Centralized and Distributed Intrusion Detection for Resource-Constrained Wireless SDN Networks

Gustavo A. Nunez Segura, *Student Member, IEEE,* Arsenia Chorti, *Senior Member, IEEE,* and Cintia Borges Margi, *Senior Member, IEEE*

***Abstract*—Software-defined networking (SDN) was devised to simplify network management and automate infrastructure sharing in wired networks. These benefits motivated the application of SDN in resource-constrained wireless networks to leverage solutions for complex applications. However, some of the core SDN traits expose the networks to denial of service attacks (DoS). There are proposals in the literature to detect DoS in wireless SDN networks; however, not without shortcomings: there is little focus on resource constraints, high detection rates have been reported mostly for small networks and the detection is disengaged from the identification of the type of attack or the attacker. Our work targets these shortcomings by introducing a lightweight, online change point detector to monitor performance metrics that are impacted when the network is under attack. A key novelty is that the proposed detector is able to operate in either centralized or distributed mode. The centralized detector has very high detection rates and can further distinguish the type of attack from a list of known attacks. In turn, the distributed detector can be useful to identify the nodes launching the attack. Our proposal is tested over IEEE 802.15.4 networks. The results show detection rates exceeding 96% in networks of 36 and 100 nodes and identification of the type of attack with a probability exceeding 89% when using the centralized approach.**

***Index Terms*—Internet of Things, wireless sensor networks, software-defined networking, intrusion detection, change point detection.**

I. Introduction

SOFTWARE-DEFINED networking (SDN) is a logically centralized paradigm, devised to simplify network management and automate infrastructure sharing in wired networks [1][2]. These benefits motivated the application of SDN in resource-constrained settings, such as wireless sensor networks (WSN) and Internet of Things (IoT) to leverage solutions for complex applications. Unlike other commonly used protocols that are decentralized (such as RPL - RFC 6550), SDN-based protocols are fundamentally different. The fusion of SDN – WSN and SDN – IoT are referred to as software-defined wireless sensor networks (SDWSN) and software-defined Internet of Things (SDIoT), respectively [3] [4].

Network control centralization and data and control planes separation are fundamental enablers of SDN programmability and network reconfiguration. However, these traits turn the network prone to denial of service (DoS) attacks, a vulnerability that is inadvertently passed on to SDWSNs and SDIoT [5] [6].

Gustavo A. Nunez Segura and Cintia Borges Margi are with Departamento de Engenharia de Computação e Sistemas Digitais, Universidade de São Paulo, São Paulo 05508-010, Brazil.

Arsenia Chorti is with ETIS UMR8051, CYU cergy Paris Université, ENSEA, CNRS, F-95000, Cergy, France.

There are proposals in the literature to detect and mitigate DoS attacks in SDNs, including for SDWSNs and SDIoT. However, the proposed approaches are not adapted to very restricted networks, such as out-of-band connection for control packets between switches and controllers. Additionally, most works reported high detection rates only for small networks. Other shortcoming in existing literature is a lack of solutions able to identify the type of attack and the attacker itself.

With these challenges in mind, we propose a novel DoS detector for constrained SDN networks based on change point (CP) detection theory. Our main hypothesis is that detecting a change in the monitored network metrics can be used as an alert for an anomaly, i.e., for intrusion detection purposes. A key novelty is that the proposed detector is able to operate in either centralized or distributed mode; while there exist works on decentralized detection for networks using distributed protocols, such as RPL [7], decentralized detection is largely unexplored in SDN-based networks.

In the centralized detection, a Security application monitors changes in the control packets overhead and the data packets delivery rate underlying statistics. In the distributed detection, every node is in charge of detecting a change on its own local metrics and to inform the Security application in case of a change. Notably, the centralized detector that runs on the controller permits to *identify with a very high rate the attack* and can further *distinguish the type of attack* from a list of known attacks. The distributed detector that runs on individual nodes is also able to *detect the DoS attacks with a high rate* and further provides information that permits to *identify the nodes launching the attack*.

We evaluated the performance of both approaches on the IT-SDN framework [19], simulating new-flow and neighbor information types of attacks in topologies of 36 and 100 nodes, when all the sensor nodes were emulated as TelosB mote. Our contributions are:

1) We developed DoS detectors suitable for restricted networks (IEEE 802.15.4);
2) The proposed detectors do not need large training datasets (unlike machine learning (ML)-based detectors); as will be discussed in the body of the paper, a short window (e.g., 200 samples) of normal operation suffices to capture the baseline underlying statistics;
3) We studied the parameterization of the centralized detector to optimize the detection speed versus the detection rate and studied the trade-off between them. The quickest detector achieved an attack identification rate of more than 89%, increased to 99% for less agile detectors.

TABLE I
RELATED WORK

Author	High detection rate	Multiple types of attack	Attack type identification	Resource constrained networks	Attacker identification
Bhunia and Gurusamy [8]	✓				
Jia *et al.* [9]	✓		✓		
Ravi and Shalinie [10]	✓				✓
Wani *et al.* [11]	✓	✓	✓		
Li *et al.* [12]	✓	✓			
Bagga *et al.* [13]	✓	✓			
Yin *et al.* [14]				✓	✓
Miranda *et al.* [15]		✓		✓	
Wang *et al.* [16]		✓	✓	✓	✓
Nunez *et al.* [17]	✓	✓		✓	
Nunez *et al.* [18]	✓	✓	✓	✓	
Proposed centralized / decentralized detector	✓	✓	✓	✓	✓

4) The decentralized detector is so lightweight that it can run on each individual node, which allowed us to identify the region in which the attack is launched, or even, the attacker itself with a probability exceeding 93%.

The remainder of the paper is organized as follows. In Section II the state of the art is summarized while in Section III the investigated SDWSN DoS attacks are explained. Section IV outlines the CP detector. In Sections V and VI we present the centralized and distributed detectors, respectively. Finally, Section VII concludes the paper.

II. RELATED WORK

In this section, we review security DoS approaches for resource-constrained SDN-based networks, focusing on detection and identification accuracy, type of DoS attacks detected and the consideration of resource constraints. Table I summarizes the main performance metrics of related works and of the proposed CP detector. A comparative analysis is performed for the following five metrics: i) the ability to achieve high detection rates, i.e., equal or greater than 90%; ii) multiple types of attack detection; iii) type of attack identification; iv) limitation of resources; and v) attacker identification.

Several recent works about DoS attack detection in SDN-based networks used centralized ML techniques to detect anomalies in the behavior of the network [8], [10], [9], [11], [20], [12], [13]. In these works, high detection rates were demonstrated, i.e., higher than 90%. However, none of the proposed approaches considered resource constraints or were evaluated in restricted networks, they were OpenFlow-based or required high traffic of packets monitoring the network. Regarding other metrics, in [9] and [11], an attack type identification algorithm was proposed, while in [10] an attacker identification mechanism was presented.

The proposals in [14], [15], [16] considered resource-constrained networks but did not attain high detection rates. Concerning other metrics, in [15], [16], multiple types of attack were detected while in [14], [16] attacker identification algorithms were presented.

The main shortcoming in the state of the art is the trade-off between detection rate and resources to execute the DoS attack detection. The proposals that attained high detection rate were not suited for resource-constrained networks and proposals that considered resource limitations did not attain high detection rates. As shown in Table I, our solution is able to provide high detection rates while it is well suited for resource-constrained networks. Additionally, our solution was able to detect different types of DoS attack, identify the type of attack with high probability and identify the area in which the attacker is located, or even the attacker itself, bridging the gap in the literature.

The present study builds upon our previous works in [21], [17] and [18], in which we analyzed the impact of different types of attacks on various performance metrics, proposed a universal CP DoS detector that combined an offline and an online detector as well as an entirely online multimetric CP detector. We here extend our results on the centralized approach studying the trade-off between detection rate and agility of detection and further propose a distributed approach based on metrics collected and analyzed on every node. Using the transmitting time, processing time, etc., in the decentralized DoS detection, the possibility of intrusion detection at PHY arises, along with its potential integration with physical layer security solutions [22]. Note that the centralized approach requires more bandwidth while the distributed consumes more of the nodes memory resources.

III. SDWSN DOS ATTACKS

In terms of security, SDNs have advantages and disadvantages. The access to network traffic and performance data along with the controller global view has been leveraged to develop new security strategies [23]. Based on centralized traffic analysis and security policies, the controller has an important role in determining if the network is under attack and in reconfiguring the network to mitigate the impact. In turn, SDNs are entirely dependent on the controller, if it is compromised, the whole network is compromised [24], [25].

IT-SDN [19] is an SDWSN framework comprising the sensing layer, the control layer and three communication protocols: the Southbound protocol, the Neighbor Discovery protocol and the Controller Discovery protocol. The sensing layer is composed of the wireless sensor devices used to collect data from the environment and relay data to the sinks. The control plane is in charge of making routing decisions and configuring them in the sensing layer devices. The Southbound protocol

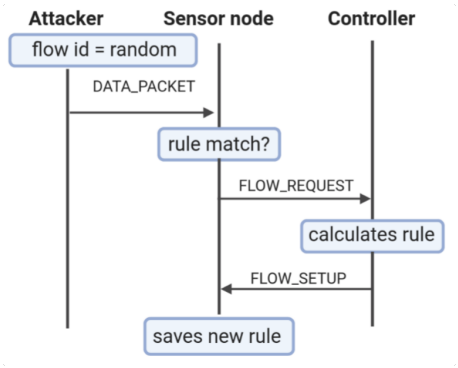


Fig. 1. FDFF attack: the attackers send data packets to their neighbors using unknown IDs. The sensor nodes request a rule to the controller to treat this packet, the controller calculates the rule and send it to the sensor node.

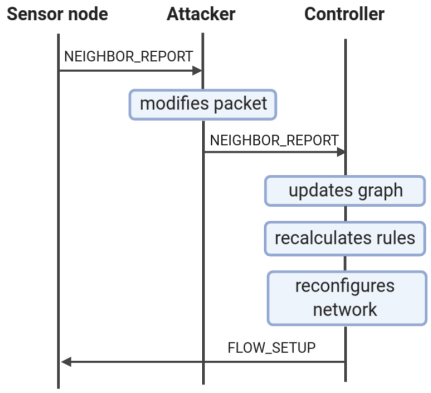


Fig. 2. FNI attack: the sensor node sends a neighbor report to the controller and the attacker in the route (in the case there is one) modifies the neighborhood information before forwarding the packet to the controller.

defines the message formats for communication between the WSN and the controller. All the nodes in the network use the Controller Discovery protocol to find a route to the controller and use the Neighbor Discovery protocol to collect neighborhood information to then send it to the controller.

Six packet types are provisioned in the Southbound protocol: flow request, flow setup, flow ID register, acknowledgement, neighbor report and data packet. To obtain information for an incoming packet, a WSN node utilizes the flow request to query the controller. The controller replies with a flow setup packet. Moreover, the controller could change already configured entries using such packet. The neighbor report packet contains the sender's neighborhood information, which is used by the controller to update the network graph. The nodes send a neighbor report to the controller if: (i) the node detects one or more new neighbors, (ii) the node detects one or more nodes are no longer its neighbors, or, (iii) there is a significant change in one or more neighbors' routing metric.

We tested our proposal when the network was under two different attacks: a new-flow-based attack [26] and a neighbor information type of attack. Based on the IT-SDN characteristics, we adapted these two attacks to target its security vulnerabilities, dubbed in the rest of this paper as a false data flow forwarding (FDFF) and a false neighbor information (FNI).

1) A FDFF attack targets the controller via network devices. First, the attacker sends data packets with unknown flow IDs to its neighbors. The neighbors receive the packet and check the flow table to determine the action required, without success, thus ask a rule to the controller by sending a flow rule request packet. The controller receives this packet, calculates the rule and replies sending a flow setup packet. Fig. 1 shows the packets exchange during this attack, which aims at increasing the packet traffic and the controller and neighbors processing overhead [21]. Since the attacker uses its neighbors to attack the control plane, locating an attacker located outside of the controller radio range is challenging. In our proposal, we overcome this issue by proposing a distributed detection approach.
2) A FNI attack modifies the packets that contain neighbor information. The attackers do not intercept the neighbor information packets but modify the ones that use them to reach the controller. When receiving a neighbor information packet, the attacker modifies either the routing metric or the node identification number, then the packet continues its normal route to reach the controller. The packets exchange for this attack is depicted in Fig. 2. This attack leads the controller to mistreat false information as true and to send erroneous routing rules to the nodes. This attack significantly disturbs the data and control packets delivery rate [21].

In the case of SDWSN and SDIoT, these type of attacks can be critical to resource-constrained devices. In Table II, we summarize some IEEE 802.15.4 compliant platforms along with Raspberry Pi 3 specifications to highlight this point.

TABLE II
WSN MOTES SPECIFICATIONS

Platform	Microprocessor model	Clock (MHz)	Flash Mem. (kB)	RAM (kB)
TelosB	MSP430	8	48	10
SensorTag	ARM Cortex-M3	48	128	20
RE-Mote	ARM Cortex-M3	32	512	32
Raspberry Pi3	4 x ARM Cortex-A53	1200	SD card	1×10^6

IV. ONLINE CHANGE POINT DETECTION ALGORITHM

In this section, we provide an outline of the online CP algorithm used for DoS attack detection in SDWSN. Generally, CP problems are formulated as hypothesis tests. The null hypothesis represents the structural stability of the process, while the alternative hypothesis indicates one or multiple CPs and is used to detect an anomaly. The test statistics may be viewed as two-sample tests adjusted for the unknown break location, thus leading to max-type procedures. Often asymptotic relationships are derived to obtain critical values for the tests. After the null hypothesis is rejected, the location(s) of the break(s) need(s) to be estimated [27].

To outline the online CP algorithm, let $\{X_n : n \in \mathbb{N}\}$ be the time series of the metric monitored. Using Wold's theorem

we can assume that, for $X_1, ..., X_N$, each sample is expressed as $X_n = \mu_n + Y_n$, where $\{\mu_n, n \in \mathbb{N}\}$ is the mean of the time series and $\{Y_n : n \in \mathbb{N}\}$ is a random zero mean term, so that we can rewrite X_n as:

$$X_n = \begin{cases} \mu + Y_n, & n = 1, \ldots, m + k^* - 1 \\ \mu + Y_n + I, & n = m + k^*, \ldots \end{cases} \quad (1)$$

where $k^* \in \mathbb{N}^*$ represents the unknown time of change and μ, $I \in \mathbb{R}^r$ represent the mean parameters before and after k^*, respectively. We here assume a period of no change in the mean of at least m samples, i.e., during the first m samples of our observation there is no change so that $\mu_1 = \ldots = \mu_m$.

During this period, our detector "learns" in real-time the statistics of the observed time series and the mean value in particular. The statistical hypothesis test is articulated as,

$$\begin{aligned} H_0 &: I = 0 \\ H_1 &: I \neq 0. \end{aligned} \quad (2)$$

The on-line sequential analysis belongs to the category of stopping time stochastic processes. In general, a chosen on-line test statistic $TS_{on}(m, l)$ and a given threshold $F(m, l)$ define the stopping time $\tau(m)$:

$$\tau(m) = \begin{cases} \min\{l \in \mathbb{N} : TS_{on}(m, l) \geqslant F(m, l)\}, \\ \infty, \text{ if } TS_{on}(m, l) < F(m, l) \ \forall l \in \mathbb{N}, \end{cases} \quad (3)$$

implying that $TS_{on}(m, l)$ is calculated on-line for every l in the monitoring period. The procedure stops if the test statistic exceeds the value of the threshold function $F(m, l)$. As soon as this happens, the null hypothesis is rejected and a CP is detected. $F(m, l)$ is defined as,

$$F(m, l) = cv_{on,\alpha} g(m, l), \quad (4)$$

where: (i) $cv_{on,a}$ is the critical value determined from the asymptotic behavior of the stopping time procedure under H_0 by letting $m \to \infty$, (ii) and $g(m, l)$ is the weight function defined as:

$$g(m, l) = \sqrt{m} \left(1 + \frac{l}{m}\right) \left(\frac{l}{l + m}\right)^{\gamma} \quad (5)$$

where the sensitivity parameter $\gamma \in [0, 1/2)$.

The online algorithm uses the standard CUSUM detector [28], with test statistic denoted by TS^{ct}_{on}. Its corresponding critical value is denoted by $cv^{ct}_{on,\alpha}$ and the stopping rule by $\tau_{ct}(m)$. The sequential CUSUM detector is denoted by $E(m, l)$,

$$E(m, l) = \left(\overline{X}_{m+1,m+l} - \overline{X}_{1,m}\right). \quad (6)$$

The standard CUSUM test statistic is expressed as:

$$TS^{ct}_{on}(m, l) = l \widehat{\Omega}_m^{-\frac{1}{2}} E(m, l), \quad (7)$$

where $\widehat{\Omega}_m$ is the estimated long-run covariance, defined as in (4), which captures the dependence between observations. Then, the stopping rule $\tau_{ct}(m)$, is defined as:

$$\tau_{ct}(m) = \min\{l \in \mathbb{N} : \|TS^{ct}_{on}(m, l)\|_1 \geq cv^{ct}_{on,\alpha} g(m, l)\}, \quad (8)$$

where the ℓ_1 norm is involved to modify TS^{ct}_{on} so that it can be compared to a one-dimensional threshold function. The critical value, $cv^{ct}_{on,\alpha}$, is derived from the asymptotic behavior of the stopping rule under H_0:

$$\lim_{m \to \infty} Pr\{\tau(m) < \infty\} \quad (9)$$

$$= \lim_{m \to \infty} Pr\left\{ \sup_{1 \leqslant l \leqslant \infty} \frac{\|TS^{ct}_{on}(m, l)\|_1}{g(m, l)} > cv^{ct}_{on,\alpha} \right\}$$

$$= Pr\left\{ \sup_{t \in [0,1]} \frac{\|W(t)\|_1}{t^{\gamma}} > cv^{ct}_{on,\alpha} \right\} = \alpha \quad (10)$$

where $W(t)$ denotes the Brownian motion with mean 0 and variance t. The on-line critical values can be computed using Monte Carlo simulations, considering that,

$$cv^{ct}_{on,\alpha} = \sup_{t \in [0,1]} \frac{W(t)}{t^{\gamma}}. \quad (11)$$

Lastly, the estimated on-line CP, $\hat{k}^*_{on}$, is derived directly from the value of the stopping time $\tau(m)$, as,

$$\hat{k}^*_{on} = m + \{\tau(m) | \tau(m) < \infty\}. \quad (12)$$

Summarizing, the overall algorithm has 3 main steps:

- Step 1: define the values of the quantities m, γ, the confidence level α and set l.
- Step 2: after collecting m samples of the metric, $\Gamma(m, l)$ (7) and the weight function in (5) are calculated for every l in the monitoring period to then apply (9).
- Step 3: If a CP is detected, the online process stops. Conversely, if period l ends, a new monitoring period is defined.

We note that the computational complexity of the algorithm is $O(N \log N)$, where N is the length of the monitoring window.

V. Centralized Detection

[21], showed that FDFF and FNI attacks have a significant impact on the data packets delivery rate and the control packets overhead. The control packets overhead include all southbound packets, except the data ones, and the neighbor discovery protocol packets. A centralized intrusion detection, first proposed in [17] and [18], can be used to determine if the network is under attack based on the monitoring of these two metrics.

Our proposal is based on the SDN architecture explained in the IRTF RFC 7426 [29], depicted in Fig. 3. The Security application is in charge of all security decisions. The management plane main purpose is to ensure the network is running optimally; here, we additionally leverage it to collect the metrics used to detect the attacks. Furthermore, the control plane, besides making the control decisions, provides topology information to the Security application.

The Security application runs the CP detection algorithm (Section IV) monitoring the data packets delivery rate and the control packets overhead time series. To construct the time series, the application requests this information from the management plane periodically. The management plane

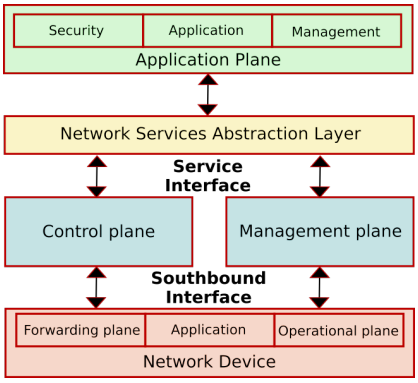


Fig. 3. SDWSN architecture based on IRTF RFC 7426 document [29].

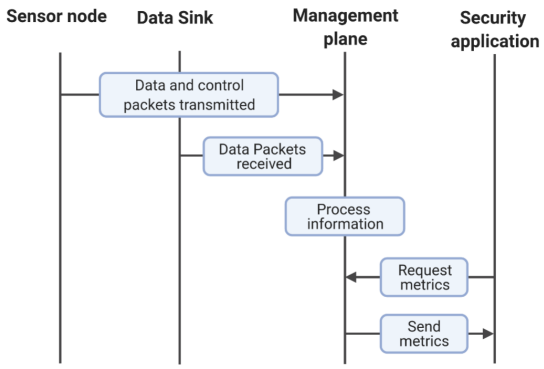


Fig. 4. Centralized detection message exchanges for time series constructions: the data sink and the sensor nodes send to the management plane the information required to calculate the data packets delivery rate and the control overhead of the whole network. Then the Security application requests the metric value from the management plane

establishes communication with the network devices using the Southbound Interface to obtain information about the network operation. The sensor nodes report to the management plane the number of data and control packets transmitted, and the data sink node reports the number of data packets received. With this information, it is possible to calculate both metrics. A sink node exclusively used for the reception of management packets is used. The sensor nodes and the data sink node can be programmed to either send this information to the management plane periodically or upon request. Fig. 4 shows the message exchange required to construct the metrics time series.

We here propose to run two detectors in parallel on the Security application to identify the type of attack based on which detector triggers an alert first. This is motivated by previous results which showed a relation between the type of attack and the metrics analyzed [21]. We also expect that using the proposal with two detectors in parallel allows detecting when the network is under attack irrespective of the type of attack with high detection rates. Additionally, the attack detection and identification information could be sent to the controller to implement mitigation strategies, yet this is outside the scope of this work. In detail, an attack is classified as a FDFF or a FNI attack based on the following reasoning, corroborating our previous results in [18]:

1) If a CP is detected in the mean value of the data packets delivery rate or the mean value of the control packets overhead, we determine that the network is under attack;
2) If the CP is first detected in the mean value of the control packets overhead, the attack is classified as FDFF; conversely, if the CP is detected first in the mean value of the data packet delivery rate, the attack is classified as FNI.

A. Experimental setup

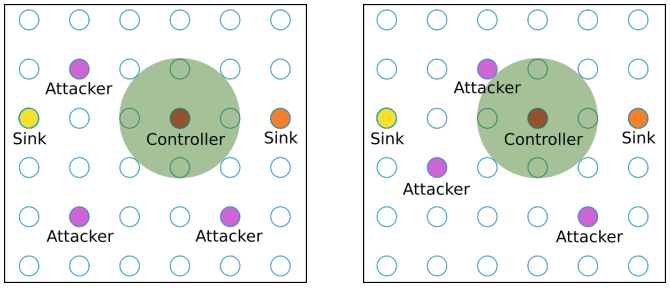
We generated a dataset comprising 480 simulations, divided into 240 simulations of FNI attacks and 240 simulations of FDFF attacks. Then, we split each subgroup into two sets: one set for parameterization to capture different trade-offs between the detection rate and the speed of detection and the other for validation. In particular, we used the first set to determine the optimal values of $\{m, \gamma\}$ (both parameters explained in Section IV) for each type of attack and each observed metric. Then, using the values determined for $\{m, \gamma\}$, we executed the CP detector algorithm over the validation sets to evaluate the performance achieved. We performed simulations on square grids with either 36 or 100 nodes and we varied the number of intruders (attackers) in three proportions: 5%, 10% and 20% of the total of nodes in the network.

First, we executed the algorithm on the first set for $m \in \{100, 150, 200\}$ and $\gamma \in \{0, 0.15, 0.25, 0.35, 0.45, 0.49\}$ to determine the values that provide the best performance for different trade-offs between the detection rate DR and the detection time median DTM. The DR is the ratio of successfully detected attacks over the total number of attacks. The DTM is the median of the number of samples required to detect the attack. From that, we introduced a "detection score" to capture the relative importance given to the DR versus the DTM (which focuses on detecting changes in a signal or a time series as quickly as possible after they occur [30]). The proposed detection score, denoted by P_{DS}, is defined as:

$$P_{DS}(A, B) = A(1 - S) + B(DR), \text{ with } A + B = 1, \quad (13)$$

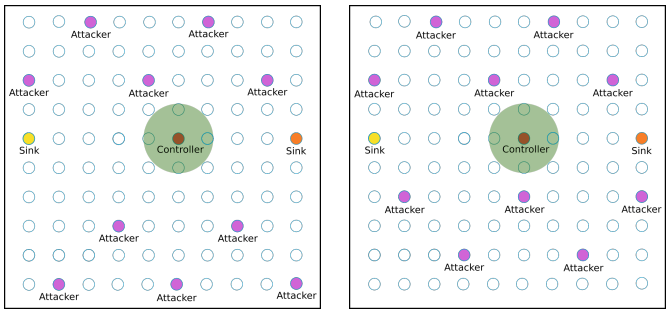
where A and B are coefficients that determine the relative weight of each term, and $S = \frac{DTM}{l}$ with l the number of samples monitored after the attack starts. We used five combinations of A and B, i.e., $(A, B) \in \{(1, 0), (0.8, 0.2), (0.5, 0.5), (0.2, 0.8), (0, 1)\}$, to compare the results when prioritizing the speed of detection ($A > B$) versus when prioritizing the detection rate ($A < B$).

During the evaluation, two CP detectors ran in parallel. One detector for monitoring the control packets overhead and the other for monitoring the data packets delivery rate. The validation set comprised both FDFF and FNI attack simulations, 50% of each, including all the topologies chosen and attack intensity levels. In the validation stage, we used the optimal pairs (m, γ) identified for each pair (A, B) to maximize the metric $P_{DS}(A, B)$. Whenever a CP was detected, we stopped the detectors, declared the network under attack and identified which metric triggered the detector to determine the type of attack.



(a) FDFF attack (b) FNI attack

Fig. 5. Topology example for 36 nodes with 10% of nodes behaving as attackers: there is one SDN controller, two sinks and three attackers. The green circle represents the radio range of all nodes. No attackers are in the radio range of any sink or controller nodes



(a) FDFF attack (b) FNI attack

Fig. 6. Topology example for 100 nodes with 10% of nodes behaving as attackers: there is one SDN controller, two sinks and ten attackers. The green circle represents the radio range of all nodes. No attackers are in the radio range of any sink or controller nodes

The SDWSN implementation uses IT-SDN [1], without changing the default configuration [19], and the simulations were performed using COOJA simulator [31], emulating TelosB motes. We used fully bidirectional square grid topologies with 36 and 100 nodes, one controller, two sinks: one sink to receive data packets and the other to receive management packets. The controller was placed in the center of the grid and the sinks were placed in the middle of the grid edge, since this location gave a better performance in terms of delay, control overhead, energy consumption and delivery rate according to [19]. The attackers were semi-randomly distributed in the network under the condition that two or more attackers cannot be neighbors and this distribution remains equal in every scenario replication. Figs. 5 and 6 show the attackers distribution for 36 nodes and 100 nodes, respectively, when 10% of nodes are attackers. The green circle around the controller represents the devices radio range. Notice that no attackers are in the radio range of any sink or controller nodes.

The sensor nodes were programmed to transmit one data packet every 30 seconds and one management packet every 2 minutes, both with a 10-byte payload. The data packets contained the application information and the management packets contained the information required by the network management plane [32]. The data packets delivery rate and the control packets overhead were observed every two minutes,

[1]Available at http://www.larc.usp.br/users/cbmargi/www/it-sdn/

TABLE III
SIMULATION PARAMETERS

Simulation parameters	
Node boot interval	$[0, 1]$ s
Data traffic start time	$[2, 3]$ min
Radio module power	0 dB
Distance between neighbors	50 m
Attacks begins after	28800 s

IT-SDN parameters	
Controller retransmission timeout	60 s
ND protocol	Collect (Contiki-3.0)
Link metric	ETX (0 - 255)
Neighbor report max frequency	1 packer per minute
CD protocol	none
Flow setup	source routed
Route calculation algorithm	Dijkstra
Route recalculation threshold	10%
Flow setup types	regular or source routed
Flow table size	10 entries

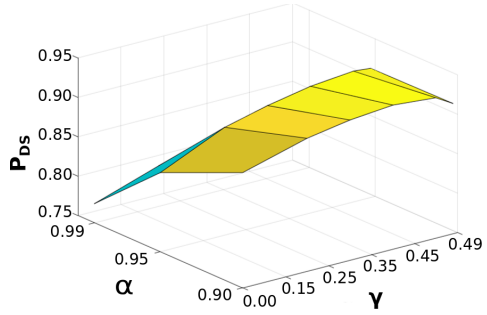
considering the exchange of messages in the whole network during this window of time. The delivery rate was calculated by dividing the number of data packets successfully received by the number of data packets sent. The control packets overhead was quantified as the number of control packets sent. Since samples were collected every two minutes, each simulation was run for 10 hours. During the first eight hours the network operated normally (i.e., for 240 samples there was no change), then the attack was triggered. This imposed a bound $m < 240$. Table III summarizes the remainder simulation and IT-SDN most important parameters.

B. Results and discussion

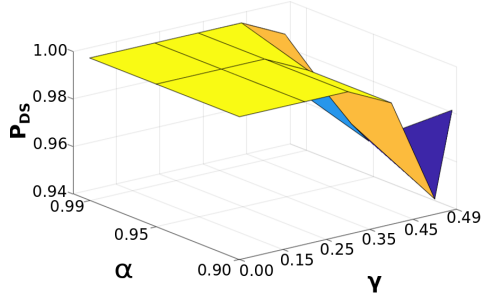
In Section V-B1, the results of the training experiments are analyzed to determine the values of m and γ that maximize P_{DS}. In Section V-B2 we discuss performance.

1) Optimizing m and γ: P_{DS} was calculated for all the topologies, attack scenarios and combinations of m and γ. For $\alpha \in \{0.90, 0.95, 0.99\}$, in 90% of all the cases P_{DS} was maximized when $m = 200$, turning this value a universally optimal choice and the m value used for the remainder of the analysis. Consequently, when running the online detector, no training is required, other than the observation of 200 samples of normal network operation.

For the next part, the results were separated grouping each attack based on the monitoring metric: for a FDFF attack, the control overhead CP detection results were analyzed, while for a FNI attack the data packets delivery rate, CP detection results were analyzed based on the analysis in [17]. The average value of P_{DS} as a function of γ and α for the case of a FDFF attack is depicted in Fig. 7. Fig. 7a shows that when prioritizing faster detection (i.e. $A = 1$) the higher values of P_{DS} are obtained for $\gamma = \{0.35, 0.45\}$ and the lowest for $\gamma = \{0, 0.15\}$. In turn, Fig. 7b shows that prioritizing the detection rate, the higher

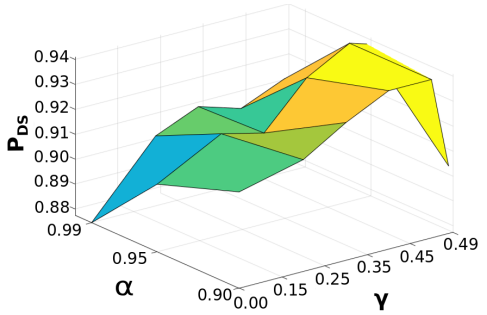


(a) $A = 1$ and $B = 0$

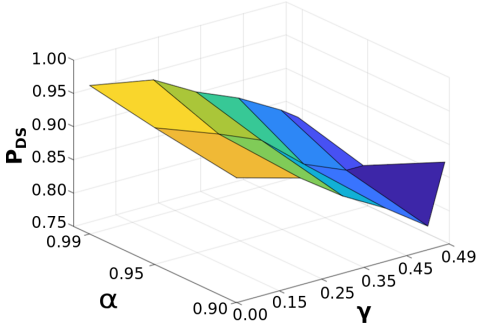


(b) $A = 0$ and $B = 1$

Fig. 7. Metric P_{DS} in function of γ and α for FDFF attack: (a) shows P_{DS} when prioritizing the quickest detection and (b) shows P_{DS} when prioritizing the detection rate.



(a) $A = 1$ and $B = 0$



(b) $A = 0$ and $B = 1$

Fig. 8. Metric P_{DS} in function of γ and α for FNI attack: (a) shows P_{DS} when prioritizing the quickest detection and (b) shows P_{DS} when prioritizing the detection rate

TABLE IV
γ THAT MAXIMIZES P_{DS}

P_{DS}	γ		
	$\alpha = 0.90$	$\alpha = 0.95$	$\alpha = 0.99$
Best γ for control overhead CP detector			
$A = 1$ and $B = 0$	0.45	0.45	0.45
$A = 0.8$ and $B = 0.2$	0.35	0.35	0.45
$A = 0.5$ and $B = 0.5$	0.25	0.35	0.45
$A = 0.2$ and $B = 0.8$	0.25	0.25	0.35
$A = 0$ and $B = 1$	0	0	0
Best γ for delivery rate CP detector			
$A = 1$ and $B = 0$	0.45	0.45	0.45
$A = 0.8$ and $B = 0.2$	0	0.15	0.15
$A = 0.5$ and $B = 0.5$	0	0	0.15
$A = 0.2$ and $B = 0.8$	0	0	0
$A = 0$ and $B = 1$	0	0	0

values of P_{DS} were obtained for $\gamma = \{0, 0.15, 0.25\}$, reaching $P_{DS} = 1$.

The average value of P_{DS} for the case of a FNI attack is presented in Fig. 8. Contrary to the results in Fig. 7, in this case the trend was not as clear because lower values of γ maximized P_{DS} when $A = B = 0.5$ and $B = 1$, i.e., the detection rate influences P_{DS} more than the detection speed.

By varying γ, we were able to configure the detector to prioritize faster detection or accuracy. However, the response was different for both attacks. In Table IV, the values of γ that maximize P_{DS} are shown. In cases whereby more than one value resulted in the same or very similar values, one was chosen arbitrarily.

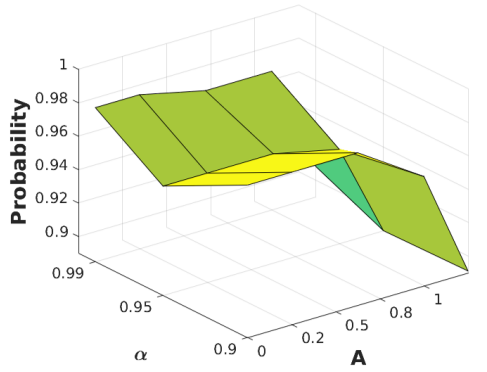
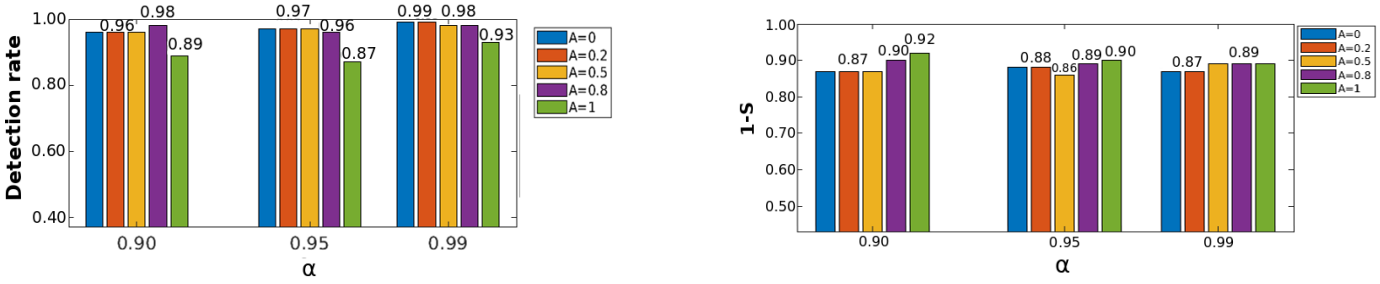


Fig. 9. Probability of control overhead CP detector being triggered first in case of FDFF attack

2) Centralized detector performance: For this part, two detectors were set to run simultaneously using $m = 200$. The first experiment was devised to identify the type of attack based on the first detector triggered. The probability of the control overhead CP detector being triggered first in case of a FDFF attack is depicted in Fig. 9. These results showed that in the worst case the detector monitoring the control overhead has a probability between 0.89 and 0.98 of being triggered first in case of a FDFF attack. In case of a FNI attack, the detector monitoring the data packets delivery rate was triggered first in 100% of the events. These results showed that there is evidence to support the hypothesis drawn in our previous works about the relation metric/attack. Next, the detection performance irrespective of the type of attack was analyzed, when both detectors were running simultaneously. The results in Fig. 10 showed a detection rate over α when



(a) Detection rate

(b) Detection speed

Fig. 10. Detection performance of FDFF and FNI attacks using γ and m values that optimize P_{DS} for five different cases: $\{A, B\} = \{\{1, 0\}, \{0.8, 0.2\}, \{0.5, 0.5\}, \{0.2, 0.8\}, \{0, 1\}\}$

$A = \{0, 0.2, 0.5, 0.8\}$ for $\alpha = \{0.90, 0.95\}$; when $\alpha = 0.99$, $DR = \alpha$ for $A = \{0, 0.2\}$ only. As a conclusion, to maximize the detection rate the configuration for $A = \{0, 0.2\}$ should be used. In terms of detection speed, as shown in Fig. 10b, to maximize the detection rate means a lag of 3 samples on average with respect to the fastest detection result obtained.

Summarizing Section V, we chose the pairs (m, γ) that maximized the detection performance metric P_{DS}. Our results showed that in 90% of cases $m = 200$ maximized P_{DS}. With respect to γ, it was observed that using $\gamma = 0.45, 0.49$ reduced the time to detect the attack but this had an adverse effect on the detection rate. Conversely, when $\gamma = 0, 0.15$ the detection rate was maximized at the cost of delaying the detection. Then, the CP detectors using the parameters values chosen before were tested. The results showed that it is possible to detect the attack with $DR \geq \alpha$ when $B > A$. Yet prioritizing the fastest detection, the detection rate drops to 0.93 or below. In terms of detection rate, this proposal and our previous proposal [17] have a similar performance. Moreover, here we provide concrete evidence to support the relation between monitored metric and the type of attack.

VI. Distributed Detection

In this section we explain our distributed detection proposal for DoS attacks in SDWSN, implementing one CP detector per node (potentially on every node). To the best of our knowledge, intrusion detection at the individual sensor level breaks new ground in SDN resource-constrained wireless networks. For the FDFF and FNI attacks, our hypothesis is that decentralized detection could be efficient if metrics related to the number of control packets exchange and the active state time (i.e., the time the node is not on sleeping mode) are monitored.

To test our hypothesis, the following metrics were monitored: the processing time, the transmitting time, the number of control packets received and the number of control packets transmitted. However, for compactness we will present results only the best performing metrics, i.e, we focus on the results from monitoring the transmitting time and the number of control packets received.[2] When a CP is detected, the sensor informs the Security application.

To reach the Security application, we include a packet in the Southbound protocol, exclusively for this purpose. In this packet, the sensor can include important information, such as the metric in which the anomaly was detected and a suspect address, among others. This packet is forwarded to the SDN controller, which sends this information to the Security application through the Network Services Abstraction Layer. Using this information, the security application can investigate which node or nodes have launched the attack and also execute mitigation strategies. In this section, we analyze the anomaly detection probability on every node running the CP detector; in Section VI-B, we propose and evaluate an attacker identification algorithm based on the distributed detection approach.

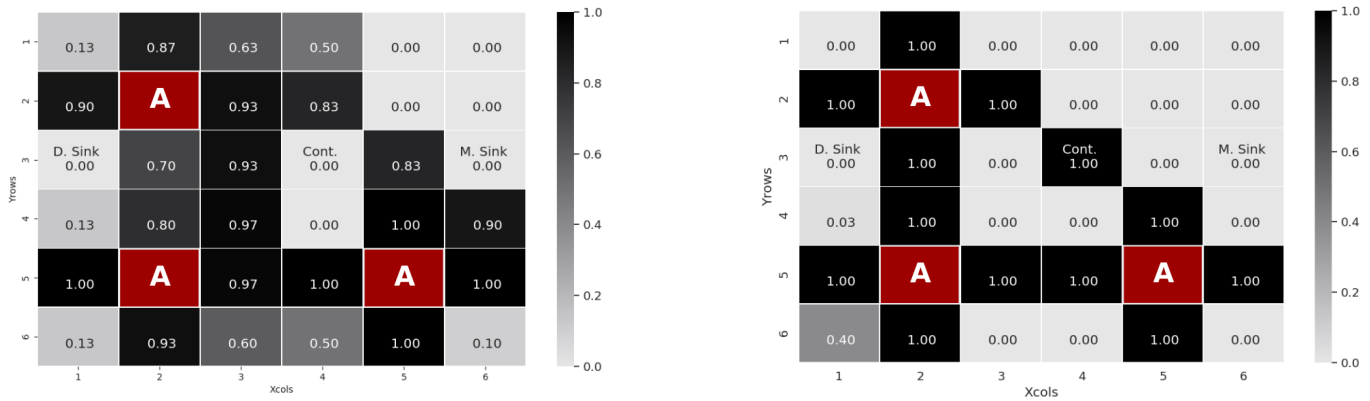
In the experiments based on Contiki 3.0 as follows, the transmitting time was obtained using Energest [33], a tool to monitor device hardware usage. Furthermore, the number of control packets was obtained by programming every node to print every packet received. Using the COOJA simulator serial output it is possible to copy this information into a text document for further analysis.

A. Experimental setup and results

A dataset of 120 simulations divided into two groups was generated, divided in two parts so that half concerned for a FDFF attack and the other half for a FNI attack. For both attacks, grid topologies of 36 and 100 nodes were simulated, whereby 10% of nodes were attackers. For these experiments, we prioritized detection accuracy over detection speed and thus the detector was configured using $\gamma = 0$ while we set the target $\alpha = 0.99$, and $m = 200$ according to the results from Section V.

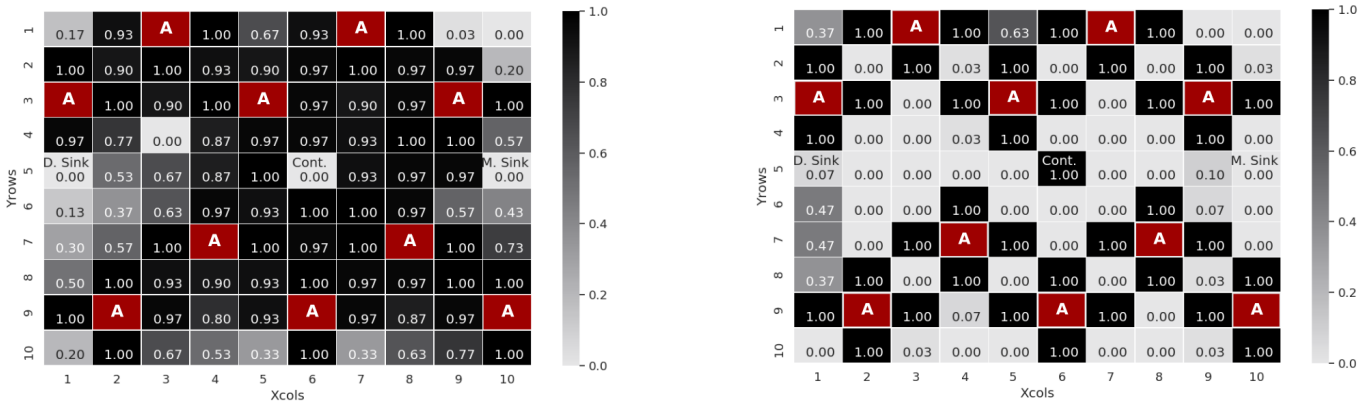
The detection performance was evaluated on every node monitoring each metric separately, i.e., running only one detector at a time due to memory constraints on the nodes. For this evaluation, the detection probability of every node on each scenario was evaluated. The same simulation parameters and attackers positions used for the centralized detection experiments, were maintained as summarized in Table III, while the attackers position are depicted in Figs. 5 and 6. The analysis of the detection performance is based on the probability of CP detection on each node and the location of nodes reporting high detection rates.

[2]The transmitting time is the time the node remains with the radio module turned. The control packets metric is the number of control packets transmitted. These packets include all southbound packets, except the data ones, and the neighbor discovery protocol packets.



(a) Transmitting time (b) Control packets received

Fig. 11. Detection probability heat maps for 36 nodes when the network is under FDFF attack. Each square represents a node in the network and the number inside them is the detection probability result. The red squares with an "A" inside are the attackers. (a) shows the results when monitoring the transmitting time and (b) shows the results when monitoring the control packets received.



(a) Transmitting time (b) Control packets received

Fig. 12. Detection probability heat maps for 100 nodes when the network is under FDFF attack. Each square represents a node in the network and the number inside them is the detection probability result. The red squares with an "A" inside are the attackers. (a) shows the results when monitoring the transmitting time and (b) shows the results when monitoring the control packets received.

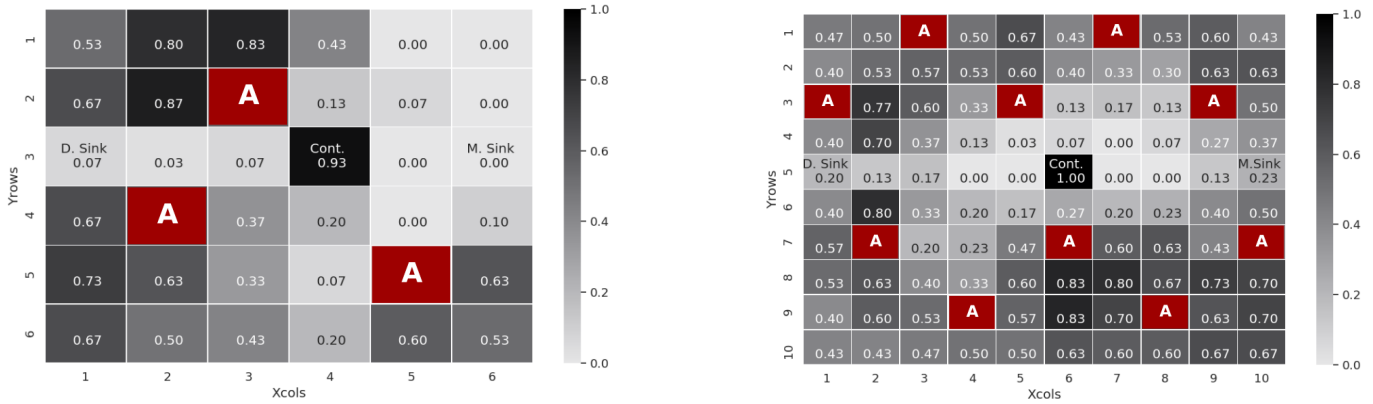
1) Results for a FDFF attack: We analyzed positioning of nodes and detection rates when monitoring the transmitting time and the control packets received. The corresponding heat maps are shown in Fig. 11 for a topology of 36 nodes. Note that i) when monitoring the transmitting time, the neighbors of the attackers have higher detection rates than farther nodes; and ii) when monitoring the control packets received, excluding the controller and the node in the lower left corner, all the nodes reporting an alarm are in the attacker neighborhood and have a $P_{DR} = 1$.

For a 100-node topology, a similar behavior was observed when monitoring the control packets received (Fig. 12b); however, when monitoring the transmitting time (Fig. 12a) it was observed that any node on the network can reach high detection rates. This is because when the network is under attack, the number of control packets increase and this impacts the radio usage of all the nodes forwarding those packets. In turn, the control packets received is a metric that impacts only the node that receives the packet.

In terms of detection time, the quantity $1 - S$ was analyzed for the 100-node case. The average $1 - S$ when monitoring the control packets received and the transmitting time are 0.84 and 0.90, respectively. Furthermore, the average $1 - S$ for the nodes neighboring the attackers is 0.96 and 0.92 when monitoring the control packets received and the transmitting time, respectively. Consequently, considering all the nodes that detected the attack, the detection is faster when monitoring the transmitting time, whereas on the nodes neighboring the attackers, the detection is faster when monitoring the control packets received.

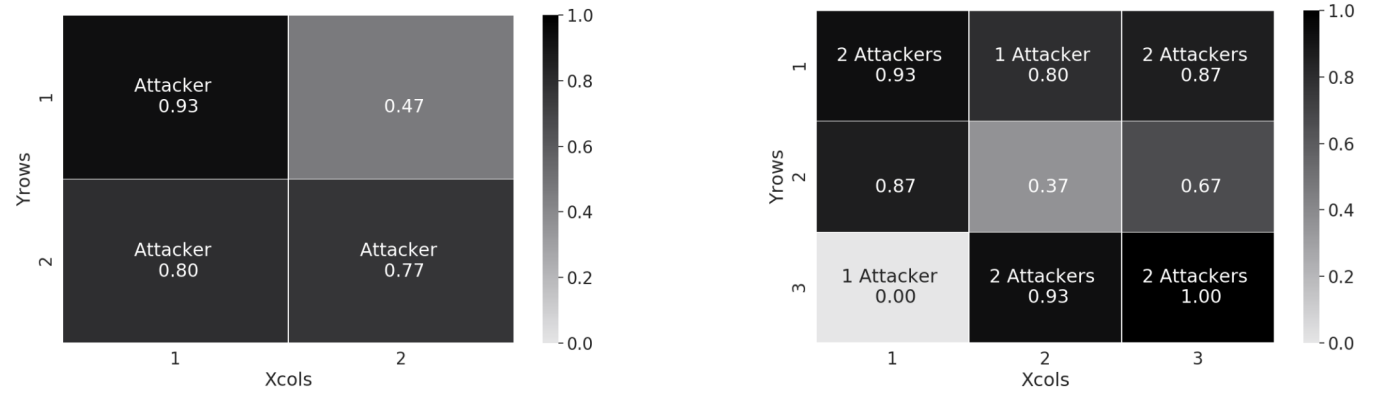
2) Results for a FNI attack: The detection rates when the network is under an FNI attack are lower compared to a FDFF attack. Many nodes report alarms with probabilities over 0.5 with an average $1 - S$ of 0.64 and 0.61 when monitoring the control packets received and the transmitting time, respectively; i.e, between 20% and 29% slower than for a FDFF attack detection.

Investigating the location of the nodes with higher detection rates in the topology, it was observed that when monitoring the control packets received, no clear conclusion for alerts risen in the immediate neighborhood of the attacker could be drawn, as shown in Fig. 13. Intuitively, only a portion of the neighborhood of the attacker nodes route their packets toward the controller through the attacker, in which case the network



(a) 36 nodes (b) 100 nodes

Fig. 13. Detection probability heat maps when the network is under FNI attack. Each square represents a node in the network and the number inside them is the detection probability result. The red squares with an "A" inside are the attackers. (a) shows the results for 36 nodes and (b) shows the results for 100 nodes.



(a) 36 nodes (b) 100 nodes

Fig. 14. Detection probability heat maps when the network is under FNI attack. Each square represents a group of nodes and the number inside them is the detection probability. (a) shows the results for 36 nodes and (b) shows the results for 100 nodes.

misconfiguration impacted them first. Due to this imbalance, we alternatively propose to perform CP detection per regions (areas) using data aggregation. To this end, the 36 nodes were divided into four groups and the 100 nodes into nine groups and one time series was generated per group. Each sample of this time series represented the sum of time series of all the nodes in the group, thus one CP detector was executed per group. In Fig. 14, the P_{DR} is depicted for 36 and 100 nodes when monitoring the control packets received. Excluding the groups that contained the controller, in all the cases, the detection probability achieved is better than that obtained by any of the nodes individually. This indicates that with data aggregation, we lose granularity but we gain in detection rates.

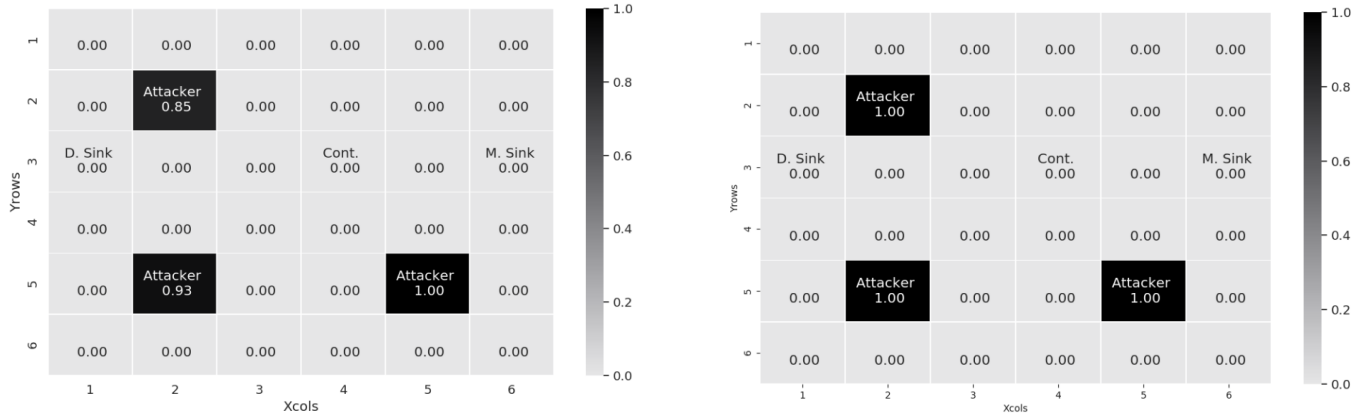
B. Attacker Identification

Next, we leverage the SDN properties by using the controller global view of the network to identify the attacker address or approximate the location based on the alarms reported by the nodes. It is worth mentioning that our proposal does not require the geographical location of the nodes, but only the network graph available at the SDN controller. We next present and evaluate an algorithm to locate the attackers under a FDFF or FNI attack.

1) Attacker detection for a FDFF attack: Our proposal is to identify the attackers IDs based on the alarms reported by their neighbors. To accomplish this, we programmed a register on the sensor nodes to store the addresses of the nodes sending data packets with unknown flow IDs. The register stores this information for the last ten packets received, corresponding to the slowest detection DTM (for $\gamma = 0$, $1 - S = 0.84 = 9.6$ samples on average). Algorithm 1 shows the proposed approach in pseudocode.

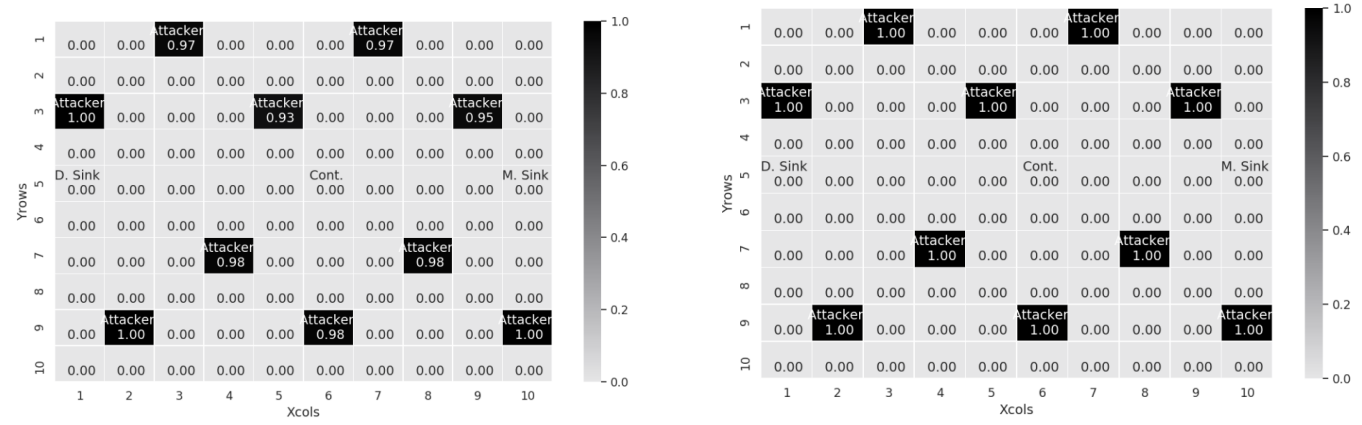
Fig. 15 allows observing that monitoring either the transmitting time or the control packets received, there were no misidentifications. When monitoring the control packets received, the identification probability was 1.00 for all the attackers, while when monitoring the transmitting time the identification probability was between 0.85 and 1.00. When evaluating the identification algorithm for 100 nodes (Fig. 16) we obtained excellent results as well; no misidentifications and identification probabilities over 0.93. In fact, when monitoring the control packets received the identification probability reached 1.00 for all the attackers.

2) Attacker detection for a FNI attack: Continuing the study on grouping of nodes in the case of a FNI attack, next the detection speed on every group is analyzed. Fig.



(a) Transmitting time (b) Control packets received

Fig. 15. Attackers identification probability using Algorithm 1 when the network is under an FDFF attack: 36-node case. Each square in the map represents a node in the network. The number in the squares represent the probability of this node being classified as an attacker.(a) shows the results when monitoring the transmitting time and (b) shows the results when monitoring the control packets received



(a) Transmitting time (b) Control packets received

Fig. 16. Attackers identification probability using Algorithm 1 when the network is under an FDFF attack: 100-node case. Each square in the map represents a node in the network. The number in the squares represent the probability of this node being classified as attacker.(a) shows the results when monitoring the transmitting time and (b) shows the results when monitoring the control packets received

Algorithm 1 FDFF attackers identification

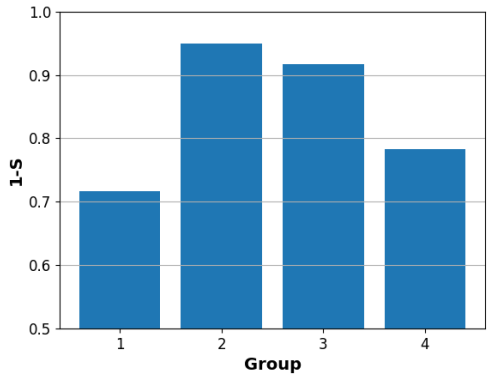
```
procedure FDFF_ATID(node_id, suspect])
    ; node_id: address of the node sending the alarm
    ; suspect: address number of the suspect detected
    ; suspect_counter: counter for suspect
    ; suspect_nei: # of neighbors of suspect
    Wait alarms[node_id, suspect]
    if new alarm then
        suspect_counter++
        suspect_nei = graph_information(suspect)
        if suspect_counter == suspect_neig then
            suspect is an attacker
        end if
    end if
end procedure
```

17 shows $1 - S$ (normalized DMT) for 36 and 100 nodes when monitoring the control packets received. For 36 nodes (Fig. 17a), it is shown that group 1 (the group without an attacker) has the lowest $1 - S$, which means this is the last group reporting an alarm. However, in the case of 100 nodes (Fig. 17b) the results did not show a similar trend. Therefore, no clear conclusion can be drawn in this case.

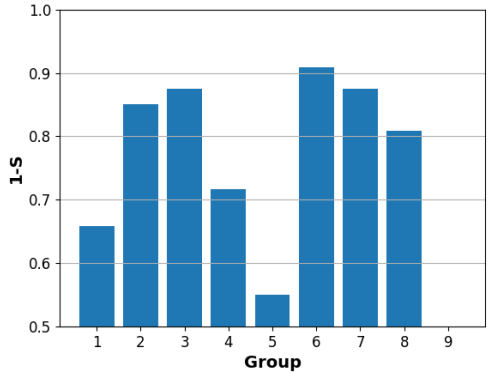
Summarizing Section VI, under a FDFF attack most nodes in the vicinity of the attackers detected the attack with a probability equal or over 90%. Under a FNI attack, the results detection rates on individual nodes were not consistent. By observing aggregate time series, the detection rates increased at the cost of losing granularity. In terms of detection speed, the distributed approach monitoring the control packets received was 5% faster than the centralized approach (Fig. 18b). In turn, when the network is under a FNI attack, the centralized approach was always faster. As regards the attacker identification, for FDFF attacks, Algorithm 1 is shown to identify attackers with a probability over 0.93 in all the cases. Conversely, for FNI attacks, no reliable relation between any metric or the presence of attackers in the groups was observed.

VII. Conclusions

In this work, we proposed online, CP-based, centralized and decentralized intrusion detection algorithms for SDWSN-

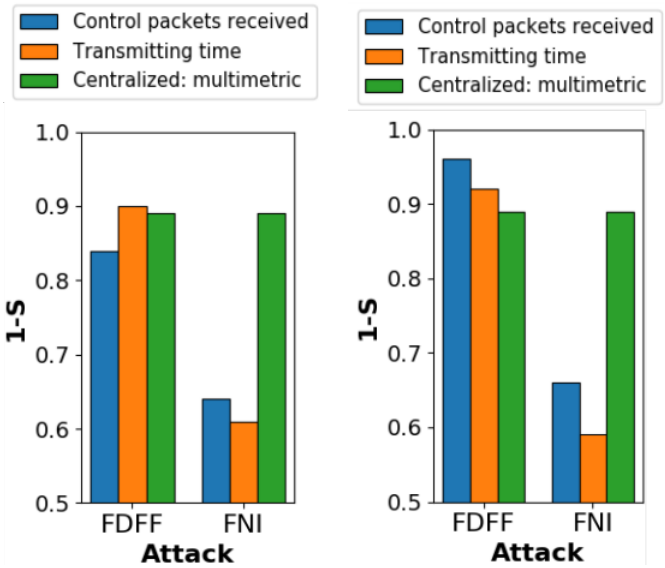


(a) 36 nodes



(b) 100 nodes

Fig. 17. Detection speed (1-S metric) for FNI detection by data aggregation when monitoring the control packets received. (a) shows the results for 36 nodes and (b) shows the result for the case with 100 nodes



(a) Average $1 - S$ value considering all the nodes for the distributed approach

(b) Average $1 - S$ value considering only the node near the attackers for the distributed approach

Fig. 18. Detection speed comparison between the centralized and the distributed approaches

constrained networks. The main strengths of our proposed detectors are the high detection rates, the identification of the type of attack and the localization or even identification of the attacker in some cases. The centralized approach leveraged the global view of the attack at the controller and allowed us to identify the type of attack; the distributed detection provided information that, in some cases, allows identifying the nodes launching the attack or at least a cluster of suspicious nodes.

The algorithms were evaluated through simulations using IT-SDN, Contiki-3.0 and the COOJA simulator, emulating TelosB motes. Topologies of 36 and 100 nodes were simulated, varying attackers proportionality ranging between 5%, 10% to 20% of the total number of nodes in the topology. The centralized detector was tuned to either maximize the detection rate or the detection speed. Our results showed detection probabilities over 0.96 in networks of 36 and 100 nodes when using the centralized approach and were able to identify the type of attack with a probability over 0.89. Furthermore, we observed a FDFF attackers identification with probability over 0.93 when using the distributed detection.

As future work, we intend to develop a full implementation of both approaches, compare their impact on the network performance and resource usage besides integrating both detectors into a single, hybrid approach.

ACKNOWLEDGMENT

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by the ELIOT project (ANR-18-CE40-0030 / FAPESP 2018/12579-7). Gustavo A. Nunez Segura is supported by Universidad de Costa Rica.

REFERENCES

[1] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proc. IEEE Proc.*, vol. 103, no. 1, pp. 14–76, Jan 2015.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[3] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements," *IEEE Access*, vol. 5, pp. 1872–1899, 2017.

[4] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1994–2008, 2017.

[5] S. W. Pritchard, G. P. Hancke, and A. M. Abu-Mahfouz, "Security in software-defined wireless sensor networks: Threats, challenges and potential solutions," in *2017 IEEE 15th Int. Conf. on Ind. Informat. (INDIN)*, 2017, pp. 168–173.

[6] F. Restuccia, S. D'Oro, and T. Melodia, "Securing the internet of things in the age of machine learning and software-defined networking," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4829–4842, 2018.

[7] G. Simoglou, G. Violettas, S. Petridou, and L. Mamatas, "Intrusion detection systems for RPL security: A comparative analysis," *Computers and Security*, vol. 104, 2021.

[8] S. S. Bhunia and M. Gurusamy, "Dynamic attack detection and mitigation in IoT using SDN," in *27th Int. Telecommun. Netw. and Appl. Conf. (ITNAC)*, Nov 2017, pp. 1–6.

[9] Y. Jia, F. Zhong, A. Alrawais, B. Gong, and X. Cheng, "Flowguard: An intelligent edge defense mechanism against iot ddos attacks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9552–9562, 2020.

[10] N. Ravi and S. M. Shalinie, "Learning-driven detection and mitigation of ddos attack in iot via sdn-cloud architecture," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3559–3570, 2020.

[11] A. Wani, R. S, and R. Khaliq, "SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL)," *CAAI Trans. on Intelligence Technol.*, vol. 6, no. 3, pp. 281–290, 2021.
[12] J. Li, Z. Zhao, R. Li, and H. Zhang, "Ai-based two-stage intrusion detection for software defined iot networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2093–2102, 2019.
[13] M. Bagaa, T. Taleb, J. B. Bernabe, and A. Skarmeta, "A Machine Learning Security Framework for Iot Systems," *IEEE Access*, vol. 8, pp. 114 066–114 077, 2020.
[14] D. Yin, L. Zhang, and K. Yang, "A DDoS Attack Detection and Mitigation With Software-Defined Internet of Things Framework," *IEEE Access*, vol. 6, pp. 24 694–24 705, 2018.
[15] C. Miranda, G. Kaddoum, E. Bou-Harb, S. Garg, and K. Kaur, "A collaborative security framework for software-defined wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, pp. 1–1, 2020.
[16] R. Wang, Z. Zhang, Z. Zhang, and Z. Jia, "ETMRM: An Energy-efficient Trust Management and Routing Mechanism for SDWSNs," *Computer Networks*, vol. 139, pp. 119 – 135, 2018.
[17] N. S. Gustavo, S. Skaperas, A. Chorti, L. Mamatas, and B. M. Cintia, "Denial of Service Attacks Detection in Software-Defined Wireless Sensor Networks," in *SecSDN IEEE Int. Conf. Commun. (ICC)*, Dublin, Ireland, Jun. 2020.
[18] G. A. N. Segura, A. Chorti, and C. B. Margi, "Multimetric online intrusion detection in software-defined wireless sensor networks," in *2020 IEEE Latin-American Conf. Commun. (LATINCOM)*, 2020, pp. 1–6.
[19] R. C. A. Alves, D. A. G. Oliveira, G. A. Nunez Segura, and C. B. Margi, "The Cost of Software-Defining Things: A Scalability Study of Software-Defined Sensor Networks," *IEEE Access*, vol. 7, pp. 115 093–115 108, Aug 2019.
[20] M. Du and K. Wang, "An SDN-Enabled Pseudo-Honeypot Strategy for Distributed Denial of Service Attacks in Industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 648–657, 2020.
[21] G. A. N. Segura, C. B. Margi, and A. Chorti, "Understanding the Performance of Software Defined Wireless Sensor Networks Under Denial of Service Attack," *Open Journal of Internet Of Things (OJIOT)*, 2019.
[22] A. Chorti, C. Hollanti, J.-C. Belfiore, and H. V. Poor, "Physical Layer Security: A Paradigm Shift in Data Confidentiality," in *Physical and Data-Link Security Techniques for Future Communication Systems*. Cham: Springer International Publishing, 2016, pp. 1–15.
[23] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in Software Defined Networks: A Survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2317–2346, 2015.
[24] M. P. Singh and A. Bhandari, "New-flow based DDoS attacks in SDN: Taxonomy, rationales, and research challenges," *Computer Communications*, vol. 154, pp. 509 – 527, 2020.
[25] D. B. Rawat and S. R. Reddy, "Software Defined Networking Architecture, Security and Energy Efficiency: A Survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 325–346, 2017.
[26] S. Shin and G. Gu, "Attacking Software-Defined Networks: A First Feasibility Study," in *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 165–166.
[27] A. Aue and L. Horvath, "Structural breaks in time series," *Journal of Time Series Analysis*, vol. 34, no. 1, pp. 1–16, 2013.
[28] S. Fremdt, "Asymptotic distribution of the delay time in page's sequential procedure," *Journal of Statistical Planning and Inference*, vol. 145, pp. 74 – 91, 2014.
[29] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Software-defined networking (SDN): Layers and architecture terminology," Internet Research Task Force (IRTF), Tech. Rep., 2015.
[30] H. V. Poor and O. Hadjiliadis, *Quickest detection*. Cambridge University Press, 2008.
[31] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-Level Sensor Network Simulation with COOJA," in *Proc. IEEE Conf. Local Comput. Netw. (LCN)*, Nov 2006, pp. 641–648.
[32] T. Luz, G. Nunez, C. Margi, and F. Verdi, "In-network performance measurements for Software Defined Wireless Sensor Networks," in *16th IEEE Int. Conf. Netw., Sens. and Control (ICNSC 2019)*, May 2019.
[33] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based on-line energy estimation for sensor nodes," in *Proceedings of the 4th Workshop on Embedded Networked Sensors*, ser. EmNets '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 28–32.

Gustavo A. Nunez Segura is a PhD candidate at Universidade de São Paulo. He received his M.Sc. degree (2018) in Electrical Engineering from Universidade de São Paulo and his B.Sc. in Electrical Engineering from Universidad de Costa Rica. His main research interests include energy consumption and security in wireless sensor networks and software-defined networking

Arsenia Chorti is a Professor of Communications and Networks at ETIS UMR8051, CY Cergy Paris University, ENSEA, CNRS in France and has served as a Lecturer and Senior Lecturer at the Universities of Essex and Middlesex UK. She has been a chartered engineer from the Technical Chambers of Greece since 2007, Senior IEEE member since 2020, a member of the IEEE P1951.1 Working Group on Smart Cities Standardization and of the IEEE INGR Working Group on Security. Between 2017-2020 she served as a member of the IEEE Teaching Awards Committee. Her research interests include 5G and 6G wireless, IoT, physical layer security, root cause analysis and anomaly detection.

Cintia Borges Margi obtained her Ph.D. in Computer Engineering at the University of California Santa Cruz (2006), and her Habilitation (Livre Docencia) (2015) in Computer Networks from the Universidade de São Paulo. She has been Associate Professor in the Computer and Digital Systems Engineering Department at Escola Politecnica – Universidade de São Paulo (EPUSP) since 2015, where she started as an Assistant Professor in 2010. In 2007-2010 she was an Assistant Professor at Escola de Artes, Ciencias e Humanidades da Universidade de São Paulo (EACH-USP). Her research interests include: wireless sensor networks, security and software-defined networking.