With the unsecured mode and light-security configuration, every node can monitor the link quality of the wireless link with its preferred parent when an unicast upstream message is sent. If the link has a low PDR, after experiencing a number of failed message transmissions, the node will change preferred parent in order to get a parent with a higher-PDR link. In this case the Trickle timer resets and the node sends lots of RPL messages to advertise its new routing choices to its neighbors. On the other hand, with the full-security configuration, the first preferred parent that the nodes choose enjoys a higher PDR. This is because every node processes Secured DIOs only after a CC handshake with the DIO sender, but such CC handshake probably fails when performed over a low-PDR link. If such CC handshake fails, the node will avoid choosing the low-PDR DIO sender as its preferred parent. By choosing instead a "good" preferred parent (i.e., a preferred parent with a high-PDR link), nodes avoid to change it afterwards, and they save the transmission of the relative RPL messages.

In order to corroborate this explanation, we measured the *network stretch* metric [26]. The network stretch is defined as the fraction of nodes having a sub-optimal route towards the DODAG root, i.e., a route which noticeably differs from the optimal route in terms of cumulative link quality estimations. Routes optimality is measured in RPL through *Expected Transmission Count* (ETX) metric [27]. With the ETX metric, a route is considered optimal if every link over this route has the highest PDR. In a RPL network sub-optimal routes towards DODAG root indicate that nodes choose preferred parents with low-PDR link on average.

Figure 17 illustrates the network stretch measured at the time T=90s, i.e., when the network formation procedure is terminated for every considered configuration and any network size. Note that low network stretch values mean that the number of nodes which choose choose preferred parents with low-PDR link on average reduces. As expected, the network stretch increases with the network size. With the light-security configuration the network stretch remains the same as the unsecured mode. On the other hand, by introducing the CC handshake we noticeably decrease the network stretch. For example the full-security configuration reduces the network stretch by around 33% with the 14x14 grid topology, from 0.46 to 0.31. These results show that, in the full-security configuration, nodes choose "good" preferred parents. So they avoid to change the preferred parents afterwards, and they save the transmission of the relative RPL messages. Furthermore, the results obtained by the network stretch metric explain why the routes construction time values in Table 3 are lower with the full-security configuration respect

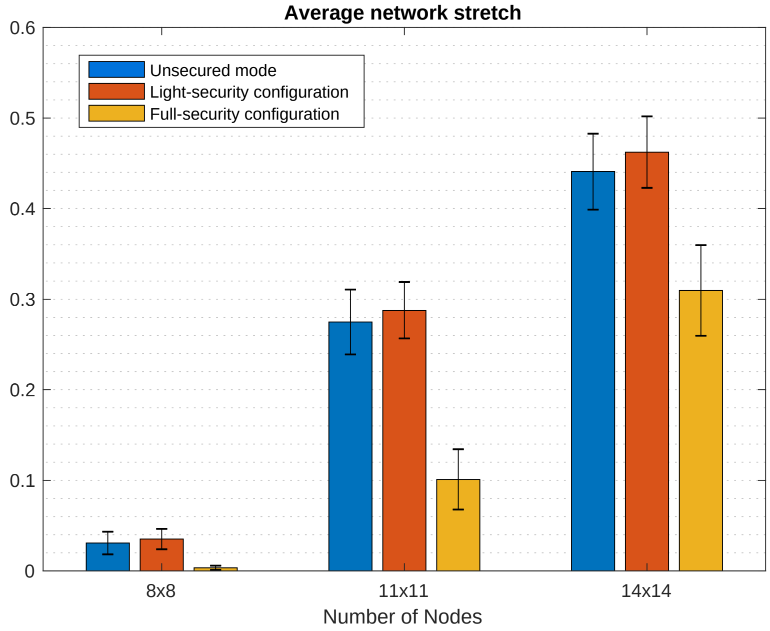


Figure 17: Average network stretch at time T=90s.

to the other two security modes on average.

The simulation results presented above show that the light-security configuration does not have a relevant impact on the critical network procedures, i.e., network formation and downward routes formation procedures. As the light-security configuration offers security properties such as confidentiality, authentication and a lightweight replay protection without degrading critical network procedures, we suggest to always use the light-security configuration instead of the unsecured mode. Moreover, we highlight that the full-security configuration offers a complete replay protection mechanism and improves routes optimality, at the cost of a slight decrease of the network formation time. We then recommend to use as much as possible the RPL protocol with the full-security configuration.

5. Optimized Full-Security Configuration

In Section 4 we showed that the full-security configuration suffers from high network formation time due to CC handshakes. In order to reduce the impact of the CC handshakes, in this section we propose an *optimized*

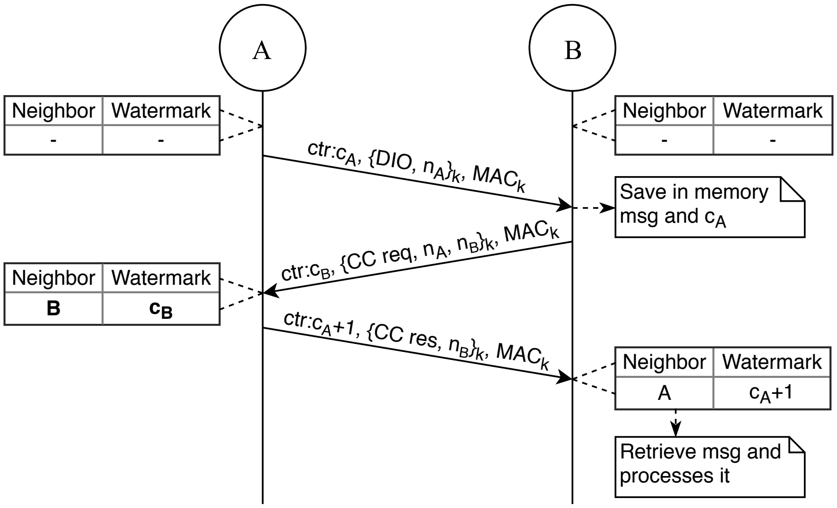

A
B
Neighbor | Watermark
- | -
Neighbor | Watermark
- | -
ctr:c_A, {DIO, n_A}$_k$, MAC$_k$
Save in memory msg and c_A
ctr:c_B, {CC req, n_A, n_B}$_k$, MAC$_k$
Neighbor | Watermark
B | c_B
ctr:c_A+1, {CC res, n_B}$_k$, MAC$_k$
Neighbor | Watermark
A | c_A+1
Retrieve msg and processes it

Figure 18: CC handshake with optimized full-security configuration.

full-security configuration. This optimized full-security configuration provides for integrity, confidentiality and a complete replay protection service, in the same way as in the full-security configuration, however, it reduces the number of CC handshakes between neighbors, thus revealing in a reduced network formation time. To do so, we extend the Secured DIO format and the CC format in order to initialize *two* watermarks with a single CC handshake: one at the DIO receiver and the other at the DIO sender. As a result, the network formation globally needs less CC handshakes, and it is simpler, quicker, and less energy-consumptive. With the optimized full-security configuration, every node includes a random nonce inside Secured DIO messages. In Figure 18 the CC handshake with optimized full-security configuration is presented. We assume that both Nodes A and B do not have any watermark. Node A multicasts a Secured DIO message carrying a random nonce, $\mathrm{n_A}$, to its neighbors. Upon receiving this message, Node B starts a CC handshake with Node A. The CC request carries the same nonce of the received Secured DIO $\mathrm{n_A}$ and another random CC Nonce $\mathrm{n_B}$. Upon receiving the CC request, Node A checks if the message carries the same nonce conveyed by the last Secured DIO message sent. If this is the case, Node A initializes the watermark for Node B with the value of the CC request Counter field. Finally, Node A answers with a CC response carrying the same CC Nonce $\mathrm{n_B}$. Upon receiving the CC response, Node B checks if the CC Nonce $\mathrm{n_B}$ is the same

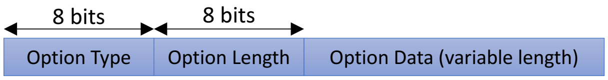


Figure 19: Generic RPL Option format.

as that transmitted in the CC request. If this is the case, Node B initializes the watermark for Node A with the value of the CC response Counter field. At the end, a single CC handshake has initialized two watermarks: one at Node A for Node B, and the other at Node B for Node A.

With respect to Figure 8, where Node A and Node B need 6 messages for initializing a watermark for the other node, in Figure 18 Node A and Node B need only 3 messages for performing the same procedure.

5.1. Implementation

To extend the Secured DIO and the CC formats without losing standard compliance, we implemented the optimized full-security configuration by exploiting the RPL options carried by a RPL message. In Figure 19 the format of a RPL option is shown. The Option Type field is an 8-bit identifier expressing the type of the option. Out of 255 possible values for the RPL Option Type, 10 are specified by the standard. The remaining values (starting from 0x0A) are currently reserved for future uses. We propose to use one free value to encode a new RPL option: the *RPL Nonce Option* type. The RPL Nonce option carries a 16-bit nonce value in the Option Data field.

5.2. Simulation Results

In order to evaluate the benefits of the optimization, we carried out some additional simulation experiments in the same conditions described in Section 4. We considered two configuration, namely, the full-security configuration and the optimized full-security configuration. We measured the average total number of CC messages exchanged during the experiments, whose duration is fixed and equal to 30 minutes.

In Figure 20 the average number of CC messages exchanged during the entire experiment is presented. As expected, the number of CC messages exchanged increases with the network size. This can be explained considering that the number of CC handshakes are directly proportional to the number of nodes. Regardless the network size, with the optimized full-security configuration we have a noticeable reduction of the CC messages exchanged. For

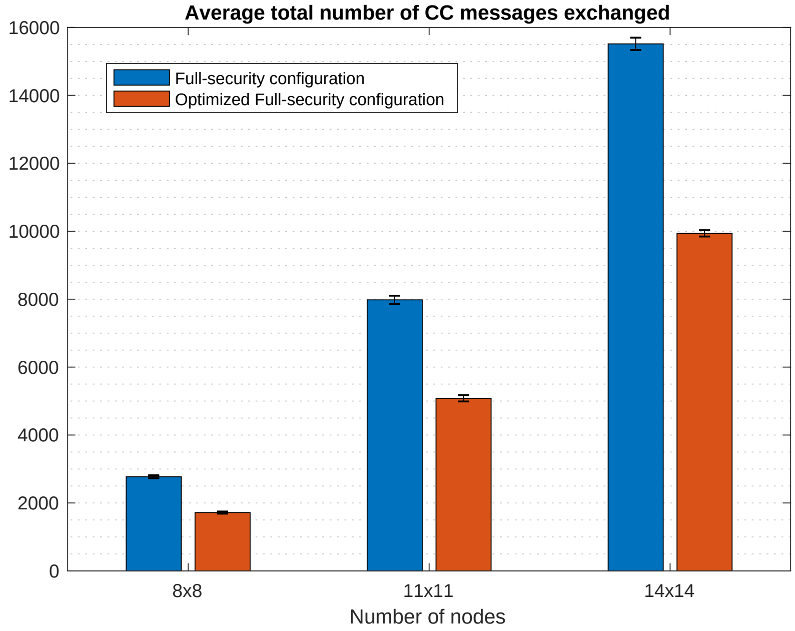


Figure 20: Average total number of CC messages exchanged.

example, with the 14x14 grid topology the metric reduces by approximately 36%.

With the reduction of the CC handshakes experienced during the network life, we expect an improvement in the network formation and routes construction times. In Figs. 21 and 22 the CDFs for network formation time and routes construction time in the 14x14 grid topology are presented. As expected, the network formation time decreases with the optimized full-security configuration. As a consequence, the routes construction time with the optimized full-security configuration decreases in both storing and non-storing modes.

Finally, Table 4 and Table 5 compare the average network formation time and routes construction times with both full-security configuration and optimized full-security configuration with different network sizes.

6. Experimental Results

Since simulation experiments might not take into account all the factors that can occur in a real environment, we also performed a set of measurements

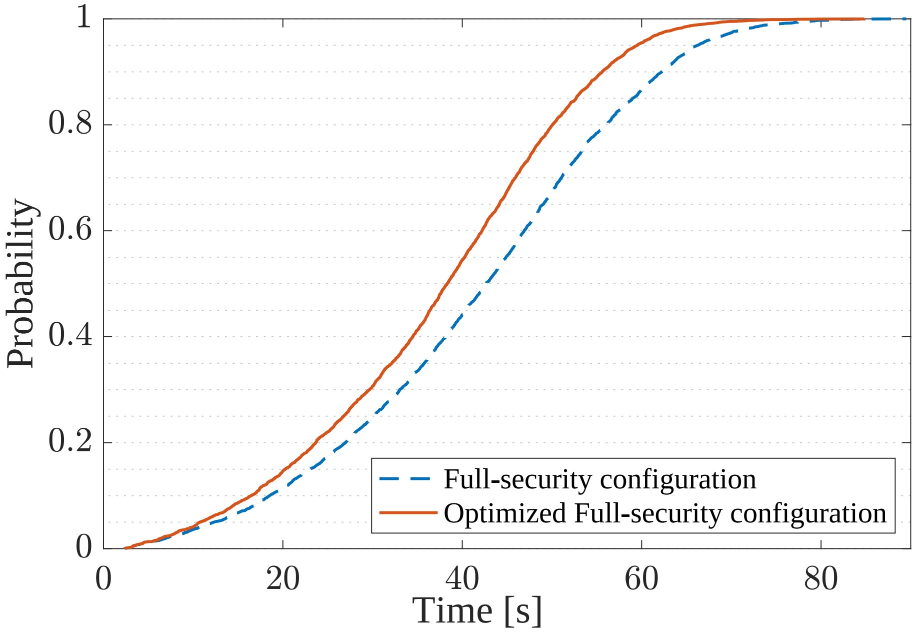


Figure 21: CDF for network formation time, with 14x14 grid topology.

Table 4: Average network formation time.

Network formation time		
Topology	Security mode	Values [s]
8x8 Grid	Full-security configuration	43.0 ± 1.6
	Optimized Full-security configuration	37.5 ± 1.6
11x11 Grid	Full-security configuration	61.5 ± 2.5
	Optimized Full-security configuration	52.9 ± 2.2
14x14 Grid	Full-security configuration	76.0 ± 2.4
	Optimized Full-security configuration	66.8 ± 2.3

on a real testbed. The purpose of this experimental analysis is twofold: (i) validating the previous simulation results, and (ii) showing that the proposed security mechanisms are viable in a real environment. For our measurements, we used the FIT IoT-LAB platform, an open large-scale and multiuser infrastructure [28]. The FIT IoT-LAB platform is a shared platform with potential concurrent experiments, thus providing a realistic environment for IoT-related systems and application experiments. We employed in our ex-

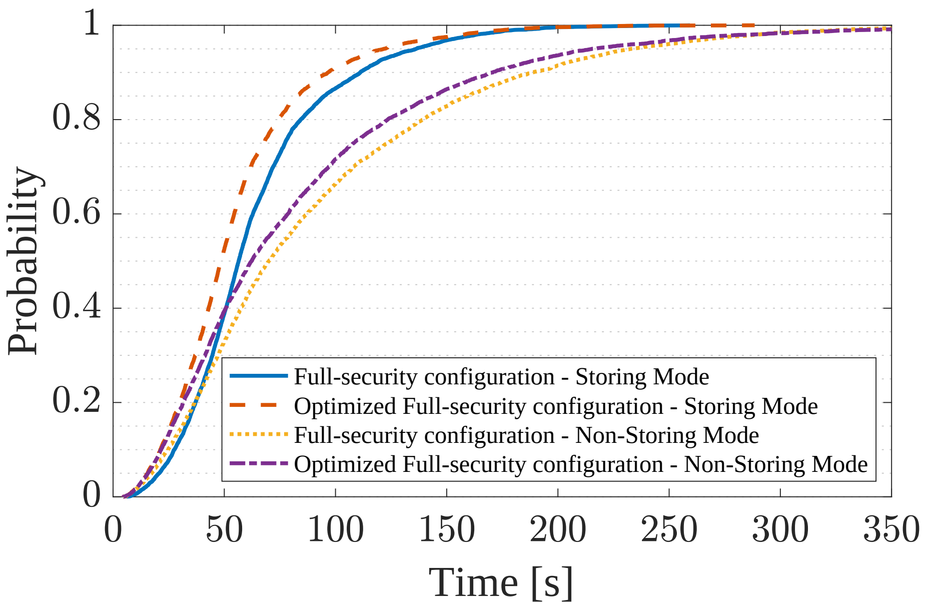


Figure 22: CDF for routes construction time in storing and non-storing modes, with 14x14 grid topology.

Table 5: Average routes construction time with both storing mode and non-storing mode.

Routes construction time			
Topology	Security Mode	Storing mode [s]	Non-storing mode [s]
8x8 Grid	Full-security Configuration	59.5 ± 5.6	74.3 ± 3.3
	Optimized Full-security Configuration	50.0 ± 2.9	69.4 ± 4.2
11x11 Grid	Full-security Configuration	103.6 ± 7.6	191.5 ± 12.1
	Optimized Full-security Configuration	87.4 ± 5.4	184.0 ± 10.9
14x14 Grid	Full-security Configuration	204.2 ± 10.7	421.8 ± 33.0
	Optimized Full-security Configuration	172.8 ± 10.1	409.7 ± 29.7

periments the M3 Open motes[3]. The M3 mote is a Cortex M3-based board with an IEEE 802.15.4-compatible Atmel AT86RF231 radio chip. The M3 mote supports ContikiOS and is able to run the same code used in the simulated environment. Furthermore, the M3 mote is equipped with a complete power monitoring tool. In particular, the M3 mote is connected on a gate-

[3] FIT IoT-LAB M3 motes website: https://www.iot-lab.info/hardware/m3/

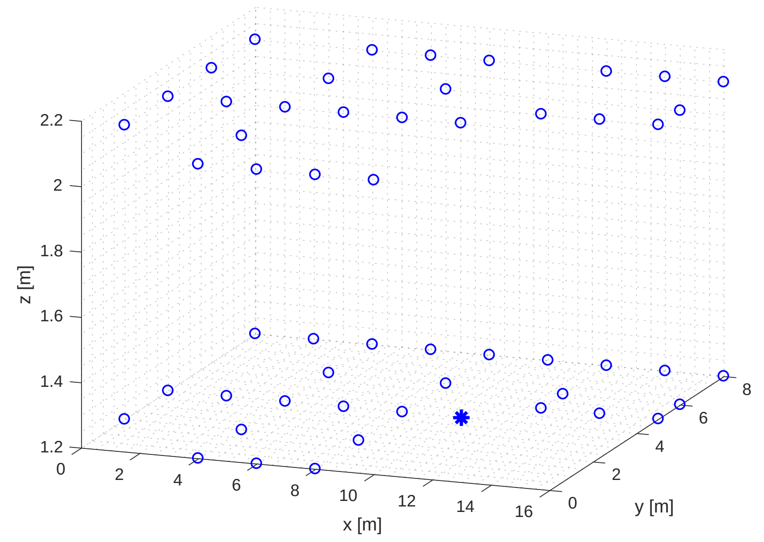


Figure 23: Topology of FIT IoT-LAB placed in Strasbourg. The starred node has been configured to behave as the DODAG root.

way which measures its consumption through resistor shunts and an INA226[4] current/power monitor component. The experiments have been run in the testbed placed in Strasbourg, using 54 nodes all placed in a 3D fixed grid, as shown in Figure 23. The starred node has been configured to behave as DODAG root in all experiments. Note that all nodes are placed in a confined space. Therefore, every node has wireless links with high PDR with every other node in the topology. Due to this, all nodes would choose the root node as preferred parent. In order to obtain a multi-hop topology, the radio transmission power is set to the lowest available value, e.g. $-17\,\mathrm{dBm}$, for every node, while messages with relative power below $-69\,\mathrm{dBm}$ are not processed. Each experiment has been run for 30 minutes. In order to obtain statistically sound results, 32 independent repetitions have been run for each experiment. All the RPL settings are configured according to the Contiki default parameters. The radio duty cycle algorithm adopted by each node is the ContikiMAC one. The cryptographic algorithm has been set to CCM with the AES-128 software implementation provided by the Contiki community.

[4] Texas Instruments INA226 website: http://www.ti.com/product/INA226

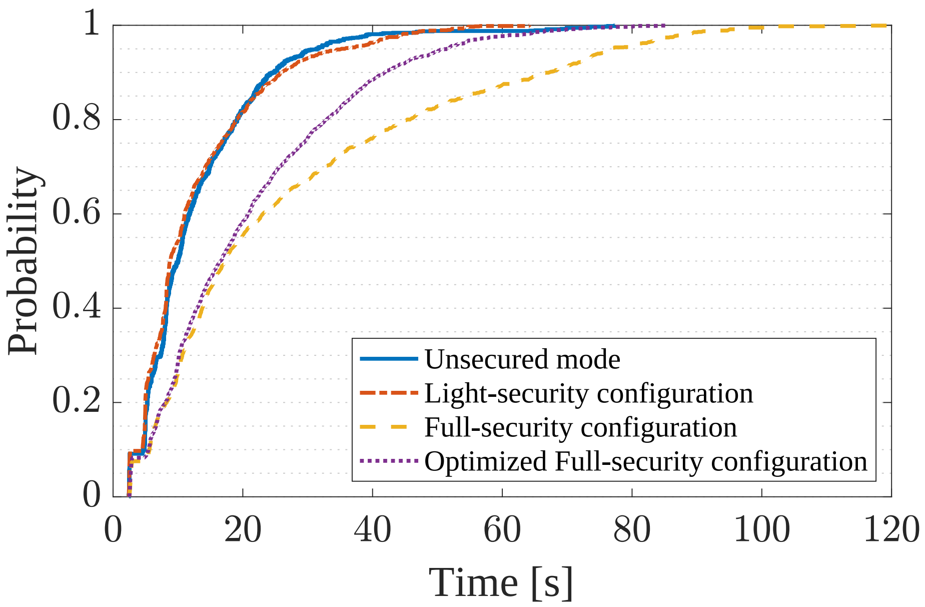


Figure 24: CDF for network formation time in the FIT IoT-LAB Testbed.

In Figure 24 the CDF for the network formation time is reported. As expected, the usage of encrypted and authenticated RPL messages (light-security configuration) does not influence the network formation time. Instead, the introduction of the replay protection mechanism (full-security configuration) increases noticeably the network formation time. With the optimized full-security configuration we obtain a relevant decrease, compared to the full-security configuration, only after 55% of nodes has joined the network. This can be explained considering that the total number of the CC messages exchanged during the network setup decreases around 45% compared to the full-security configuration. Thanks to this decrease, the network experiences a lower network overhead when a portion of the network has joined the network.

In Figure 25 the CDF for the routes construction time is reported. As expected, the full-security configuration has the highest routes construction time. Also, the light-security configuration presents the same trend of the unsecured mode. With the optimized full-security configuration, the routes construction time has a noticeable improvement during the bootstrap. This can be explained considering the improvement in the network formation time

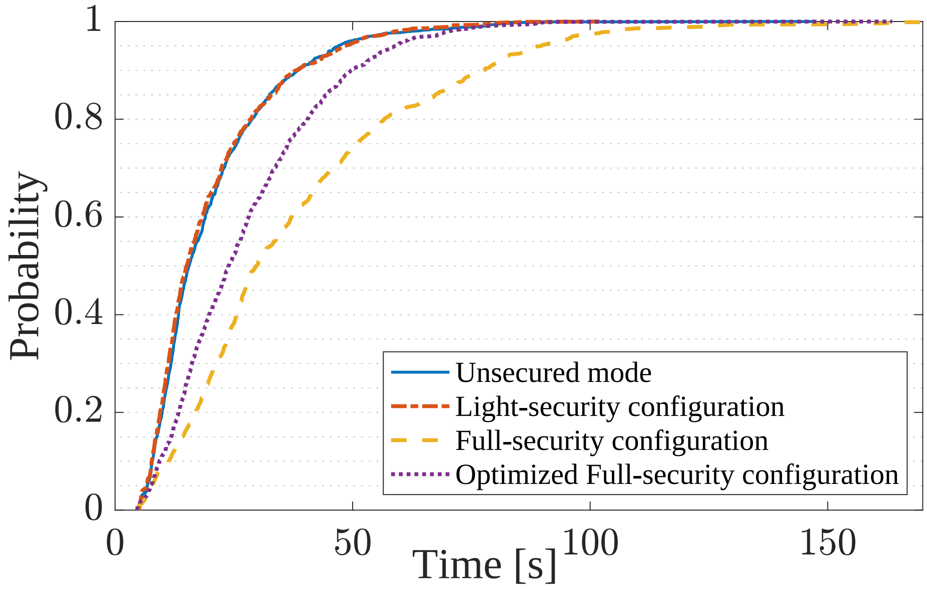


Figure 25: CDF for routes construction time in the FIT IoT-LAB Testbed.

Table 6: Average network formation and routes construction times in the FiT IoT-LAB Testbed.

Security mode	**Network formation time** [s]	**Routes construction time** [s]
Unsecured mode	38.8 ± 5.2	63.5 ± 9.3
Light-security configuration	39.0 ± 4.5	59.5 ± 7.3
Full-security configuration	61.7 ± 5.1	71.1 ± 6.7
Optimized Full-security configuration	56.5 ± 5.2	67.9 ± 9.0

and the reduction of CC messages exchanged. Therefore, with the optimized full-security configuration there is no need of performing CC handshakes with a Secured DAO message because both parent and child nodes initialize a watermark during the network formation procedure. Table 6 compares the average values of the network formation time and routes construction time in the FiT IoT-LAB testbed with every security mode.

In Figure 26 the average per-node power consumption is reported. The average power consumption includes both Cortex M3 chip and ATMEL radio chip consumption. The figure reports only the average power consumption until the time T=300 s. After that all the configurations reach the same steady value of 146.5 mW. Also, the figure reports a zoom on the power

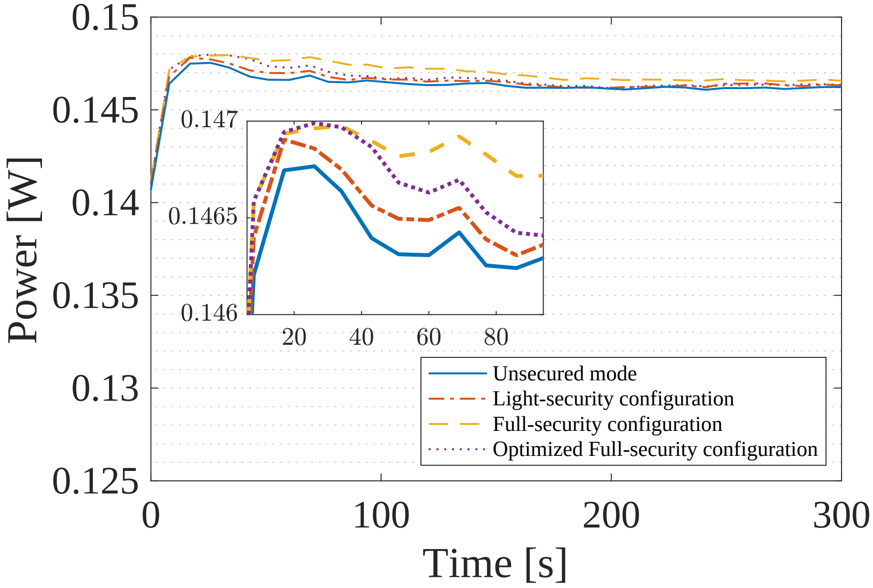


Figure 26: Average per-node power consumption in the IoT-Lab Testbed during the first 5 minutes of experiments. After 5 minutes the average power consumption stabilizes around a steady value for all configurations.

consumption during the bootstrap phase, i.e., until time T=100 s. As expected, the lowest power consumption is obtained with the unsecured mode, whereas with the light-security configuration the power consumption slightly increases. This can be explained considering the computational overhead of the cryptographic operations and the increased size of the transmitted RPL messages due to the Security Section. The highest value of the average power consumption is obtained with the full-security configuration. This is due to the high number of CC handshakes that takes place during the network formation and downward routes formation procedures. Finally, with the optimized full-security configuration, the average power consumption slightly decreases compared to the full-security configuration because of the reduction of CC messages exchanged in the network life.

7. Related Work

Many research papers [29, 10, 11, 12, 13, 14, 15] have studied possible attacks against RPL, and proposed countermeasures. Tsao et al. [29] presented a security threat analysis for networks employing RPL as routing protocol. They studied a comprehensive list of threats and attacks to RPL, which in general are applicable to any routing protocol. The authors conclude that many of the attacks can be avoided exploiting different RPL security features or the underlying MAC layer security mechanisms. Dvir et al. [11] took into consideration Rank attack and DODAG Version attack. Both these attacks can be considered as RPL-specific instances of the more general sinkhole attack [30], in which a malicious node attracts a large amount of traffic from surrounding nodes in order to eavesdrop or interrupt it. The authors presented a countermeasure to both attacks based on asymmetric cryptography. Perrey et al. [14] presented an improvement of such countermeasure which corrects some of its vulnerabilities, but requires round-trip protocols for path validation. Weekly and Pister [15] presented and evaluated the synergy between two countermeasures against sinkhole attacks in RPL: parent fail-over and rank authentication. Iuchi et al. [12] presented a countermeasure against Rank attack based on a particular next-hop selection policy. This countermeasure requires the nodes to choose sub-optimal routes. Le et al. [13] studied the impact of an attack in which a malicious node deviates from the normal behavior by selecting as next hop the worst neighbor instead of the best one. Airehrour et al. [10] proposed a countermeasure against blackhole attack, in which a malicious node drops all the traffic forwarded to it, and breaks the availability of large parts of the network. Their countermeasure requires every node to operate in promiscuous mode, and to receive and process also packets not destined to it.

All these countermeasures provide partial security, since they defend against specific attacks only, namely Rank attacks [11, 12, 14, 15], DODAG Version attacks [11, 14], blackhole attacks [10], and attacks involving next-hop selection [13]. Also, the threat model of all these countermeasures is based on external adversaries and on employing the unsecured version of the RPL protocol. All these attacks can be avoided, in case of external adversaries, simply by securing the RPL protocol. Indeed, they impede a malicious entity to become part of the network and transmit routing control messages.

Perazzo et al. [31] studied the impact of a wormhole attack, in which a malicious actor establishes and controls an out-of-band channel between two distant nodes of the network. As a result, the malicious actor can control a potentially large amount of traffic and can eavesdrop or discard it. The authors also stated that the most convenient way to counteract a wormhole attack in a WSAN may be to avoid subsequent attacks, i.e., traffic eavesdropping and selective packet dropping. Again, most of these attacks can be simply avoided by securing the RPL protocol.

Many research papers [32, 5, 33, 34, 35] implemented and evaluated the impact of security services provided by different network protocols for IoT networks. Kothmayr et al. [32] implemented a DTLS based end-to-end security architecture for the IoT. They presented an extensive evaluation based on a real IoT testbed and demonstrated the feasibility of their solution for IoT networks. Raza et al. [33] presented the first IPsec specification and implementation for IoT networks. They also evaluated the presented implementation and demonstrated the feasibility of IPsec for securing communication between sensor nodes and hosts in the Internet. Daidone et al. [34, 35] presented their implementation of the 802.15.4 security sub-layer. They also gave an extensive evaluation by means of analytical and experimental results and showed that the 802.15.4 security services have a meaningful impact on network performances. In [5] we presented a preliminary evaluation of the impact of the RPL security mechanisms on very small topologies by means of simulations. The present paper extends [5] by introducing an evaluation of the RPL security mechanisms in order to cover also downward routes formation. We evaluated the impact of the security features with both storing and non-storing modes enabled on upward and downward routes formation. The performance evaluation has been significantly extended considering larger simulation topologies and also real-world experiments. Finally, we propose a standard-compliant optimization of the RPL security features in order to be moderately lower the impact of security on all routing procedures.

8. Conclusions

In this paper, we gave an extensive evaluation of the impact of the RPL security mechanisms on the performance of large-scale network topologies by means of simulations and real experiments. We also analyzed and tackled all the aspects left unspecified by the standard in order to develop, to the best of out knowledge, the first implementation of the RPL security features publicly available. We showed that the RPL security mechanisms have a negligible impact on the performance, if they do not have to defend against replay-based attacks. Otherwise, if also a complete replay protection is needed, the power consumption remains roughly the same, while the network bootstrap time increases noticeably. On the other side, we showed that the replay protection reduces the number of RPL control messages exchanged and improves routes optimality. We also proposed a standard-compliant optimization that reduces the length of the bootstrap phase and still defends against replay-based attacks. Finally, the experiments in the FIT IoT-LAB validated the results obtained by means of simulations and confirmed that the proposed implementation is viable in a real environment. The conclusions of our analysis can be summarize as follows: (i) the basic security features included in RPL should be enabled in the majority of the deployments, considering that they have a negligible impact on the performance; (ii) the adoption of the replay protection results in an additional overhead in terms of network bootstrap, which, however, can be considered acceptable in the majority of the deployments. Whenever protection from replay attacks and low network formation is required, the proposed standard-compliant optimization can be adopted.

Acknowledgements

This work was partially supported by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence) and by the project PRA_2018_81 ”Wearable sensor systems: personalized analysis and data security in healthcare” funded by the University of Pisa.

[1] L. Atzori, A. Iera, G. Morabito, The Internet of Things: A survey, Computer Networks 54 (15) (2010) 2787 – 2805, ISSN 1389-1286.

[2] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. P. Vasseur, R. Alexander, RPL: IPv6 routing protocol for low-power and lossy networks, Tech. Rep., 2012.

[3] O. Gaddour, A. Koubâa, RPL in a nutshell: A survey, Computer Networks 56 (14) (2012) 3163–3178.

[4] A. Mayzaud, R. Badonnel, I. Chrisment, A taxonomy of attacks in RPL-based Internet of Things, International Journal of Network Security 18 (3) (2016) 459–473.

[5] P. Perazzo, C. Vallati, A. Arena, G. Anastasi, G. Dini, An Implementation and Evaluation of the Security Features of RPL, in: Ad-hoc, Mobile, and Wireless Networks: 16th International Conference on Ad Hoc

Networks and Wireless, ADHOC-NOW 2017, Messina, Italy, September 20-22, 2017, Proceedings, ISBN 978-3-319-67910-5, 63–76, 2017.

[6] P. Levis, T. Clausen, J. Hui, O. Gnawali, J. Ko, The Trickle algorithm, RFC 6206, RFC Editor, 2011.

[7] J. Hui, J. Vasseur, D. Culler, V. Manral, An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL), RFC 6554, RFC Editor, 2012.

[8] D. Whiting, R. Housley, N. Ferguson, Counter with CBC-MAC (CCM), RFC 3610, RFC Editor, 2003.

[9] N. Tsiftes, J. Eriksson, A. Dunkels, Low-power Wireless IPv6 Routing with ContikiRPL, in: 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), 406–407, 2010.

[10] D. Airehrour, J. Gutierrez, S. K. Ray, Securing RPL routing protocol from blackhole attacks using a trust-based mechanism, in: 26th International Telecommunication Networks and Applications Conference (ITNAC), 115–120, 2016.

[11] A. Dvir, T. Holczer, L. Buttyan, VeRA – Version number and rank authentication in RPL, in: IEEE 8th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS), 709–714, 2011.

[12] K. Iuchi, T. Matsunaga, K. Toyoda, I. Sasase, Secure parent node selection scheme in route construction to exclude attacking nodes from RPL network, in: 21st Asia-Pacific Conference on Communications (APCC), 299–303, 2015.

[13] A. Le, J. Loo, A. Lasebae, A. Vinel, Y. Chen, M. Chai, The Impact of Rank Attack on Network Topology of Routing Protocol for Low-Power and Lossy Networks, IEEE Sensors Journal 13 (10) (2013) 3685–3692.

[14] H. Perrey, M. Landsmann, O. Ugus, M. Wählisch, T. Schmidt, TRAIL: Topology authentication in RPL, in: International Conference on Embedded Wireless Systems and Networks (EWSN), 59–64, 2016.

[15] K. Weekly, K. Pister, Evaluating sinkhole defense techniques in RPL networks, in: IEEE 20th International Conference on Network Protocols (ICNP), 1–6, 2012.

[16] P. Perazzo, C. Vallati, G. Anastasi, G. Dini, DIO Suppression Attack Against Routing in the Internet of Things, IEEE Communications Letters 21 (11) (2017) 2524–2527, ISSN 1089-7798.

[17] A. Francillon, C. Castelluccia, TinyRNG: A Cryptographic Random Number Generator for Wireless Sensors Network Nodes, in: 2007 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 1–7, 2007.

[18] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, T. Voigt, Cross-level sensor network simulation with COOJA, in: 31st IEEE Conference on Local Computer Networks (LCN), 641–648, 2006.

[19] A. Dunkels, B. Gronvall, T. Voigt, Contiki – A lightweight and flexible operating system for tiny networked sensors, in: 29th Annual IEEE International Conference on Local Computer Networks (LNC), 455–462, 2004.

[20] M. Stehlik, Comparison of Simulators for Wireless Sensor Networks, PhD Thesis, 2011.

[21] A. Dunkels, The ContikiMAC radio duty cycling protocol, Tech. Rep., URL `http://soda.swedish-ict.se/5128/`, 2011.

[22] G. Montenegro, N. Kushalnagar, J. Hui, D. Culler, Transmission of IPv6 Packets over IEEE 802.15.4 Networks, RFC 4944, RFC Editor, URL `http://www.rfc-editor.org/rfc/rfc4944.txt`, 2007.

[23] J. A. Gutierrez, E. H. Callaway, R. Barrett, IEEE 802.15.4 Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensor Networks, IEEE Standards Office, New York, NY, USA, ISBN 0738135577, 2003.

[24] C. Vallati, E. Ancillotti, R. Bruno, E. Mingozzi, G. Anastasi, Interplay of Link Quality Estimation and RPL Performance: An Experimental Study, in: Proceedings of the 13th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor and Ubiquitous Networks, PE-WASUN '16, ACM, New York, NY, USA, ISBN 978-1-4503-4505-7, 83–90, 2016.

[25] E. Ancillotti, R. Bruno, M. Conti, E. Mingozzi, C. Vallati, Trickle-L2: Lightweight link quality estimation through Trickle in RPL networks, in:

Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014, 1–9, 2014.

[26] C. Vallati, E. Mingozzi, Trickle-F: Fair broadcast suppression to improve energy-efficient route formation with the RPL routing protocol, in: 2013 Sustainable Internet and ICT for Sustainability (SustainIT), 1–9, 2013.

[27] D. S. J. D. Couto, D. Aguayo, J. Bicket, R. Morris, A high-throughput path metric for multi-hop wireless routing, Wireless Networks 11 (4) (2005) 419–434, ISSN 1572-8196.

[28] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, T. Watteyne, FIT IoT-LAB: A large scale open experimental IoT testbed, in: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), 459–464, 2015.

[29] T. Tsao, R. Alexander, M. Dohler, V. Daza, A. Lozano, M. Richardson, A Security Threat Analysis for the Routing Protocol for Low-Power and Lossy Networks (RPLs), RFC 7416, RFC Editor, 2015.

[30] C. Karlof, D. Wagner, Secure routing in wireless sensor networks: attacks and countermeasures, Ad Hoc Networks 1 (2) (2003) 293–315.

[31] P. Perazzo, C. Vallati, D. Varano, G. Anastasi, G. Dini, Implementation of a wormhole attack against a RPL network: Challenges and effects, in: 2018 14th Annual Conference on Wireless On-demand Network Systems and Services (WONS), 95–102, 2018.

[32] T. Kothmayr, C. Schmitt, W. Hu, M. Brnig, G. Carle, A DTLS based end-to-end security architecture for the Internet of Things with two-way authentication, in: 37th Annual IEEE Conference on Local Computer Networks - Workshops, 956–963, 2012.

[33] S. Raza, S. Duquennoy, T. Chung, D. Yazar, T. Voigt, U. Roedig, Securing communication in 6LoWPAN with compressed IPsec, in: 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS), ISSN 2325-2936, 1–8, 2011.

[34] R. Daidone, G. Dini, G. Anastasi, On evaluating the performance impact of the IEEE 802.15.4 security sub-layer, Computer Communications 47 (2014) 65 – 76, ISSN 0140-3664.

[35] R. Daidone, G. Dini, M. Tiloca, On experimentally evaluating the impact of security on IEEE 802.15.4 networks, in: 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS), ISSN 2325-2936, 1–6, 2011.