Enabling Online Quantitative Security Analysis in 6LoWPAN Networks

Alex Ramos, Ronaldo T. P. Milfont, Raimir Holanda Filho, and Joel J. P. C. Rodrigues, *Senior Member, IEEE*

***Abstract*— IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) are comprised of resource-constrained nodes equipped with sensing capabilities that communicate with each other and are linked to the Internet contributing to the Internet of Things (IoT). Because of their constraints, these objects are exposed to attacks from inside the 6LoWPANs. Tipically, Intrusion Detection Systems (IDSs), are employed to identify such attacks. However, the alarms generated by IDSs provide no information about the severity of the attacks or to which extent the attacks affect the sensor data routed by the nodes. To address these issues, the present work proposes the *node security quantification* (NSQ), a probabilistic model which makes it possible to quantitatively assess, in an online manner, the security status of the constrained nodes and the sensor data users consume. The security awareness capability provided by this model can support the security-related decision-making process of both network administrators and users. The obtained simulation results demonstrate that NSQ quantifies node and data security with high accuracy, while keeping performance and energy overhead very low.**

***Index Terms*— Internet of Things, Data Security Quantification, Network Security Status.**

I. Introduction

Recent advances in networking technologies, such as the IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) standard [1], have allowed to interconnect constrained objects to the Internet, contributing to the Internet of Things (IoT) [2]. Tipically, a 6LoWPAN [3] can be characterized as a wireless sensor network (WSN) consisting of numerous sensor nodes with limited power, memory, and processing capabilities. These nodes use multi-hop communication to interact with each other and route data. 6LoWPANs can be applied to several application domains including industrial automation, military, health monitoring, home automation, among others [2].

To make this applications operational, the constrained nodes sense physical data from the operation environment and send them to a border router (BR), where the collected data is made accessible for users [4]. In many applications, the constrained nodes are deployed in open regions, which exposes them to physical capture [2]. An attacker in possession of a node can reprogram it to execute attacks against the network (*e.g.,* eavesdropping, data modification, packet dropping, etc) aiming to violate the confidentiality, integrity, or availability of sensor data from other nodes [2], [5].

Intrusion detection Systems (IDS) are commonly used to identify such attacks [6]. When an IDS detects an attack, it generates alerts and transfer them to the BR. When receiving an alert, the administrator further investigates it to decide which responsive actions should be executed [6], [5]. Nevertheless, such alerts are not sufficient to understand the actual security condition of the network, because they do not provide information specifying the impact the attacks have on the nodes and the sensor data generated. Because the amount of risk posed by the detected attacks is unknown, it is challenging for a network administrator to determine how to properly react to them. Besides, for users, it becomes difficult to rely on the collected sensor data, since they are not aware of the extent to which the data has been affected by the attacks.

To tackle such issues, this work proposes the *node security quantification (NSQ)*. NSQ is a probabilistic model that enables, in an online manner, to quantitatively assess the security status of the constrained nodes and the sensor data users consume. The proposed model provides two security metrics: the message security value (MSV) and the damage level (DL). In particular, a node's MSV corresponds to the probability that data messages generated by this node are not affected by the compromised nodes that the IDS has identified. In other words, MSV indicates how secure data messages from a given node are. Hence, a larger MSV value indicates a more secure data message. In turn, the DL metric estimates the severity of an attack by means of the probability that a malicious node compromises any given data message in the 6LoWPAN. A larger DL value implies a more dangerous malicious node.

The online security evaluation provided by NSQ is enabled by the combination of the received IDS alarms with a component called *probabilistic communication behavior graph (PBG)*. The PBG models the communication pattern of nodes by showing how messages are routed by the nodes until reaching the BR. To construct the PBG, information about the network traffic needs to be periodically transmitted to the BR, where the PBG is maintained. This is accomplished by means of the *Transmission Analysis Agent* (T2A), a distributed module installed in each node. Thus, every time the IDS generates a new alarm or the network communication pattern changes, the estimates regarding the current security condition of the network are recalculated by NSQ. Since NSQ relies on

Manuscript received Month XX, 20XX; revised Month XX, 20XX.

Alex Ramos and Raimir Holanda Filho are with University of Fortaleza (UNIFOR), CE, Brazil (e-mail: alex.lacerda@edu.unifor.br; raimir@unifor.br).

Ronaldo T. P. Milfont is with Federal University of Ceará (UFC) and Federal Institute of Education, Science and Technology of Ceará (IFCE), CE, Brazil (e-mail: ronaldotpmilfont@gmail.com).

Joel J. P. C. Rodrigues is with National Institute of Telecommunications (Inatel), Santa Rita do Sapucaí – MG, Brazil; Instituto de Telecomunicações, Portugal; Federal University of Piauí, Teresina – PI, Brazil (e-mail: joeljr@ieee.org).

the outputs of the underlying IDS, the inaccuracies of alerts have to be taken into account. This is done by combining the NSQ model with the Trust Probability metric (P_t), described in our previous publication [7].

Security metrics proposed for sensor networks already exist in the literature [8], [9], [10], [11]. In contrast to those works, the model proposed in this paper provides a more complete set of features: it is designed to be executed during network operation, it takes into account the outputs of intrusion detection mechanisms, it has been designed to run on 6LoWPANs, and its performance has been extensively evaluated, since 6LoWPANs are comprised of resource-limited nodes. In summary, this paper presents the following main contributions:

- Definition of the NSQ model and its security metrics.
- Definition and implementation of the T2A module.
- Performance and accuracy evaluation of T2A and NSQ security metrics in a realistic simulation environment.

The paper is structured as follows. Section II reviews the related work. Section III describes the system models. Section IV presents the probabilistic communication behavior graph. Section V defines proposed MSV and DL security metrics. Section VI shows the results of the performance evaluation carried out in the proposed NSQ model. Finally, Section VII presents the conclusions and future work.

II. Related Work

Assessing security in WSNs is a quite challenging task. A great deal of works in the literature quantify security in the scope of traditional networks. They typically do so by using attack graphs as underlying model [12]. However, due to WSN particularities, such works are not applicable, requiring solutions specifically developed for the WSN context.

To the best of our knowledge, just a small number of works is tailored for WSNs. In [8], the authors provided a model to measure the resilience of sensor applications to attacks against data confidentiality. However, their proposal is not prepared to assess security during run time, only before deployment.

Ramos et al. [9] presented an online approach to quantify security in WSNs. The authors propose three security metrics to assess WSN security mechanisms. However, this approach demands the use of a path addressing protocol to route packets. Thus, a higher overhead is introduced since the transmitted packets have to store the entire paths in their header.

In [10], the authors performed an enhanced version of their previous work [9], which resulted in the *S3* metric. In this metric, a Bayesian network is combined with IDS alarms to provide a probability estimation of how nodes can be affected by worm attacks. It is important to mention how NSQ differs from S3. While NSQ quantifies security by evaluating how data messages may become corrupted by compromised nodes, S3 measures security by assessing how the nodes can become malicious when receiving manipulated messages. Therefore, these models can be used together to complement each other.

III. System Models

In the present section, the system models and main suppositions are described.

A. Network Model

In this paper, a common 6LoWPAN, where the constrained nodes are fixed and repeatedly send data to the BR, is considered. To transmit messages along the network, nodes use the Routing Protocol for Low Power and Lossy Networks (RPL), which is the standard routing protocol for 6LoWPANs. The RPL protocol makes use of a destination oriented directed acyclic graph (DODAG), to route messages from the nodes to the BR. In order to transmit their messages, each node selects a *preferred* RPL parent, as Fig. 1(a) illustrates. Moreover, each node regularly changes its preferred parent based on predefined routing metrics. In both the physical layer and the link layer, the constrained nodes implement the IEEE 802.15.4 protocol [13]. In IEEE 802.15.4, nodes can be uniquely identified within a network by a 16-bit address (node ID), which is also supported by the 6LoWPAN standard. As usually considered in the literature [6], [5], the BR cannot be affected by node compromises, is physically protected against attackers, and has no resources limitation.

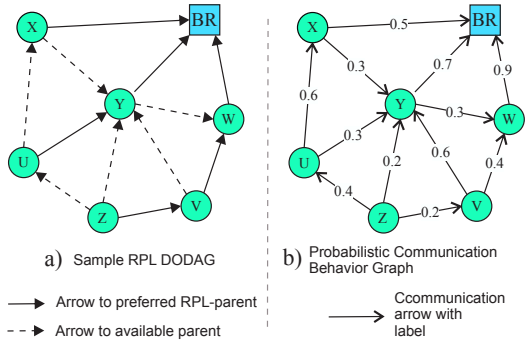


Fig. 1. Illustration of RPL-DODAG and PBG.

B. Attack Model

6LoWPANs can be target of several types of attack [14], [2]. Typically, attacks are initiated from node capture. When an attacker captures a node, he or she can reprogram the node to make it execute attacks to compromise the network. In particular, due to the multi-hop nature of sensors' communication, a malicious node can be used to compromise the sensor data (from other nodes) it forwards to the BR [5], [2]. The attack model focuses on attacks that can compromise the confidentiality, integrity, or availability of sensor data. Examples of such attacks include eavesdropping, data modification, blackhole, selective forwarding, among others [2].

C. Security Model

NSQ assumes that an IDS is present in the 6LoWPAN to find out which nodes are executing malicious tasks. There are a lot of IDSs applicable to the 6LoWPAN context [5], [6]. NSQ also assumes the existence of an end-to-end security message scheme to secure the exchange of messages between the T2A distributed module and the BR. The IDSs usually require that end-to-end security is available to protect their communications. For example, SVELTE [15], a well-known RPL-based IDS for 6LoWPANs, requires the availability of security mechanisms like DTLS [16] or IPsec [17].

IV. Probabilistic Communication Behavior Graph

As previously discussed, from time to time, a constrained node regularly selects a different preferred RPL-parent, which results in multiple possible paths from the node to the BR over time. To characterize all these paths, the PBG is used. It consists of a directed acyclic graph where nodes are modeled as vertices and edges (arrows) indicate whether pairs of nodes communicate with each other directly. The communication direction is indicated by the direction of edges. For instance, to indicate that node n_i sends data to the BR via n_j, the edge $n_i \rightarrow n_j$ is used. Moreover, each PBG edge is labeled with the probability $Pr(n_i \rightarrow n_j)$ of n_i using n_j to route its packets.

Fig. 1(b) shows the PBG of an example 6LoWPAN. As can be seen, the chance of node U using parents X and Y as next hop is 0.6 and 0.3, respectively. This means that 60% of the time X has been the preferred parent, while Y has been the preferred parent 30% of the time. Note that the graph captures packet loss by only displaying messages delivered with success. In the given example, one can conclude that 10% of the data transmitted by U has been lost, *i.e.,* $1-(0.6+0.3)$.

A. Data to Generate the PBG

The T2A module deployed in each node has been developed to gather the probabilistic information required to represent the communication behavior of a node. This information is periodically sent to the BR in a *T2A packet*. This packet carries the node's ID, the fraction of application messages that have not been forwarded with success to the node's parents (P_{loss}), the number of parents ($\#Par$), the IDs of all parents, and the respective fraction of application messages delivered with success to each one of these parents (P_{deliv}). When the BR receives a T2A packet, it parses the packet to build the PBG. The format of the T2A packet is illustrated in Fig. 2. Note that the packet has a fixed portion with a length of 5 bytes and a variable portion consisting of blocks with size of 4 bytes.

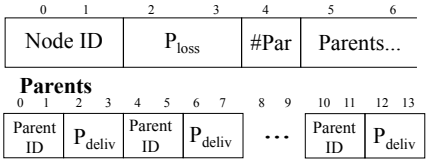


Fig. 2. T2A packet format.

B. T2A Counting Procedure

To compute the probabilities inserted into T2A packets, the T2A module monitors, at the link layer, the transmission state of the messages transmitted by the underlying node. Specifically, T2A maintains a set of counter variables containing the number of messages transmitted with success (one counter for each parent). Thus, every time a message is successfully transmitted to a given parent, its respective counter variable is incremented. In contrast, if the MAC layer fails to deliver a given message, T2A increments a variable that stores the number of messages lost.

The T2A counting task is executed during a predefined time interval ΔT_a, referred to as *active period*. This time interval can be manually configured by the network administrator according to his or her preference (Section VI empirically evaluates what would be a suitable value for ΔT_a). Once ΔT_a elapses, T2A stops its counting task and uses the counter variables to calculate all the P_{deliv} and P_{loss} probabilities. Then, a T2A packet is created and sent to the BR.

C. Keeping the PBG Up to Date

A variety of reasons can modify the topology and communication pattern of the 6LoWPAN over time, like the addition or removal of nodes. Thus, T2A periodically restarts its counting procedure to collect updated information and notify the BR if anything changes, so as to maintain the PBG always updated.

D. Inclusion of Control Packets

Because NSQ metrics are designed to evaluate the security level of sensor data, the T2A counting task only accounts for application packets, and ignores any control packets (*e.g.,* routing packets) transmitted to the RPL-parents. However, by taking control packets into account, it may be possible to reduce the active time interval ΔT_a, since a greater number of packets will be processed in a smaller amount of time. Despite this advantage, it might be possible that some "noise" is introduced in the probability values, even though only control packets sent to the preferred RPL-parent are considered. Thus, the inclusion of control packets is defined as an additional (optional) functionality.

E. Ensuring the security of T2A Packets

By manipulating a malicious node, an attacker may try to eavesdrop on or modify T2A packets while they are in transit to the BR. To address this situation, T2A can make use of end-to-end security mechanisms like DTLS or IPsec, which are usually a requirement of the IDS itself. Moreover, T2A packets can be simultaneously routed by distinct paths [18], in order to guarantee that they are successfully received by the BR, even if there is a compromised node in the network intentionally dropping T2A packets. It is important to emphasize that because T2A packets are only transmitted if the network communication behavior changes, the overhead added due to the use of multiple simultaneous paths and encryption is small.

V. Proposed Security Metrics

A. MSV Metric

The MSV for a node n_i gives the probability of any message originated from this node not becoming compromised when attacks occur. Whenever the PBG is updated or an IDS alert is raised, the MSV metric is recomputed for all nodes.

The NSQ model considers that a message is compromised if it is routed by any compromised node in its path to the BR. Considering that the probability of a message from a given node n_i getting compromised is defined as Q_i, the MSV_i for that node is defined as $MSV_i = 1 - Q_i$.

In order to understand how the value of Q_i is calculated, observe the example PBG shown in Fig. 3. In the associated 6LoWPAN, the IDS has discovered two malicious nodes, B and C. To illustrate the simplest case in the PBG, observe that messages from node D can be routed through B but never through C. Thus, Q_D can be given by $Q_{DB} = Pr(D \rightarrow B) = 0.7$, *i.e.,* the probability of path $D \rightarrow B$ occurring.

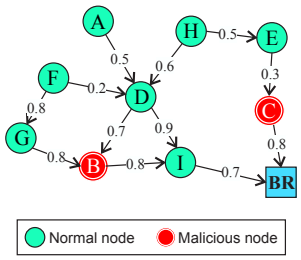


Fig. 3. Sample PBG with two nodes marked as malicious.

Now, observe that $Q_A = Q_{AB}$ and that the only path connecting A and B is $A \rightarrow D \rightarrow B$. Thus, the chance that this path occurs is the product of the probabilities of both its edges, namely, $e_1 = A \rightarrow D$ and $e_2 = D \rightarrow B$. Thus, $Q_{AB} = 0.5 \times 0.7 = 0.35$. More formally, suppose p_{ij} is a path from node n_i to n_j. The chance of p_{ij} being used is:

$$Pr(p_{ij}) = \prod_{e_m \in p_{ij}} Pr(e_m) \tag{1}$$

The malicious node B is also reachable from F; but in contrast to A, node F has two paths to B. In that case, Q_F is given by the chance of either paths $p^1_{FB} = F \rightarrow G \rightarrow B$, or $p^2_{FB} = F \rightarrow D \rightarrow B$ occurring. Hence, $Q_F = Q_{FB} = 0.64 + 0.14 = 0.78$. More formally, suppose S_{ij} is the set of paths from n_i to n_j. The chance of messages originated from n_i being forwarded by n_j can be given by:

$$Q_{ij} = \sum_{k=1}^{|S_{ij}|} Pr(p^k_{ij}) \tag{2}$$

Consider now a more complicated case where messages from a given node can be compromised by two different malicious nodes. This is the case of H, whose messages can reach either malicious nodes B or C. In such case, the probability Q_H is computing by summing the probabilities Q_{HB} and Q_{HC}. From the PBG, it is clear that $Q_H = 0.42 + 0.15 = 0.57$. To generalize, consider R_i as the set of compromised nodes that can be reached from a given node n_i. The chance that messages from n_i become compromised can be given by:

$$Q_i = \sum_{j \in R_i} Q_{ij} \tag{3}$$

Lastly, consider N as the set that contains all network nodes and M as the subset of N consisting of the compromised nodes the IDS has pointed out. The aforementioned equations can be combined to compute the probability Q_i for any given node $n_i \in N - M$, which results in the following equation:

$$Q_i = \sum_{j \in R_i \subset M} \sum_{k=1}^{|S_{ij}|} \prod_{e_m \in p^k_{ij}} Pr(e_m) \tag{4}$$

It is worth mentioning that the NSQ model considers that a compromised node only generates compromised data. More precisely, $\forall n_j \in M$, $Q_j = 1.0$.

B. DL Metric

The DL_j metric for a compromised node n_j can be given by dividing the following two values: the total amount of messages that node n_j can compromise and the total amount of messages transmitted so far in the entire network. More precisely, supposing a_i is the amount of messages transmitted by a node n_i, and G_j is the set of nodes n_i whose messages can be compromised by n_j, DL_j can be calculated as follows:

$$DL_j = \frac{\sum_{x \in G_j} Q_{xj} \cdot a_x}{\sum_{y \in N} a_y} \tag{5}$$

By providing the DL_j values of all malicious nodes $n_j \in M$ in descending order to the administrator, he or she can be aware of which nodes impose a higher impact on the security of the 6LoWPAN. These more dangerous nodes should then be prioritized when responsive actions are undertaken.

C. Handling of IDS Inaccuracy

So far, NSQ regarded the IDS alerts as being totally correct. However, false positives should be considered, since they mean that the IDS is wrong and an allegedly compromised node is in fact not compromised. To do so, a metric called Trust Probability (T) [7] is introduced in the computation of MSV. This metric gives the conditional probability that a node $n_j \in M$ is actually compromised given that alerts from the IDS have been raised against it. Based on this, the chance of data messages generated by a specific node $n_i \in N - M$ being in fact corrupted by n_j is given by $T_j \cdot Q_{ij}$. Therefore, Eq. 3 should be rewritten in the following way: $Q_i = \sum_{j \in R_i} T_j \cdot Q_{ij}$. The computation of T_j is provided in [7].

VI. Evaluation of the Proposed Approach

This section evaluates both the performance of the T2A module and the accuracy of the proposed metrics are analyzed.

A. Environment Configuration and Implementation

The T2A module has been implemented in the Contiki OS [19], a widely adopted operating system for the IoT. Contiki uses largely tested implementations of RPL (based on IPv6) and IEEE 802.15.4 protocols. For the development of the algorithms to compute the proposed metrics and construct the PBG, the C++ language has been used.

To conduct the evaluation, the Cooja simulator [20] has been used. Cooja is the official network simulator for Contiki and has proved to produce reliable results [15]. The application code executed in Cooja can be directly deployed in real devices. In the performed simulations, Tmote Sky [21] nodes

have been used. As border router, a real laptop with 4 GB of RAM and a 2.20 GHz Intel Core i5 processor has been used. The laptop was running Ubuntu 16.04 and communicated with Cooja through a serial interface.

For each simulated scenario, 10 executions have been performed in order to obtain sufficient data to compute the mean and corresponding standard deviation (error bars). In each scenario, the simulated 6LoWPAN is comprised of 30 nodes, unless otherwise specified. The nodes are randomly distributed in a rectangular area and send data to the BR every 30 seconds. The radio transmission range of nodes is $50m$. Two topologies have been evaluated, a *dense* one and a *sparse* one. In the dense topology, the dimensions of the deployment area are $150m \times 140m$, which resulted in an average neighborhood size of 10. In the sparse topology, the dimensions of the terrain are $270m$ by $240m$, resulting in a neighborhood size of 4. The use of those two topologies enables us to evaluate the scalability of the T2A module, which depends on the neighborhood sizes of nodes, rather than on the size of the network.

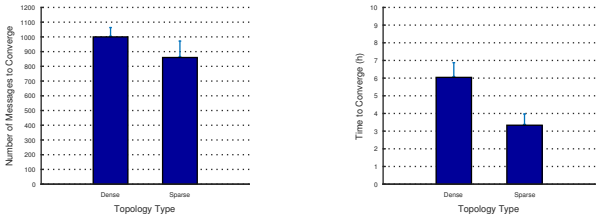
B. Convergence of PBG labels

To construct the PBG, it is important that the amount of messages counted by T2A is balanced. In other words, this amount needs to be sufficiently large so that a representative PBG is constructed; and, at the same time, this amount needs to be small enough, so that the time required to execute the counting procedure is reduced, which avoids introducing too much overhead in the constrained nodes. In this section, the convergence of the PBG's labels is analyzed, as the counting variables of T2A are incremented.

More specifically, as T2A counters are incremented, the normalized edge label updates are calculated. These updates are given in terms of the absolute difference between the current probability values and the updated probability values. The edge probabilities are considered to converge when their normalized update values start falling below a given threshold. In the evaluated scenario, this threshold is set to 1% (*i.e.,* to indicate that the normalized updates approach zero) and the edge probabilities are updated every 100 counted messages.

Fig. 4(a) shows the average number of messages T2A needs to count for the edge labels begin to converge, for each evaluated topology type. As can be seen, in the dense network, the convergence starts when the number of counted messages reaches $1,000$; while in the sparse network 860 counted messages are needed for the edge labels to convergence. The reason why the dense network requires more messages to converge is mostly because of the greater radio interference that results from the larger neighborhood sizes (*i.e.,* nodes are more close to each other), which results in a reduced message delivery rate, making the convergence slower.

The amount of time required for the edge probabilities to converge is shown in Fig. 4(b). While in the dense network the probability values take $6.04h$, on average, to converge, in the sparse network they only take $3.34h$. When comparing the results of the two different topology types evaluated in Fig 4, note that the difference between the times to converge is more expressive than the difference between the number of messages



(a) Number of counted messages required to converge.

(b) Amount of time required to converge.

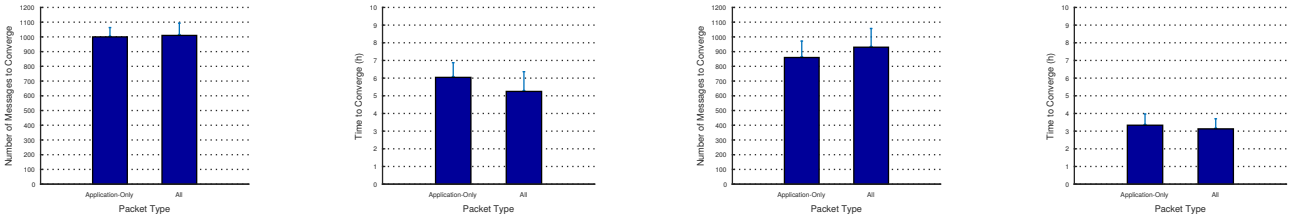
Fig. 4. PBG's edge probabilities convergence, with respect to *topology type*.

to converge, even though the messages are transmitted by nodes at regular intervals. This is because the topology type affects the average number of messages transmitted per node during a given time period (node throughput). In the sparse network, the rate of packet loss is smaller and the path lengths are greater, which results in a larger node throughput, when compared to the dense network. This means that even if the number of messages required to converge were the same for the two topology types, the time to converge would still be smaller for the sparse network.

In the results presented in Fig 4, T2A has been setup to account only for application packets (T2A's default behavior). Considering that T2A can be configured to also account for control packets (*e.g.,* routing packets) in addition to application packets, it is important to evaluate how these two configurations compare to each other. Thus, Fig. 5 shows how the edge probabilities converge with respect to packet type, *i.e., application-only* packets and *all* packets (which includes both application and control packets). While Figs. 5(a) and 5(b) present the results for the dense network scenario, Figs. 5(c) and 5(d) provide the results for the sparse topology.

With respect to the dense network scenario, in Fig. 5(a), it can be seen that the amount of counted messages required to converge is almost the same for both packet type configurations (*i.e.,* $1,000$ for application-only packets option and $1,010$ for all packets option). In turn, Fig. 5(b) shows that when T2A has the all packets option enabled, the PBG's edge labels take almost $1h$ less to converge (*i.e.,* $6.04h$ for application-only packets option and $5.25h$ for all packets option). This is expected, since when the all packet types option is enabled, it will take less time for T2A to count the number of packets required to converge (*i.e.,* $1,010$), due to the increased amount of transmitted packets per time unit.

Regarding the sparse network scenario, Fig. 5(c) shows that the number of messages T2A needs to count for the PBG's edge labels to converge is slightly greater when the all packet types option is enabled (*i.e.,* 860 messages for application-only packets option and 930 messages for all packet types option). This occurs because of the *noise* introduced by the control packets in the behavior of the communication pattern of nodes, resulting in an increased number of packets to converge. In turn, Fig. 5(d) shows that the two evaluated T2A packet type options require almost the same amount of time for the edge labels to converge (*i.e.,* $3.34h$ for application-only packets option and $3.13h$ for all packets option). This is



(a) Number of counted messages required to converge.

(b) Amount of time required to converge.

(c) Number of counted messages required to converge.

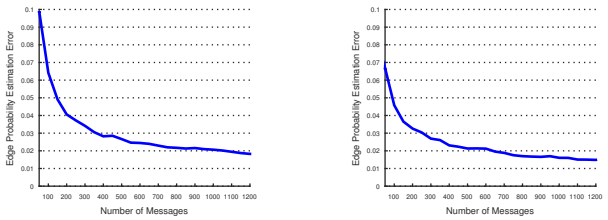
(d) Amount of time required to converge.

Fig. 5. PBG's edge probabilities convergence for *dense* network (a,b) and for *sparse* network (c,d), with respect to packet type.

expected, because when the all packet types option is enabled, more packets will be counted per time unit.

As presented in Fig. 5, in both of the evaluated topology types (dense and sparse), the time to converge is smaller when the all packet types option is enabled. Therefore, using this option seems more advantageous than the application-only packets option. However, as previously mentioned, with the use of control packets, it is possible that some kind of "noise" is introduced in the communication behavior of nodes and, consequently, in the edge probability values.

Hence, Fig. 6 shows how the accuracy of the edge probabilities is affected when control packets are considered in the T2A counting process. More specifically, this figure's graphs show, in the vertical axis, the absolute error introduced in the PBG's edge probabilities when control packets are taken into account (as compared to the edge probabilities when only application packets are considered). In the horizontal axis, the corresponding number of counted packets is presented. Figs. 6(a) and 6(b) presents the error results for the dense network and sparse network, respectively. In both of the graphs, the error decreases as the number of counted messages grows. This is because in RPL, due to the use of the Trickle algorithm, the number of control messages tends to decrease with time. Furthermore, note that the error becomes smaller than 0.02 as the number of counted messages gets sufficient to start convergence (*i.e.,* $1,010$ messages for the dense network and 930 messages for the sparse network). Therefore, the slight gain in time to converge of the all packet types option (*i.e.,* a decrease of less than one hour) comes at the cost of a 0.02 deviation in the edge probability values.



(a) Dense network.

(b) Sparse network.

Fig. 6. Edge probability estimation error introduced by *control packets*.

It should be noted that the convergence of the PBG's edge probabilities is quite dependent on the preferred parent selection procedure of RPL, which can be configured by the network administrator. Nonetheless, according to the results presented for the network configurations used, running T2A for about $6h$ in a dense network and for around $3h$ in a sparse network is sufficient to generate representative PBGs.

C. Energy Overhead of T2A

Considering that energy is one of the most limited resources available in 6LoWPANs, due to the use of batteries in the constrained nodes, the mean energy spent by T2A is analyzed, during the time period needed for the convergence of PBG labels (according to the results of the previous section). In the dense network scenarios, the simulation times are $6,04h$ and $5,25h$, for the application-only packets option and the all packet types option, respectively. Likewise, in the sparse network, the simulation times are $3,34h$ and $3,13h$.

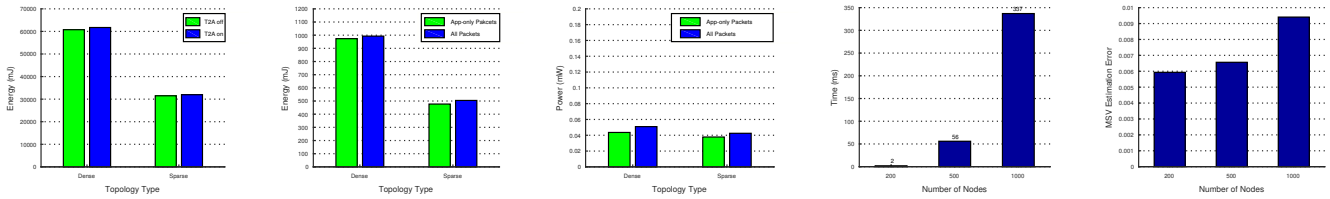
In addition to the energy consumption, the power consumption of T2A has also been analyzed. To collect the energy and power data, the Powertrace application was used. This tool is available in the Contiki OS. In this paper, the same procedure and equations described in [15] were used to compute both power and energy consumption, according to the values in the datasheet of the Tmote Sky node.

The first energy consumption evaluation is with respect to the application-only packets scenario, for both the dense and sparse topologies. Specifically, Fig. 7(a) shows the average energy consumption per node for two cases, namely, *T2A off* (in which the T2A counting process is disabled) and *T2A on* (in which the T2A counting procedure is active). As shown in the figure, in both topologies, the node energy consumed when T2A is on is only a bit greater than the energy consumed when T2A off. Thus, the energy consumption overhead generated by T2A is very small when compared to the energy consumed by Contiki to maintain the basic network operation.

Still regarding energy consumption, the application-only packets scenario is also compared to the all packets scenario. Fig. 7(b) shows the energy consumption *overhead* of those two scenarios, when compared to the case where T2A is off, to see the amount of energy exclusively spent by T2A . Note that in both topologies (dense and sparse), the all packets option consumes slightly more energy than the application-only packets option, due to the processing of additional control packets, as expected. In the dense topology, the average overhead of both options is around $1,000mJ$, while in the sparse topology this overhead falls to around $500mJ$.

From the presented results, it possible to confirm that the energy consumed by the different configurations of T2A, either

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2019.2904302, IEEE Internet of Things Journal



(a) Node Energy consumption. (b) Energy consumption overhead per node (c) Node Power consumption. (d) Time Required to Compute NSQ Metrics. (e) Error of MSV estimation.

Fig. 7. Energy and power requirements of T2A (a-c). Time requirement and Accuracy of NSQ metrics (d,e).

in a dense or sparse network, is acceptable, since, when compared to the energy spent when T2A is off, the overhead is very small. However, due to changes that can occur in the network during its operation, T2A should run periodically, in order to maintain the PBG updated in the BR. For that reason, it is essential to evaluate the power consumption of T2A, so that one can understand its energy overhead in the long run.

Hence, Fig. 7(c) shows the node power consumption for two different T2A configurations. Note that, in both cases, the power consumption is very low, *i.e.,* below $0.05\ mW$. To better understand how low are these values, take as example a mote with two $1.5\ V$ batteries of $2500\ mAh$. In this case, T2A can be active, uninterruptedly, for almost 17 years.

D. Node's Memory Consumption

The extra ROM requirements of the different T2A configurations is presented in Tab. I, regarding the total ROM used (by the whole Contiki system along with T2A) and the additional overhead added by T2A. The baseline for the default configuration is the Contiki system, and for the other configuration is the Contiki system along with T2A default. Thus, it can be seen that T2A in its default configuration requires 840 bytes of ROM, and the app-only packets configuration requires additional 30 bytes. Hence, in comparison with the Contiki system, T2A can be considered very small, keeping the total ROM usage well below the $48k$ of Tmote sky.

Finally, Tab II shows the estimated RAM usage of T2A for different configurations. The total RAM size in Tmote sky is $10k$. Then, the additional 298 bytes in either configuration can be easily executed in constrained sensor nodes.

TABLE I
ROM USAGE OF DIFFERENT T2A CONFIGURATIONS.

Configuration	Total ROM (byte)	Overhead (byte)
Default	45441	840
App-only packets	45471	30

TABLE II
RAM USAGE OF DIFFERENT T2A CONFIGURATIONS.

Configuration	Total RAM (byte)	Overhead (byte)
Default	7970	298
App-only packets	7970	0

E. Time Necessary to Calculate the Security Metrics

As soon as the PBG is built (*i.e.,* when the BR receives T2A packets from all nodes), the MSV and DL metrics are computed for all nodes. In this section, the amount of time it takes to compute the two NSQ metrics is evaluated. This experiment has been performed with three randomly generated networks. The first network has 200 nodes, the second one has 500 nodes, and the last one has a total of $1,000$ nodes. In each network, 10% of the nodes have been randomly marked as malicious. Fig. 7(d) shows the results. As can be seen, those times are negligible; all of them are smaller than $340\ ms$.

F. Error of the Estimated Metric Values

In this section, the accuracy of the MSV values calculated by the proposed model are analyzed. This accuracy is measured in terms of the error in the estimated metric values, with respect to the actual fraction of data messages that get compromised by the present attacks. The simulations are performed in the same networks presented in the previous section. For each case, $1,000$ data messages are generated by every node and routed to the BR. The simulation counts the amount of messages corrupted by the present malicious nodes. This allows to obtain, for each source node, the fraction of data messages not corrupted. This fraction is then compared to the MSV for each node, computed by the NSQ model.

Fig. 7(e) presents the average error in the MSV values computed with respect to the actual fractions of corrupted data. These error values correspond the absolute difference between the computed metric values and the actual fraction values. As depicted in the figure, the error in the MSV probabilities are quite small. In all networks, this error remains smaller than 0.01. Therefore, it is possible to conclude that the proposed model provides an accurate metric to quantify the amount of security of the sensor data colleted in the network. Since DL is derived from MSV, it can also be regarded as accurate.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, the Node Security Quantification (NSQ) model is presented. NSQ consists of a security metric model that combines IDS alerts with the network communication pattern to evaluate the security level of constrained nodes and the sensor data consumed by users. The security awareness provided by the NSQ metrics support users' and network administrators' decision-making. The obtained simulation results demonstrate that NSQ accurately quantifies security level with

very low energy and performance overhead. As future work, new methods to update the PBG could be developed and evaluated. The T2A component could also be extended with new functionalities to save more energy. Despite the fact that Cooja is a quite realistic simulator, an additional future work is the assessment of the NSQ model in a real network.

ACKNOWLEDGMENTS

This study was supported by the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES)* – Finance Code 001, by National Funding from the FCT – *Fundação para a Ciência e a Tecnologia* through the UID/EEA/50008/2019 Project, by RNP, with resources from MCTIC, Grant No. 01250.075413/2018-04, under the *Centro de Referência em Radiocomunicações* – CRR project of the *Instituto Nacional de Telecomunicações* (Inatel), Brazil; and by Brazilian National Council for Research and Development (CNPq) via Grant No. 309335/2017-5.

REFERENCES

[1] J. Hui and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks," *RFC 6282*, 2011.
[2] J. Lin, W. Yu *et al.*, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.
[3] N. Kushalnagar, G. Montenegro, and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals," *RFC 4919*, 2007.
[4] C. Bormann, M. Ersue, and A. Keranen, "Terminology for Constrained-Node Networks," *RFC 7228*, 2014.
[5] I. Butun, S. D. Morgera, and R. Sankar, "A Survey of Intrusion Detection Systems in Wireless Sensor Networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.
[6] B. B. Zarpelao, R. S. Miani *et al.*, "A survey of intrusion detection in internet of things," *J NETW COMPUT APPL*, vol. 84, pp. 25 – 37, 2017.
[7] A. Ramos, M. Lazar *et al.*, "A Security Metric for the Evaluation of Collaborative Intrusion Detection Systems in Wireless Sensor Networks," in *IEEE ICC*, 2017.
[8] M. Anand, Z. Ives, and I. Lee, "Quantifying eavesdropping vulnerability in sensor networks," in *ACM DMSN Wksp*, 2005, pp. 3–9.
[9] A. Ramos and R. Holanda Filho, "Sensor Data Security Level Estimation Scheme for Wireless Sensor Networks," *Sensors*, vol. 15, p. 33, 2015.
[10] A. Ramos, B. Aquino *et al.*, "Quantifying Node Security in Wireless Sensor Networks under Worm Attacks," in *SBRC*, 2017.
[11] ——, "A quantitative model for dynamic security analysis of wireless sensor networks," in *GLOBECOM*, 2017, pp. 1–6.
[12] A. Ramos, M. Lazar *et al.*, "Model-based quantitative network security metrics: A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2704–2734, Fourthquarter 2017.
[13] IEEE802.15.4, "Ieee standard for low-rate wireless networks," *IEEE Std 802.15.4-2015 (Rev. of IEEE Std 802.15.4-2011)*, pp. 1–709, April 2016.
[14] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the rpl-based internet of things," *International Journal of Distributed Sensor Networks*, vol. 9, no. 8, p. 794326, 2013.
[15] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, nov 2013.
[16] S. Raza, D. Trabalza, and T. Voigt, "6lowpan compressed dtls for coap," in *DCOSS*, 2012, pp. 287–289.
[17] S. Raza, S. Duquennoy *et al.*, "Securing communication in 6lowpan with compressed ipsec," in *DCOSS*, 2011, pp. 1–8.
[18] M. A. Lodhi, A. Rehman *et al.*, "Multiple path rpl for low power lossy networks," in *APWiMob*, 2015, pp. 279–284.
[19] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *IEEE LCN*, 2004, pp. 455–462.
[20] F. Osterlind, A. Dunkels *et al.*, "Cross-level sensor network simulation with cooja," in *IEEE LCN*, 2006, pp. 641–648.
[21] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *IPSN*, 2005, pp. 364–369.