Secure and Energy-Efficient Network Topology Obfuscation for Software-Defined WSNs

Manaf Bin-Yahya and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Network topology obfuscation (NTO) is generally considered as a promising proactive mechanism to mitigate traffic analysis attacks. The main challenge is to strike a balance among energy consumption, reliable routing, and security levels due to resource constraints in sensor nodes. Furthermore, software-defined wireless sensor networks (WSNs) are more vulnerable to traffic analysis attacks due to the uncovered pattern of control traffic between the controller and the nodes. In this article, a new energy-aware NTO mechanism is proposed, which maximizes the attack costs and is efficient and practical to be deployed. Specifically, first, a route obfuscation method is proposed by utilizing ranking-based route mutation, based on four different critical criteria: 1) route overlapping; 2) energy consumption; 3) link costs; and 4) node reliability. Then, a sink node obfuscation method is introduced by selecting several fake sink nodes that are indistinguishable from actual sink nodes, according to the *k*-anonymity model. As a result, the most suitable routes and sink nodes can be selected, and a highest obfuscation level can be reached without sacrificing energy efficiency. Finally, extensive simulation results demonstrate that the proposed methods can strongly mitigate traffic analysis attacks and achieve effective NTO for software-defined WSNs. In addition, the proposed methods can reduce the success rate of the attacks while achieving lower energy consumption and higher network lifetime.**

***Index Terms*—IoT, moving target defence (MTD), obfuscation, software-defined WSN (SDWSN), security, wireless sensor network (WSN).**

I. Introduction

WIRELESS sensor networks (WSNs) integrated with IoT are considered as an indispensable enabler of the IoT paradigm [1]. WSNs play a vital role in industrial IoT (IIoT) to support automation, monitoring, and control functions [2]. In WSN-enabled IIoT networks, a multihop transmission is used to connect a large number of sensor devices. A software-defined networking (SDN) paradigm introduces solutions that add programmable and flexible features to WSNs [3]. In SDN, a centralized controller acts as an intelligent element in the architecture whereby (over-the-top) services are implemented above this controller [4]. An SDN paradigm intends to split between the data and the control planes [5]. As a result, the SDN-enabled devices (such as sensors) turn into nonintelligent devices in terms of routing decisions. For the routing function, the controller provides the sensor nodes with flow rules for message forwarding. The controller collects information about the network to create the routing map. To assign a path for a particular flow, the controller uses the shortest path algorithm to select the best route based on the hop counts from all feasible routes in the network [6].

WSN-enabled IIoT networks are highly vulnerable to traffic analysis attacks similar to conventional WSNs [7]. In these passive attacks, the adversary tries to obtain detailed knowledge about the network behavior (topology) [8]. Usually, the goal of these attacks is to identify flow routes and high-profile nodes that play significant roles in network communication, including sink nodes, intermediate nodes, and shared nodes of both control and data traffic. After discovering these nodes, the adversary can compromise (hijack) these nodes later or launch a DDoS attack. In most cases, sensed data and control traffic are transmitted along paths from source nodes to determined destinations such as sink nodes. This produces pronounced traffic patterns that can be revealed by monitoring the network traffic. Thus, the network cannot be guarded by applying only conventional privacy solutions such as encryption [9]. Moreover, some traffic analysis attacks from the conventional WSNs are exacerbated in the software-defined WSNs (SDWSNs). Recall that SDN introduces communication between the controller and the nodes for the control plane. Therefore, due to the evident pattern of this additional control traffic, SDWSNs are more vulnerable to traffic analysis attacks.

Network topology obfuscation (NTO) is an effective technique for hiding the entire network and securing it against the traffic analysis adversary as a proactive defense. The primary goal of the NTO solution is to minimize the damage level and maximize the cost to the adversary to launch efficient attacks. The damage level determines how successful the attack is, e.g., how many flows may drop down when flooding a link. Network reliability is classed as an essential design concern alongside the security issue in WSN-enabled IIoT networks. WSNs are supposed to be operable for an extended lifetime and should overcome routing challenges. By applying NTO, each node decides to hide the routes of messages inside the network; meanwhile, the communication of the whole network is not harmed. Energy constraint and resource utilization are critical issue in the WSNs [2], [10]. Therefore, any NTO mechanism must consider both the energy and the security issues. In other words, energy consumption and traffic overhead should be highly optimized when applying defense mechanisms. Notably, it is challenging to entirely

Manuscript received 27 July 2021; revised 21 November 2021; accepted 7 January 2022. Date of publication 20 January 2022; date of current version 24 January 2023. *(Corresponding author: Manaf Bin-Yahya.)*

The authors are with the Department of Computer and Electrical Engineering, University of Waterloo, ON N2L 3G1, Canada (e-mail: mbenyahya@uwaterloo.ca; sshen@uwaterloo.ca).

Digital Object Identifier 10.1109/JIOT.2022.3144873

hide specific nodes in the network, such as the sink node, without forcing excessive overhead. Thus, it is reasonable to consider maximizing the attack cost to the adversary instead of proposing unpractical solutions. Conventional NTO solutions [11]–[15] still need improvements to be suitable for the WSNs' application. Different from those proposals, a solution is proposed that jointly considers energy consumption and obfuscation level (defense performance).

In this article, an SDN-based NTO solution is proposed to hide the network from the traffic analysis adversary. SDN allows for the setup of adaptive flow rules in the network. In the proposed solution, all decisions are executed by the controller rather than the sensors themselves. There are no exchanges among sensors, which is the main feature of existing proposals for conventional WSNs. Moreover, the proposed NTO is an energy-aware solution that is especially suited to the characteristics of WSNs and exhibits several advantages contrary to some existing solutions. This research aims to design obfuscation techniques to increase the cost for an adversary to discover the network topology. The proposed techniques aim to secure the network against network traffic attacks, such as sniffer attacks, link-flooding attacks, CrossPath attacks [16], and heuristic attacks [15], [17]. Specifically, the main contributions of this research can be summarized as follows.

1) An SDN-based NTO solution is proposed to secure the network against network traffic attacks. The proposed solution provides load balancing between security requirements and resource and QoS restrictions.
2) A ranking-based route mutation (RM) mechanism is proposed where paths are selected for flows' routes to obfuscate the high-profile nodes in the network. These paths are selected according to several criteria that ensure a high obfuscation level of the network based on route overlapping and consider the compatibility of WSN requirements, such as energy consumption and link cost. Moreover, multiple mutated routes can be generated to deceive the adversary and provide additional obfuscation level.
3) A sink obfuscation technique is proposed for fully centric WSNs. Multiple fake sink nodes are selected to deceive the adversary who aims to locate the sink node. The selection of fake sink nodes approach jointly considers the residual energy of nodes and the gained obfuscation level. The approach further uses a fitting parameter that considers the residual energy of the selected fake sink nodes' neighbors due to the expected additional communication overhead in their zone.
4) The proposed solutions are evaluated by showing the network and security performance and conducting a performance comparison with some existing solutions. Realistic adversary models are defined. An attack scenario is provided for each adversary model associated with the damage level evaluation parameter. The extended performance evaluation shows that the proposed NTO techniques outperform the existing solutions to defend the network against traffic analysis attacks while achieving lower overhead and resources consumption. Moreover, the cost of the proposed solution is moderate and applicable to large sensor networks.

The remainder of this article is organized as follows. A review of related works is discussed in Section II. In Section III, the system and threats models are introduced and the design goals are defined. The details of route and sink obfuscations are provided in Sections IV. and V, respectively. Next, a performance evaluation of the proposed techniques is provided in Section VI. Finally, a conclusion of this article is provided in Section VII.

II. Related Works

Several obfuscation mechanisms are proposed in the literature. These mechanisms aim to secure different types of networks. However, general SDN-based solutions [18] are not applicable for resource-constrained networks such as WSNs. Solutions for general wireless networks, such as multihop mobile networks, have a high energy consumption [19], [20]. Additionally, most of these solutions require high computational and storage resources. Several research studies are proposed to achieve location privacy in conventional WSNs using cryptographic approaches [21]. However, these approaches fail against adversaries that launch traffic analysis attacks. There are several proposed defense techniques against traffic analysis attacks in WSNs [9]. There are two types of solutions: 1) noncentric and 2) centric techniques. The noncentric solutions (standalone or cooperative) have a higher rate of energy consumption and a higher cost (in terms of path length, E2E delay, etc.). For example, the probability-based routing protocols in [17] rely on broadcasting fake packets from fake sources concurrently with the transmission of real packets from the real source nodes to deceive the adversary. In addition, the discovery of alternative routes requires more broadcasting messages as deceptive traffic or to collect the relevant information from neighbor nodes; this results in additional energy consumption and higher overhead [22]. Random walk (RW) protocols [13] deliver the messages through a random route every time. Ring routing, an improved version of the RW protocol, is proposed in [23]. Liu *et al.* [23] proposed a multirepresentative refusion (MRRF) data collection technique. MRRF is designed to ensure acceptable energy consumption and end-to-end delay of the RW ring routing. However, the technique has demonstrated some performance limitations. The technique considers only energy factors, and no security constraints are introduced.

The existing centric solutions, such as SDN-based solutions, have their limitations. Duan *et al.* [12] proposed a proactive random RM mechanism against sniffer and DoS attacks. The selected routes are dynamically and randomly changed while preserving QoS end-to-end connectivity. However, multiple uncrossed routes for each flow are required. The mechanism ensures that a previously selected route consisting of certain links must not be selected for the current route. It is challenging to satisfy this requirement in WSNs topology. Zhou *et al.* [11] proposed a scalable node-centric RM

(SNcRM) technique that formulates the problem into a signature matching problem and solves it by a binary branch and bound method. The technique obfuscates the network topology by decreasing the variety of historically accumulative traffic volume among the SDN-enabled nodes. The SDN controller recognizes previously highly loaded nodes with high exposure risks and finds alternative routes for their traffic flows via other lowly loaded nodes. This is determined based on the accumulative traffic of the node. Rauf *et al.* [14] proposed the secure route obfuscation (SRO) scheme, which is an SDN-based solution. To obfuscate the network, the controller randomly generates random routes for each communicating pair of nodes. Only a reliability score proposed in [24] is considered toward the route selection. No energy or security constraints are designed to select routes. Thus, SRO shows weak performance in terms of energy and security when applied to WSNs due to the uncontrolled randomness. Chai *et al.* [25] proposed a sink obfuscation technique against a global adversary based on the k-anonymity model. At least k nodes are selected in the network to mimic the sink node. Thus, the traffic loads around these nodes make the sink area indistinguishable. However, due to the significant overhead of the fake sink nodes' deceptive traffic, this solution has comparatively costly energy consumption. The network's lifetime is decreased due to every node sending messages to these fake sink nodes when sending the real one. Moreover, the authors do not consider acknowledgment (ACK) messages. This limits the solution to not be applied to SDWSNs, which have many built-in ACK-based messages. In addition, the data aggregation nodes are the same selected fake sink nodes, thereby leading to a traffic collision and low message delivery and reliability. Baroutis and Younis [15] proposed the preserve location anonymity through uniform distribution of traffic volume (PLAUDIT) technique to obfuscate WSNs. This technique achieves uniform distribution of traffic volume by injecting deceptive messages. Several dedicated nodes are selected to generate the deceptive flows that challenge the adversary's mission to reveal the network topology. In addition, the deceptive traffic rate is determined to balance the traffic density across the network and avoid network overhead. The authors try to achieve robust anonymity with load balancing and energy consumption. However, PLAUDIT fails to hide the centralized architecture of SDWSNs, especially from the heuristic traffic analysis attacks. Many research studies [26] define mutated identification methods. However, for WSNs, changing IP addresses too frequently may cause serious ramifications, including service interruptions, routing inflation, delays, and security violations [27]. Moreover, these techniques do not secure the network from a traffic analysis attack, for example, when the adversary uses physical equipment to collect the network information. All solutions described above attempt to obfuscate network topology or enhance the anonymity of certain high-profile nodes. Nonetheless, different from those countermeasures, in this article, a balance between energy consumption and obfuscation level (defense performance) is achieved. The proposed mechanisms minimize energy consumption, prolong the network lifetime, and suffer lower attack success rates.

TABLE I
TABLE OF NOTATIONS

Notations	Descriptions
$G(V, E)$	Network graph (WSN) G, where V is the set of vertices (nodes) and E is the set of edges (links).
$\|V\|$	Number of sensor nodes in the network.
e_{uv}	Direct connection (link) between nodes u and v.
CR_v	Communication range of node v.
d_{uv}	Distance between node u and v.
$Energy_v^0$	Initial residual energy of node v.
$E_{Tx}(l, d)$	The energy consumed to transmit l bytes for distance d.
$E_{Rx}(l)$	The energy consumed to receive l bytes.
$Energy_v$	The current (residual) energy level of node v.
f	Data flow that is defined by a source node and a destination node.
F_t	Set of data flows at time t ($F_t = \{f_1, f_2, ..., f_{\|F_t\|}\}$).
X^f	Boolean variable: $X^f = \{x_{v_1}^f, x_{v_2}^f, ..., x_{v_{\|V\|}}^f\}$ $x_{v_1}^f$ determines if node v_1 is assigned to data flow f.
F_t^c	Set of control flows at time t ($F_t^c = \{f_1^c, f_2^c, ..., f_{\|F_t\|}^c\}$).
X^{f^c}	Boolean variable: $X^{f^c} = \{x_{v_1}^{f^c}, x_{v_2}^{f^c}, ..., x_{v_{\|V\|}}^{f^c}\}$ $x_{v_1}^{f^c}$ determines if node v_1 is assigned to control flow f^c.
Δt	Control period (time window).
Route Obfuscation:	
C_v^{th}	Capacity limit (maximum number of flow rules (entries)) of the flow table for node v.
L^{th}	Route length limit in terms of the numbers of hops.
s_v	Similarity score of node v (equation 12).
h_v	History score of node v (equation 14).
k_r	Number of multiple mutated routes.
Sink Obfuscation:	
$\mathcal{S}$	Set of real sink nodes.
$\hat{\mathcal{S}}$	Set of nodes that are selected to be fake sink nodes.
k_s	Number of fake sink nodes in $\hat{\mathcal{S}}$.
th_E	Minimum energy level to be selected as a fake sink node.
$Fit_E(v)$	A score that determines how fit node v to be selected as a fake sink node (equation 19).
$v.H(k)$	Set of node v's neighbours that have k hops connection.
$\|v.H(k)\|$	Number of nodes that have k hops connection with node v.
$\Psi(\mathcal{S}, \hat{\mathcal{S}})$	Travel cost of the minimum spanning tree of real and fake sink nodes.
Ψ_{min}	Minimum travel cost threshold of minimum spanning tree of real and fake sink nodes.
th_d	Minimum distance threshold between the sink node and any fake sink node.
$v.cell()$	Cell's ID that node v belongs to.

III. Problem Formulation

In this section, first, the system and threat models are presented. Finally, the design goals of the proposed solution are discussed. A list of notations is given in Table I.

A. System Model

The system model consists of several components, namely, the SDN controller, sink nodes, and SDN-enabled sensor nodes, as shown in Fig. 1. The network is modeled as a graph $G(V, E)$, where V is the set of vertices (nodes) and E is the set of edges (links). The SDN controller has the supervisory view of the network and is responsible for flow management. The controller receives the statistical updates from the underlying network components to build comprehensive network maps. Based on these maps, the controller can provide several services efficiently, such as routing, network management, and security. The sink node, which connects the network and the controller, has powerful resources. The network is deployed randomly, and network nodes are homogeneous, which means every sensor node has the same communication range (CR)

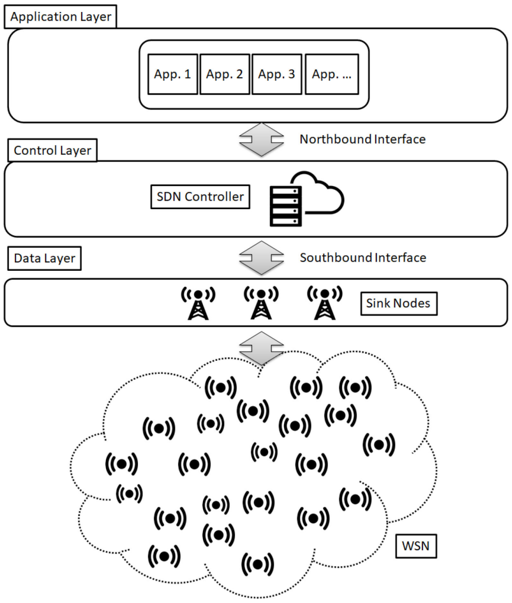


Fig. 1. SDWSNs.

and initial residual energy (Energy^0). Node i is connected to node j with edge e_{ij} only if it is within its CR ($d_{ij} \leq \mathrm{CR}_i$). In this research, a well-known transmission energy model is used [28]. The sensor node consumes $E_{Tx}(l, d)$ when it sends l bytes for distance d (1). While, it consumes $E_{Rx}(l)$ when it receives l bytes, $E_{Rx}(l) = lE_{\mathrm{elec}}$

$$E_{Tx}(l, d) = \begin{cases} lE_{\mathrm{elec}} + l\epsilon_{fs}d^2, & d < d_0 \\ lE_{\mathrm{elec}} + l\epsilon_{\mathrm{amp}}d^4, & d \geq d_0. \end{cases} \tag{1}$$

E_{elec} denotes the transmission circuit loss (50 nJ/bit). Depending on the distance between the sender and receiver nodes, the free space (d^2 power loss) or the multipath fading (d^4 power loss) channel models are applied. The energy needed in both models for power amplification is ϵ_{fs} (10 pJ/bit/m^2) and ϵ_{amp} (0.0013 pJ/bit/m^4). After each transmission, the energy level is updated for the sender i and receiver j nodes ($\mathrm{Energy}_i = \mathrm{Energy}_i - E_{Tx}(l, d_{i,j})$ and $\mathrm{Energy}_j = \mathrm{Energy}_j - E_{Rx}(l)$, respectively).

Each node has a flow table. The SDN controller is responsible for updating the flow tables of each node in the network. If there is no particular rule for an incoming message, then a packet-in control message must be sent to the controller to obtain a new routing rule. When the controller receives the packet-in message, it responds with a packet-out message, which contains the new flow entry [3]. To maintain network control and keep the flow table updated at all times, SDN provides several services [16]. Packet service manages packets exchanged between the control and data planes. Flow-rule service installs or updates rules in sensors via flow-mod messages. The topology service maintains the topology of sensors and links, discovers new sensors and tracks their locations, and establishes the control channel between sensors and controllers via several handshake messages. The liveness of sensors is periodically checked via echo request and echo reply messages. Therefore, when a sensor node in the network stops working due to energy exhaustion, the controller will be informed. Flow-metrics service is responsible for collecting flow statistics. It periodically queries the flows on network devices via stats requests and replies.

The controller defines F as the flow set of all computed flows in a certain control period (time window) Δt ($F = f_1, f_2, \dots, f_i$). A Boolean variable is used to indicate whether or not a node is selected as a part of flow f as follows:

$$x_v^f = \begin{cases} 1, & \text{this node is selected as intermediate node in } f \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

There are no dedicated links between nodes in wireless networks as the wireless node transmits the packets to the medium within its CR. Thus, the flow is defined as follows:

$$X^f = \left\{ x_1^f, x_2^f, \dots, x_{|V|}^f | \forall f \in F \right\}. \tag{3}$$

Moreover, the controller keeps the prior calculated flow sets from the previous control periods in a flow set matrix $F^T = [F_t, F_{t-\Delta t}, F_{t-2\Delta t}, \dots, F_{t-T\Delta t}]$ where t is the current time. The controller does not determine the routes for the control flow. The routes are determined by the nodes themselves using a discovery approach [6]. However, the controller has knowledge of these paths. The control flow set for all nodes in the network is defined as F^c ($F^c = f_1^c, f_2^c, \dots, f_{|V|}^c$). A similar Boolean variable $x_v^{f^c}$ is used to indicate whether or not a node v is selected as a part of control flow f^c. The control flow can be defined as follows:

$$X^{f^c} = \left\{ x_1^{f^c}, x_2^{f^c}, \dots, x_{|V|}^{f^c} | \forall f^c \in F^c \right\}. \tag{4}$$

The node that is a part of a data flow X^f and a control flow X^{f^c} is considered as a shared node and vulnerable for the CrossPath attack (Section III-B).

B. Threat Model

In this research, we assume an outsider adversary, which is an unauthorized user who does not have permission to control the sensor network. The adversary wants to attack the network availability, however; s/he cannot attack the controller directly. To launch an effective attack, the adversary must learn the network topology and identify the high-profile nodes that play significant roles in network communication, including sink nodes, intermediate nodes, and shared nodes of both control and data traffic. Most messages are transmitted along paths that have high-profile nodes. This produces pronounced traffic patterns that reveal traffic path information, direction, and thus, the location of these nodes. The adversary must first launch a traffic analysis attack, a remote software-based attack, or a physical attack on the network. This adversary can hijack (capture) sensor nodes; consequently, s/he is capable of obtaining its flow table, eavesdropping communications within the node's range (passive monitoring), and revealing some statistics about the neighborhood. The adversary can gather information about the network without being detected, as the sensor node will continue to act normally with no malicious actions. The adversary can compromise only a small number of nodes at any reasonable cost (time). In this research, without loss of generality, we consider that the adversary can only compromise one node during a control period Δt.

In this research, an adversary can launch a sniffer attack, link-flooding attack, a CrossPath attack, or a heuristic attack. Also, several attack scenarios and the damage level evaluation based on these attacks are provided in Section VI-B. In the *sniffer attack*, the adversary captures and analyzes network communication packets. A sniffer adversary is capable of eavesdropping on the data traffic of nodes or links, monitoring network status, and stealing sensitive data. The overlapping points of data flows are convenient for the adversary during a sniffer attack. The *link-flooding attack* is a DoS attack in which the adversary targets a number of links by flooding packets. The adversary can disrupt a limited number of links or nodes without being detected for a specific time. The selection of the target set of links is based on the belief of what is significant within this set for certain flows. Therefore, the adversary uses data reconnaissance to gain knowledge about the high-profile data forwarding nodes.

A *CrossPath attack* is a link-flooding DoS attack that targets the shared links between the control and data traffic in the in-band control SDN [16]. A probing technique called adversarial path reconnaissance (APR) is used to find the target links. The technique was inspired by the key observation that the delay of a control path is higher if a short-term burst of the data traffic passes through the shared links. Thus, an adversary can use a compromised node to identify the key data paths by generating data traffic and measuring the delay variations of the control paths. To identify a shared link using APR, the target data path must cross with a control path of a sensor belonging to the data path. After the discovery of these target links, the adversary will be able to launch a link-flooding attack. In this research, we assume an ideal case in which the adversary can identify all the possible shared paths using APR. In the *heuristic attack*, the adversary goal is to maximize the attack on the high-profile nodes. To achieve the attack goal, the adversary uses a heuristic approach to move from one node to another [17]. A greedy heuristic expands the attack graph by selecting the most profitable node based on an evaluation function [29]. Using the evaluation function, the adversary can reveal the traffic pattern of neighbor nodes based on traffic volume and communication directions gathered through passive monitoring. This can be achieved either physically or by a software-based method. The adversary continues to move until s/he finds the high-profile targets. The adversary may face a deadlock, i.e., the candidate list of possible targets is empty. Then, the adversary chooses the next target randomly.

C. Design Goals

The proposed solution aims to achieve the following goals.

1) *High Obfuscation Level:* The primary goal of the NTO solution is to minimize the damage level and maximize the cost to the adversary to launch efficient attacks.
2) *Reliable Routing:* The routing path of each flow must be provided with high reliability.
3) *Energy Efficiency:* The NTO solution needs to be energy efficient to eliminate the side effects of defense mechanisms and guarantee network performance. Primarily, the objective function of the problem is formulated as maximizing the obfuscation level achieved by a certain solution Ω and minimizing the cost that the network must pay for this defense level ($\min \mathrm{Cost}(\Omega)\ \&\ \max \mathrm{Obf}(\Omega)$).

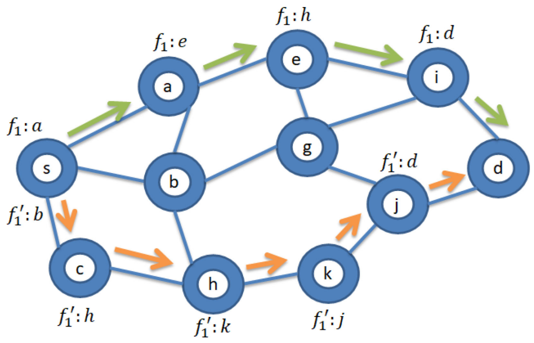


Fig. 2. Example of shortest route for flow f compared to the muted route (f').

IV. Route Obfuscation

The pattern of data and control traffic can be revealed as the network uses shortest path routing. Moreover, measuring the control delays may reveal the shared paths between data and control traffic [16]. Thus, to provide route obfuscation, a ranking-based RM (rRM) mechanism is proposed. In the proposed solution, paths are ranked based on several criteria. Then, based on this ranking, the controller sets the mutated path for each flow under two key considerations. The first consideration is the total path cost, which combines several key criteria to achieve reliable and energy-aware routing. The second consideration is the obfuscation level that is gained by the given paths. This determines the defence efficiency. Therefore, the route obfuscation problem is formulated as follows:

$$\min_{\forall f \in F_t} \mathrm{Cost}(f)\ \&\ \max_{\forall f \in F_t} \mathrm{Obf}(f). \tag{5}$$

When the controller sets the mutated path for a flow, it will update the flow rules for each node in the selected paths. Thus, all these rules will be deployed in the network when required. When the flow rule expires, the controller redetermines the mutated path for the active flows and updates the network. As shown in Fig. 2, in normal operations, the shortest path (f_1) is set for a flow from node s to node d. However, using RM, a different path (f_1') is set for the flow.

A. Path Cost Criteria

The cost of a path is the cost of all links in that path

$$\mathrm{Cost}(f) = \sum_{\forall e_{uv} \in f} \mathrm{Cost}(e_{uv}). \tag{6}$$

Four key criteria are defined for determining link cost: 1) node energy level; 2) edge energy cost; 3) node table flow capacity; and 4) node reliability. The residual energy is a significant factor in selecting a particular node in the path. The energy level's weight of a node is calculated as follows:

$$\varepsilon_v = \frac{\mathrm{Energy}_v}{\mathrm{Energy}_v^0}. \tag{7}$$

The energy consumption model of packet transmission is a function of distance. As a result, the distance d_{uv} is considered to determine the edge energy cost for edge e_{uv}. The higher distance results in the higher energy consumption (1). Thus, the edge energy cost of e_{uv} is computed as follows:

$$e^c_{uv} = \frac{d_{uv}}{\mathrm{CR}_u} \tag{8}$$

where the value is $0 < e^c_{uv} \leq 1$ for direct (valid) edges, and when there is no direct connection, it will be excluded by the algorithm. On the other hand, due to the limited capacity of the flow table, the capacity score C^c_v of a node is defined by dividing the number of flow rule entries at the previous control period over the capacity limit of the flow table for the node

$$C^c_v = \frac{\sum_{\forall f \in F_{t-\Delta t}} x^f_v}{C^{th}_v}. \tag{9}$$

We assume that C^{th}_v is equal for all the sensor nodes ($\forall v \in G.V$). The flow set $F_{t-\Delta t}$ is used because they are the current deployed flow rules in the network. Finally, the node reliability in packet transmission is considered to avoid nodes that have a history of higher failure rate due to the congestion, for example. A Bayesian method is used to compute the reliability score of a node in transmitting packets based on the total number of successful transmissions $N^{\text{success}}_{\text{trans}}$ and the total number of transmissions $N^{\text{all}}_{\text{trans}}$ of this node [30]

$$\mathrm{Rel}_v = \frac{N^{\text{success}}_{\text{trans}} + 1}{N^{\text{all}}_{\text{trans}} + 2}. \tag{10}$$

All of the above scores are normalized, and their values are represented between 0 and 1. Therefore, the cost of a link e_{uv} is calculated based on the above scores as follows:

$$\begin{aligned} \mathrm{Cost}(e_{uv}) = {} & \omega_\varepsilon(1-\varepsilon_v) + \omega_e e^c_{uv} + \omega_C\left(1 - C^c_v\right) \\ & + \omega_r(1-\mathrm{Rel}_v) \end{aligned} \tag{11}$$

where $\omega_\varepsilon + \omega_e + \omega_C + \omega_r = 1$. Assigning the weighting parameters depends on specific application to utilize the network performance.

B. Route Obfuscation Level

To determine the obfuscation level of the generated mutated paths of flows, two parameters are defined: 1) similarity s and 2) history h. These parameters determine the overlapping criteria of route selection for a node. The similarity s score is used to compute the overlapping between paths in the flow set F in the same control period (time window) Δt. Also, s score determines the number of shared intermediate nodes between the data flow F and the control flow F^c

$$s_v = \sum_{\forall f \in F_t} x^f_v + \sum_{\forall f^c \in F^c_t} x^{f^c}_v. \tag{12}$$

History h score is used to compare paths to flows in the previous control periods (F^T). h score is used to avoid selecting nodes that have already been selected several times. The selection effect of previous control periods decays over time; i.e., the effect of flows in control period $t - a\Delta t$ is higher than that in $t - b\Delta t$ when $a < b$. Thus, the h score is defined as follows:

$$h_v = \sum_{\tau=1}^{T} \left(\sum_{\forall f \in F_{t-\tau\Delta t}} x^f_v + \sum_{\forall f^c \in F^c_{t-\tau\Delta t}} x^{f^c}_v \right) * 2^{-\tau} \tag{13}$$

where $2^{-\tau}$ is the decay factor in which τ is equal to one for the preceding control period and is equal to T for the last stored control period. Using (12), (13) can be written as follows:

$$h_v = \sum_{\tau=1}^{T} s^{t-\tau\Delta t}_v * 2^{-\tau}. \tag{14}$$

Therefore, the obfuscation level of selecting a path is computed by combining the similarity and history scores of the intermediate nodes in that path. Hence, to maximize the obfuscation level in (5), a function of s and h is minimized, as follows:

$$\max_{\forall f \in F_t} \mathrm{Obf}(f) = \min_{\forall f \in F_t} \sum_{\forall v \in G.V} x^f_v(\alpha s_v + \beta h_v). \tag{15}$$

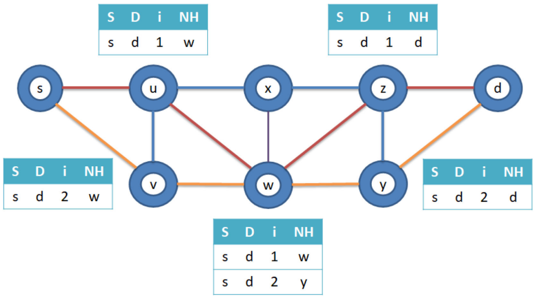
Then, based on (6) and (15), the objective function of RM in (5) is written as

$$\begin{aligned} \min_{\forall f \in F_f} \quad & \sum_{\forall u,v \in G.V} x^f_u x^f_v \mathrm{Cost}(e_{uv}) + \sum_{\forall v \in G.V} x^f_v(\alpha s_v + \beta h_v) \\ \text{s.t.} \quad & \sum_{\forall f \in F_t} x^f_v \leq C^{th}_v \\ & \sum_{\forall v \in G.V} x^f_v \leq L^{th} \quad \forall f \in F. \end{aligned} \tag{16}$$

The first constraint ensures that the selected node cannot be beyond the capacity limit, i.e., $C^{th}_u \geq \sum_{\forall f \in F_t} x^f_u$. The objective function is extended to consider the QoS constraints on routes. We assume that the QoS requirement of a mutated route is defined by the maximum allowed path length (L^{th}) of the route in terms of the numbers of hops (second constraint).

C. Multiple Mutated Routes

In this section, multiple paths are used for each flow to deceive the adversary. The controller assigns k_r paths with the objective function in (16). The generated k_r paths can be used in two ways. First, all the k_r paths must be used when sending a message. Thus, the source divides the message into k_r fragments, with each fragment sent individually through one of the k_r paths. Second, the source will use a round-robin approach to select a path from the assigned k_r paths to send each message in the flow. As shown in Fig. 3, two paths ($k_r = 2$) are set for the flow. Moreover, the flow table is extended to have an extra field called path ID (i), ranging from 1 to k_r as it indicates the mutated path ID. This ID will be appended to the message header. Assume that there are three possible paths (orange, red, and green) for a flow ($s \rightarrow d$) in which node s is the source and node d is the destination (as shown in Fig. 4). Assume that all these paths are equally ranked, and all nodes have the same h scores. Only two paths ($k_r = 2$) are needed for the flow. Minimizing the number of shared nodes (edges) in the k_r mutated paths is needed. If the similarity between each pair of paths is computed, then $s(\text{red, green}) = 2$, $s(\text{red, orange}) = 1$,



S	D	i	NH
s	d	1	w

S	D	i	NH
s	d	1	d

S	D	i	NH
s	d	2	w

S	D	i	NH
s	d	2	d

S	D	i	NH
s	d	1	w
s	d	2	y

Fig. 3. Example of multiple mutated routes.

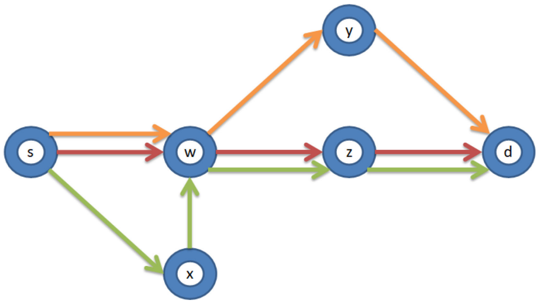


Fig. 4. Example for paths similarity and history.

and $s(\text{green}, \text{orange}) = 1$. Both pairs (red and orange) and (green and orange) have the least similarity score. However, the pair (red and orange) is the best choice in terms of the shortest path. As another example, assume that one path is needed for the flow ($s \rightarrow d$) (Fig. 4). If history h score of node indicates how frequent this node was selected in a prior paths, then assume $h(w) = 2$, $h(x) = 1$, $h(y) = 1$, and $h(z) = 5$. This indicates that node z was selected most often for previous flows and/or in the previous time windows. Thus, it is better to avoid selecting this node as it has a higher h score than the other possible nodes. Both red and green paths pass through the node z. Therefore, selecting the orange path is the best choice based on the history of the nodes.

D. Route Obfuscation Algorithm

The route obfuscation algorithm takes the initial and current energy level, reliability score, and history score of nodes as inputs. All of this information are obtained by the controller due to its supervisory view of the network. First, energy weight E_j, capacity score C_j^c, and reliability score Rel_j are computed for all nodes in the network (7), (9), and (10). Also, the edge cost e_{ij}^c for all edges in the network is computed based on (8). Thus, the cost of each link in the network is precomputed for all links.

In Algorithm 1, the detail of paths assignment of the RM is presented. A modified version of the Dijkstra algorithm is used to guarantee that the algorithm will select the optimal minimum cost route for each flow based on the path cost given in (6). The Dijkstra algorithm is a greedy heuristic algorithm that at every step the fittest option possible is chosen at that step without consideration to future consequences. The algorithm excludes nodes to be the next hop when their flow table limit is exceeded [$available\_adjacent()$]. The edge is valid as a possible candidate only if the combination of h_v and s_v is minimum. This step is defined as $\alpha s_v + \beta h_v < \Phi_{\max}$ where $\Phi_{\max}$ is initially very low and then increases when the number of generated mutated paths is less than k_r, or the path length exceeds L^{th}. After every failure, $\Phi_{\max}$ is increased by ϕ^+. Finally, s_j and h_j scores are updated for all nodes in the selected paths. The time complexity of Algorithm 1 is $O(\varphi k_r |E| \log |V|)$, where $|E| \log |V|$ is the running time of the Dijkstra algorithm. For the multiple mutated routes, it is multiplied by k_r as the algorithm runs the Dijkstra algorithm k_r times. φ is defined as the average number of failures to generate a valid path. φ is inversely proportional with the step size ϕ^+ due to the size of the selection pool. An SNcRM algorithm has a time complexity of $O(|V|^2 * 2^{|V|})$ [11] while SRO algorithm's time complexity is $O(k_r |E| \log |V|)$ [14].

Algorithm 1: Route Obfuscation Algorithm

```
Input  : $G(V, E)$, $Cost(Link)$, $h$, $s$
Output: $f$
while ($f.paths(k_r)! = valid$) do
    for $k = 1 \longrightarrow k_r$ do
        $Q.init(G.V)$;
        while not $Q.isEmpty()$ do
            $u \longleftarrow Q.extractMin()$;
            for each $v \in u.available\_adjacent()$ do
                if ($\alpha s_v + \beta h_v) < \Phi_{max}$ then
                    if $v.cost() > u.cost() + Cost(Link_{u,v})$
                    then
                        $v.cost \longleftarrow u.cost() + Cost(Link_{u,v})$;
                        $v.parent \longleftarrow u$;
                    end
                    $Q.modifyKey(v)$;
                end
            end
        end
    end
    $\Phi_{max} = \Phi_{max} + \phi^+$;
end
for each $v \in f.paths(k_r)$ do
    $update(v, s_v, h_v)$;
end
```

V. Sink Obfuscation

In fully centric WSNs, most of the data messages are delivered to the sink node to reach the application server. Moreover, in SDWSNs, the controller adds another level of centrality, as shown in Fig. 1. First, the sensed data are delivered to the application layer above the controller. Second, there is the network configuration exchange between the controller and the SDN-enabled sensors [31]. This produces a pronounced communication pattern that exposes the sink identification. The unique function of the sink node to connect the network and the controller makes it a single point of failure. Thus, an adversary that attempts to attack the network availability can reveal the sink node by applying traffic analysis techniques. The goal of sink node obfuscation is to minimize the traceability of a sink node by an adversary. To hide the sink node, misleading the adversary and covering the exclusive traffic pattern are needed, which usually exposes the sink node. To achieve this, k_s fake sink nodes are employed in the network that have a similar deceptive traffic pattern as the sink nodes. This solution is inspired by the k-anonymity model [25]. As k_s fake sink nodes increases, sink node discovery becomes more difficult. However, the deceptive traffic will result in extreme and redundant overhead. As a result, the problem is formulated to determine how these fake sink nodes are selected while simultaneously maximizing the obfuscation level and minimizing the cost

$$\max \mathrm{Obf}\left(\hat{\mathcal{S}}\right) \;\&\; \min \mathrm{Cost}\left(\hat{\mathcal{S}}\right) \tag{17}$$

where $\hat{\mathcal{S}}$ is the set of nodes that are selected to be fake sink nodes ($\hat{\mathcal{S}} = \{\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_{k_s} | \forall \hat{s} \in G.V\}$). $\mathrm{Cost}(\hat{\mathcal{S}})$ is the cost of selecting $\hat{\mathcal{S}}$, which is determined by maintaining the energy constraints of the selected nodes and generating deceptive traffic. The goal is creating more local maxima where the fake sink nodes act as traps for the heuristic adversary. The deceptive traffic is created when a real message is generated. In contrast to existing techniques, all messages are delivered to the nearest fake sink node only and *vice versa* to minimize the overhead. Furthermore, the sink node forwards the broadcast message to the fake sink nodes to reduce the overhead. When the fake sink node receives the broadcast message, it will broadcast the message within its cell.

A. Selection of Fake Sink Nodes

First, the network is divided into k_s nonoverlapping cells to minimize the traffic overhead. Then, one of the k_s fake sink nodes is selected from each cell. The deceptive traffic inside a cell is generated between the cell members and the chosen fake sink node to create local maxima. The formation of a cell can be achieved based on the number of nodes (node density), distances, and/or expected traffic. The controller sorts all nodes in the cell based on an energy-based fit score to select the fake sink node from a cell. Then, the $\hat{\mathcal{S}}$ set is formed of the top nodes in each cell. Initially, $\hat{\mathcal{S}}$ are chosen randomly under the security constraint as all the nodes have the same initial energy. However, after one round, the residual energy will be different. When the residual energy of the cell's fake sink node falls below a threshold th_E (computed based on the energy level of all nodes in the cell), the controller reselects the cell fake sink node and updates the flow rules. Thus, (17) is rewritten such that the cost of selecting the fake sink for each cell is minimized while the residual energy of the selected node is greater than th_E, as follows:

$$\begin{aligned} &\max \sum_{\forall \hat{s} \in \hat{\mathcal{S}}} \mathrm{Obf}(\hat{s}) \;\&\; \min \sum_{\forall \hat{s} \in \hat{\mathcal{S}}} \mathrm{Cost}(\hat{s}) \\ &\text{s.t. } \mathrm{Energy}_{\hat{s}} > th_E \quad \forall \hat{s} \in \hat{\mathcal{S}}. \end{aligned} \tag{18}$$

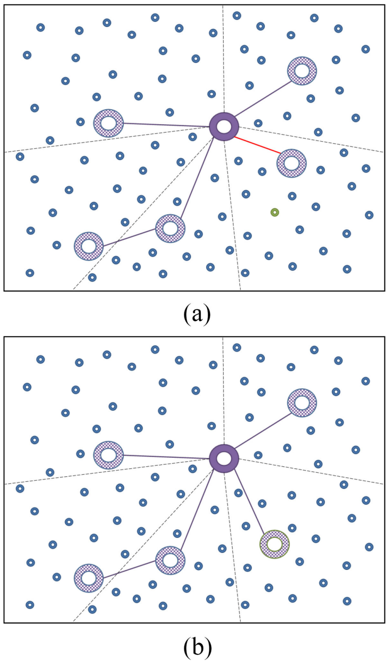
When a node is selected in $\hat{\mathcal{S}}$, both its own energy level and also the energy levels of its neighbors are crucial. All nodes in the cell deliver deceptive messages to this fake sink node. The candidate nodes are sorted based on a fit score (Fit_E) to minimize the effect of energy consumption due to the deceptive traffic inside the cell. The fit score considers the energy level of the node and the energy level of m-levels neighbor nodes in the cell. If the neighbor node has fewer hop connections to the candidate node, then there is a more significant effect on its fit score. The nodes closest to the fake sink node will consume more transmission energy due to deceptive traffic delivery. For example, if node v is a candidate node, node u is a one-hop neighbor to node v while node w connects to v in three hops. The energy level of these neighbor nodes u and w is considered when determining the fit score of node v. However, the influence of node u's energy level must be greater than that of node w due to the closer distance. Hence, 2^{-k} factor is associated with each neighbor nodes when computing Fit_E where k is the number of hops. Therefore, the fit score Fit_E of node v is calculated as follows:

$$\mathrm{Fit}_E(v) = \mathrm{Energy}_v + \sum_{k=1}^{m}\left(\frac{\sum_{i=1}^{|v.H(k)|} \mathrm{Energy}_{v.H(k)[i]}}{|v.H(k)|} * 2^{-k}\right) \tag{19}$$

where Fit_E is a score in the range of [0, 2]. v is any candidate node, and m is the highest number of hops for node v with the farthest node in the cell ($v.cell()$). $v.H(k)$ returns the set of neighbor nodes that have exactly k hops connection with node v where $v.H(k)[i]$ is the ith node in the set and $|v.H(k)|$ is the number of nodes in that set. The average energy level of each node in the k-levels is computed. The information needed to determine Fit_E score is easily obtained by the controller due to its supervisory view. Thus, the cost function of selecting a fake sink node in (18) is determined by Fit_E as follows:

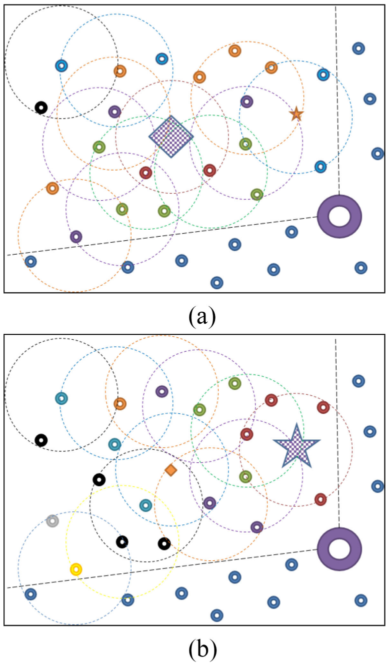
$$\begin{aligned} &\max \sum_{\forall \hat{s} \in \hat{\mathcal{S}}} Obf(\hat{s}) \;\&\; \sum_{\forall \hat{s} \in \hat{\mathcal{S}}} \mathrm{Fit}_E(\hat{s}) \\ &\text{s.t. } \mathrm{Energy}_{\hat{s}} > th_E \quad \forall \hat{s} \in \hat{\mathcal{S}} \\ &\quad * \mathrm{Fit}_E(\hat{s}) > \mathrm{Fit}_E(v) \quad \forall \hat{s} \in \hat{\mathcal{S}} \;\&\; \forall v \in \hat{s}.\mathrm{cell}(). \end{aligned} \tag{20}$$

The second constraint ensures that the selected fake sink nodes have the lowest fit score of their cells. Fig. 5 shows an example of $\hat{\mathcal{S}}$ selection procedure. First, the network is divided into five cells; thus, five fake sink nodes are selected ($k_s = 5$). Fig. 5(a) presents the initial set of $\hat{\mathcal{S}}$ where the selected nodes are the highest Fit_E nodes of their cells. Fig. 5(b) presents the final selected set of $\hat{\mathcal{S}}$. Fig. 6 shows the m-levels neighbor nodes of two nodes (Diamond $\diamond$ and Star $\star$) of the cell in the upper left corner of Fig. 5. For both cases, the red nodes are one-hop neighbors, the green nodes are two-hops neighbors, the purple nodes are three-hops neighbors, and so on. Assume that $\diamond.E$ is equal to $\star.E$, which is 0.8. For the diamond node, the average of energy level of the two one-hop neighbors is 0.6 and the average of five two-hops neighbors is 0.68. For the star node, the average of energy level of the four one-hop neighbors is 0.675 and the average of three two-hops neighbors is 0.666. If $m = 2$,


(a)

(b)

Fig. 5. Initial and final sets of fake sink nodes ($\hat{\mathcal{S}}$). (a) Initial set of fake sink Nodes ($\hat{\mathcal{S}}$). (b) Final set of fake sink nodes ($\hat{\mathcal{S}}$).


(a)

(b)

Fig. 6. m-Levels of connections when determining the Fit_E score of the star node. (a) Diamond node as a candidate node. (b) Star node as a candidate node.

then $\text{Fit}_E(\diamond) = 0.8 + 0.6 * 2^{-1} + 0.68 * 2^{-2} = 1.27$ and $\text{Fit}_E(\star) = 0.8+0.675*2^{-1}+0.666*2^{-2} = 1.304$. Thus, node $\star$ is fitter than node $\diamond$ to be a fake sink node even they have equal energy level. The effect of the cell's nodes differs for each case as well as Fit_E.

The goal is to maximize the number of steps for the adversary to locate the sink nodes. Therefore, the higher steps indicate a higher level of obfuscation. To determine the steps that adversary must endure moving from one trap (local maxima) to the next nearest trap, the travel cost (Ψ) of a minimum spanning tree (MST) of the real and fake sink nodes ($\mathcal{S}$ and $\hat{\mathcal{S}}$) is used. The travel cost $\Psi(\mathcal{S}, \hat{\mathcal{S}})$ is determined as the summation of distances of MST

$$\Psi\left(\mathcal{S}, \hat{\mathcal{S}}\right) = \sum_{\forall \bar{s}, \tilde{s} \in \left\{\mathcal{S}, \hat{\mathcal{S}}\right\}} d_{\bar{s}, \tilde{s}}. \tag{21}$$

The MST is rooted at the real sink node, where each edge in the MST determines the next nearest maxima from one sink to another (real or fake). Thus, (20) is rewritten by including the MST travel cost as follows:

$$\begin{aligned} \max \quad & \sum_{\forall \hat{s} \in \hat{\mathcal{S}}} \text{Fit}_E\left(\hat{s}\right) \\ \text{s.t.} \quad & \text{Energy}_{\hat{s}} > th_E \quad \forall \hat{s} \in \hat{\mathcal{S}} \\ & \text{Fit}_E\left(\hat{s}\right) > \text{Fit}_E(v) \quad \forall \hat{s} \in \hat{\mathcal{S}} \ \& \ \forall v \in \hat{s}.cell() \\ & \Psi\left(\mathcal{S}, \hat{\mathcal{S}}\right) \geq \Psi_{\min} \\ & d_{s,\hat{s}} < th_d \quad \forall \hat{s} \in \hat{\mathcal{S}} \ \& \ \forall s \in \mathcal{S} \end{aligned} \tag{22}$$

where $\Psi_{\min}$ is the minimum travel cost (third constraint). Furthermore, a minimum distance threshold th_d is defined such that the distance between the sink node and a fake sink node is greater than this threshold (fourth constraint). Fig. 5(a) shows the constructed MST of the initial set of $\hat{\mathcal{S}}$ with the real sink node. However, the initial set of $\hat{\mathcal{S}}$ does not meet the requirements of travel cost ($\Psi_{\min}$). Therefore, the one with minimum distance is removed and replaced with the following top node from the same cell. Then, the MST of the final set of $\hat{\mathcal{S}}$ is constructed as shown in Fig. 5(b).

B. Sink Obfuscation Algorithm

First, the network is divided into k_s cells. Then, the controller selects the top (Fit_E score) node in each cell as $\hat{\mathcal{S}}$ excluding nodes that have a distance to the real sink lesser than the minimum threshold th_d. If $\hat{\mathcal{S}}$ fails to meet the travel cost constraint, then the node in $\hat{\mathcal{S}}$ with minimum distance will be replaced with the following top candidate nodes in the same cell, and so on. In Algorithm 2, the overview of the selection of fake sink nodes algorithm is presented. To select $\hat{\mathcal{S}}$, the controller sorts the candidate nodes that meet the minimum distance threshold th_d in each cell by its current energy level. Then, $\hat{\mathcal{S}}$ are the top nodes in the k_s cells. Next, the controller constructs an MST from the $\hat{\mathcal{S}}$ set to determine the travel cost. After that, the travel cost of MST ($\Psi(\mathcal{S}, \hat{\mathcal{S}})$) is compared with the minimum travel cost ($\Psi_{\min}$). If $\Psi(\mathcal{S}, \hat{\mathcal{S}})$ is larger than $\Psi_{\min}$, then this initial set is accepted as the final set. If $\Psi(\mathcal{S}, \hat{\mathcal{S}})$ is smaller than $\Psi_{\min}$, then the node in MST that has the closer distance to the real sink(s) is replaced with the next candidate, and the MST step is repeated. The time complexity of Algorithm 2 is $O(|V|\eta^2 \log\eta)$ where η is the number of real and fake sink nodes. The first loop takes $O(|V|)$ as the algorithm sorts the cells' nodes based on Fit_E. Creating MST's time complexity is $O(\eta^2 \log\eta)$. The last loop can be repeated $|V|$ times as a worse case. Thus, this part's

Algorithm 2: Sink Obfuscation Algorithm

```
Input  : $G(V, E)$, $\mathcal{S}$, $Energy$, $cells$
Output: $\hat{\mathcal{S}}$
for each $c \in cells \subset G$ do
    $Fit_E \longleftarrow$ compute$Fit_E$(Energy); // Eq. 19
    if $(d_{v,s} > th_d)$ $\forall v \in c$ , $\forall s \in \mathcal{S}$ then
        $c.candidate\_list.append(v)$;
    end
    sort $\forall v \in c.candidate\_list$ based on $Fit_E(v)$;
    $\hat{\mathcal{S}}.append$(c.candidate_list.$extractMax$());
end
createMST($\mathcal{S}, \hat{\mathcal{S}}$);
while $(\Psi(\mathcal{S}, \hat{\mathcal{S}}) < \Psi_{min})$ do
    for each $\hat{s} \in \hat{\mathcal{S}}$ do
        $\hat{s}.sumDistance \longleftarrow \sum_{\forall s \in \mathcal{S}} d_{\hat{s},s}$;
    end
    $q \longleftarrow$ minNode(sumDistance);
    $\hat{\mathcal{S}}.remove(q)$;
    $\bar{c} = q.cell()$;
    $\hat{\mathcal{S}}.append(\bar{c}$.candidate_list.$extractMax$());
    createMST($\mathcal{S}, \hat{\mathcal{S}}$);
end
```

time complexity is $O(|V|\eta^2 \log \eta)$. PLAUDIT algorithm's time complexity is $O(|V||E|^2)$ [15].

VI. Performance Evaluation

A simulation model is used to evaluate the effectiveness of the proposed proactive defense against traffic analysis attacks, and several evaluation parameters are measured. Also, several routing mechanisms are simulated and compared. First, to obtain an estimate of a lower bound on the routing cost, SP routing is simulated, which selects the shortest path for each flow. The second routing mechanism is a ranking-based SP (rSP) scheme in which routes are selected based on the link cost weight in (11). The third routing mechanism is the RM scheme, in which routes are selected based on the shortest path while considering the similarity s and history h scores in Section IV-B. The fourth routing mechanism is the rRM scheme in which routes are selected based on the objective function in (16). The fifth routing mechanism is the Random ϕ^+ rRM (RrRM) scheme in which routes are selected based on the objective function in (16), besides, the increment in $\Phi_{\max}$ is a random variable with range $(0, \phi^+]$. Variations of multiple mutated routes for rRM and RrRM mechanisms are also simulated (krRM and kRrRM). The sixth routing mechanism is ksRM, in which k_s of fake sink nodes are used in the network using the objective function in (22). A combination of krRM and ksRM is also simulated (kskrRM). The last routing mechanism is the RW scheme, in which the next hop is randomly selected by assigning probability for each eligible link. Since RW routing generates random traffic, it provides an upper bound of the routing cost and the defense level. Moreover, the proposed NTO solution is compared with state-of-the-art solutions, such as MRRF [23], SNcRM [11], SRO [14], and PLAUDIT [15].

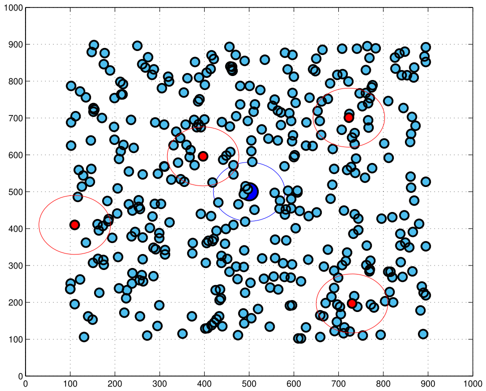


Fig. 7. Sample of simulation model.

A. Simulation Setup

The simulation is conducted using MATLAB. A network with 400 sensors is considered that are randomly deployed over an area of 800 × 800 with one sink node. The SDN controller is connected with the sink node using a secure wired network. All nodes have a CR of 80. The location of the sink node is at the center of the network (500, 500). The simulation proceeds in rounds, where various aspects related to flow routes are updated. The positions of nodes in a network would affect the experimental results. Thus, 1000 experiments are conducted while the positions of sensor nodes are randomly changed in each experiment. Then, the average results of over 1000 experiments for each topology are used. A sample of the network area with sensor nodes, sink node, and initial fake sink nodes' placement used in the simulations is shown in Fig. 7.

B. Attack Scenarios

In Section III-B, several attack types are defined. In this section, the attack scenarios used for evaluation are defined. The attack success rate defines the damage level of each attack scenario. In the first scenario, the adversary compromises nodes to sniff at the traffic that passes these nodes and around them. The success rate of the sniffer attack is determined by the degree of the node. The degree of a node is defined by the ratio of the number of routes passing the node from all possible routes. A maximum-likelihood estimation (MLE) method is used to compute the success rate. In the second scenario, the adversary launches a link-flooding attack on certain paths. The attack success rate is determined by how significant these links are. The weight of a link e_{ij} is the product of the betweenness centrality of node i and j. The betweenness centrality of a node is defined as the average of the total probabilities that routes passing through this node over all possible routes [32]

$$B_v = \frac{\sum_{\forall r \in \text{routes}} p_v^r}{0.5|V|(|V|-1)}. \tag{23}$$

In the third scenario, the adversary uses a compromised node v to learn about the shared links between data and control paths using the ARP of the CrossPath attack. The attack success rate is determined by the possible discovered shared links over the total number of control links in the network. In the last scenario, the heuristic attack is considered in which the adversary uses the traffic volume for the evaluation function. An attack failure rate is determined by the number of deadlock points the adversary may face when searching for the sink node. Also, the h-steps are defined as the number of steps needed to identify the sink node.

C. Evaluation Parameters

Several evaluation parameters are used to evaluate the proposed solution.

1) Energy Consumption: Energy consumption determines the average energy consumption of nodes.

2) Lifetime: The network's lifetime is determined by the time of the first node dies. In general, a longer lifetime implies that the communication traffic is more balanced among the nodes.

3) Path Length: Path Length is determined by the average number of hops for the generated routes.

4) Entropy: Entropy determines the randomness of network traffic (i.e., the distribution of traffic volume)

$$\text{Entropy} = -\sum_{\forall v \in G.V} \left(\frac{N_v}{N_{\text{total}}} \log_2 \frac{N_v}{N_{\text{total}}} \right). \tag{24}$$

In general, a higher value of entropy implies that the communication traffic pattern is more random.

5) Success Rate: Attack success rate is defined for each attack scenario in Section VI-B. A lower success rate implies a higher defense level.

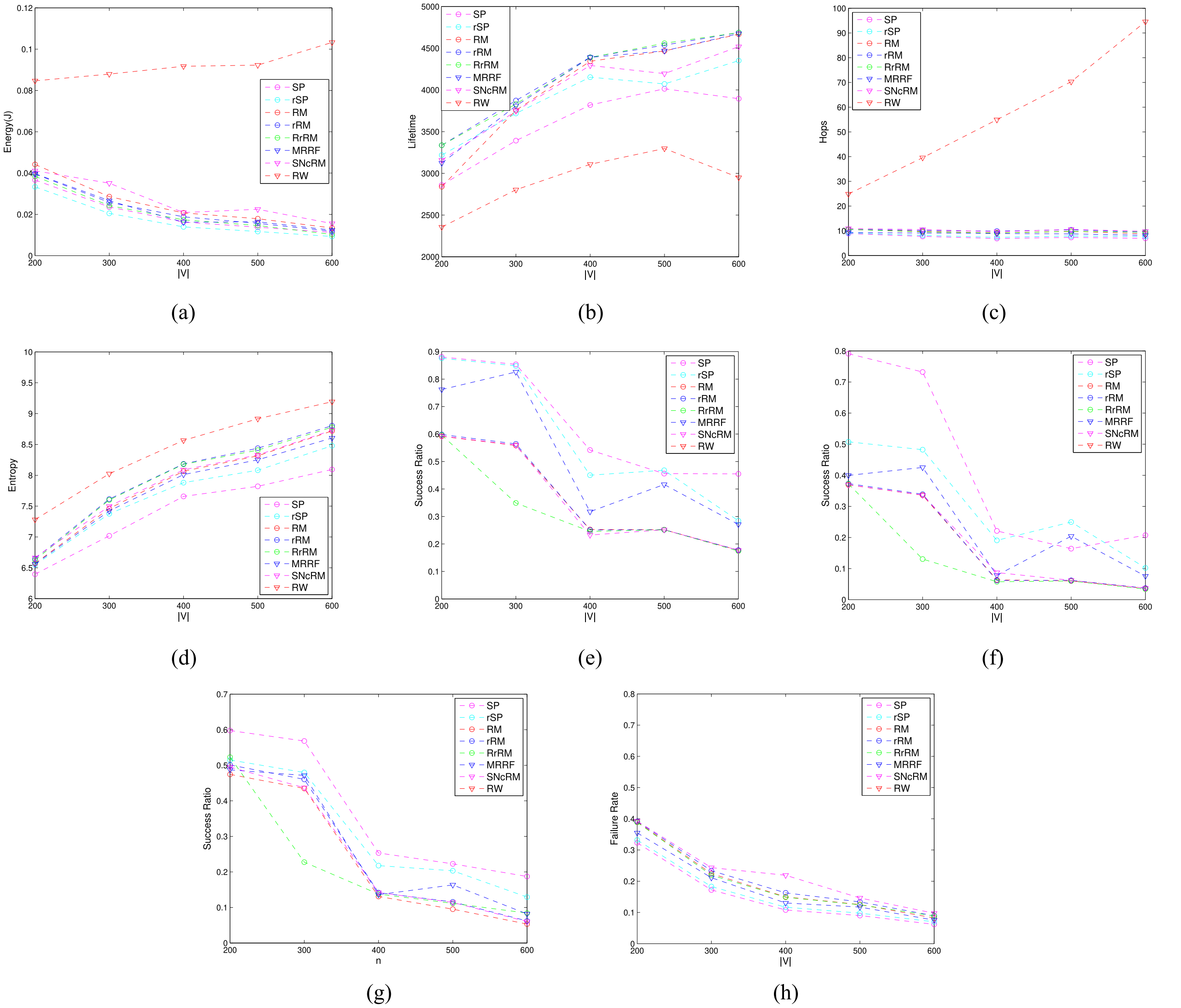
6) Failure Rate and h-Steps: Heuristic attack failure rate is defined by the deadlock rate described in Section VI-B. h-steps determine the average number of steps an adversary takes to identify the sink node using the heuristic attack.

D. Performance Analysis

1) Route Mutation: Fig. 8 shows the network and security performance of SP, rSP, RM, rRM, RrRM, MRRF [23], SNcRM [11], and RW mechanisms. In this figure, five different network sizes are considered where the number of nodes ($|V|$) is 200, 300, 400, 500, or 600. As expected, entropy is lowest for SP and highest for RW [Fig. 8(d)]. The entropy is lower with no mutated routes in SP and rSP because flows can pass a particular node repeatedly with no restriction. The entropy is higher in rRM and RrRM than in MRRF and SNcRM due to the security constraints of route selection in terms of similarity s and history h scores. More traffic distribution occurs in the ranking-based mechanisms (rRM and RrRM) than the RM. In Fig. 8(h), the average deadlock rate for heuristic attack correlates with the entropy values shown in Fig. 8(d). Higher entropy corresponds to a larger deadlock rate. This implies that entropy is a useful metric to measure the efficiency of the route obfuscation scheme. Without applying RM, the deadlock rate drops approximately 25% as the adversary can easily obtain the shared paths due to the pattern of communication traffic. Using the ranking approach decreases the success rate because the next hop can differ with time for the same flow. It is clear that SP and rSP show the worse defense performance against the sniffer, link-flooding, and CrossPath attacks. RM and rRM show a good defensive performance but lower than RrRM due to the additional randomness of choosing the next hop in terms of s and h scores. The success rate of the sniffer attack scenario in MRRF is higher than other routing mutation mechanisms. SP has the lowest obfuscation level because the node can be selected repeatedly as part of the routes. rRM and RrRM have a higher obfuscation level similar to SNcRM [Fig. 8(e)]. MRRF has no security guarantee in selecting the mutated routes; hence, it fails to protect the network. In Fig. 8(a), SP has worse energy consumption results than that of rSP. Likewise, RM and SNcRM have a worse energy consumption result than the other RM mechanisms. This is because they do not consider the link cost including the energy constraints. In Fig. 8(b), the rRMs mechanisms show a higher network lifetime due to the load balancing of the route assignment among nodes. Their network lifetime is even better than the energy-aware shortest path (rSP). SP has a lower network lifetime due to the repeated selection of specific nodes. MRRF shows a good lifetime result as the objective function is formulated to save the energy of the nodes. In Fig. 8(c), the rRMs and SNcRM mechanisms have a slightly higher path length. This is acceptable compared to the RW scheme. SP and rSP have the lowest path length as they are the shortest path mechanisms. The RW scheme has the highest defense performance in terms of sniffer, link flooding, and CrossPath attacks. However, Fig. 8 shows that RW has the poorest network performance. RW has a very high energy consumption, which results in the shortest network's lifetime. In addition, the average path length of the generated routes is five times (when $|V| = 400$) that of the next highest scheme.

2) Multiple Mutated Routes (k_r): Fig. 9 shows the influence of the number of mutated routes k_r for rRM, RrRM, and SRO [14]. The proposed route obfuscation techniques have a better network and defense performance than SRO. Fig. 9(d) shows the influence of k_r on the average entropy. As the entropy determines the randomness of network traffic, it slightly increases with the increase of k_r. In Fig. 9(h), as k_r increases, the number of heuristic attack's deadlocks increases due to the distribution of traffic. The success rate of the sniffer attack scenario is slightly decreased when the number of mutated routes is increased [Fig. 9(e)]. However, the success rate of other attack scenarios increases due to the increase in the number of overlapping points in data and/or control paths [Fig. 9(f) and (g)]. Fig. 9(a) shows that the energy consumption increases as the number of mutated routes (k_r) increases. This is an acceptable increase as the traffic volume is multiplied by the k_r factor. The network lifetime and path length are also linearly degraded [Fig. 9(b) and (c)]. RrRM has better network performance than rRM when $\phi^+ = 1$ [Fig. 9(a)–(c)]. The effect of ϕ^+ values is discussed in Section VI-D3.

3) Influence of ϕ^+: Fig. 9 also shows the influence of the ϕ^+ parameter. $\Phi_{\max}$ is a dynamic threshold that determines



(a)

(b)

(c)

(d)

(e)

(f)

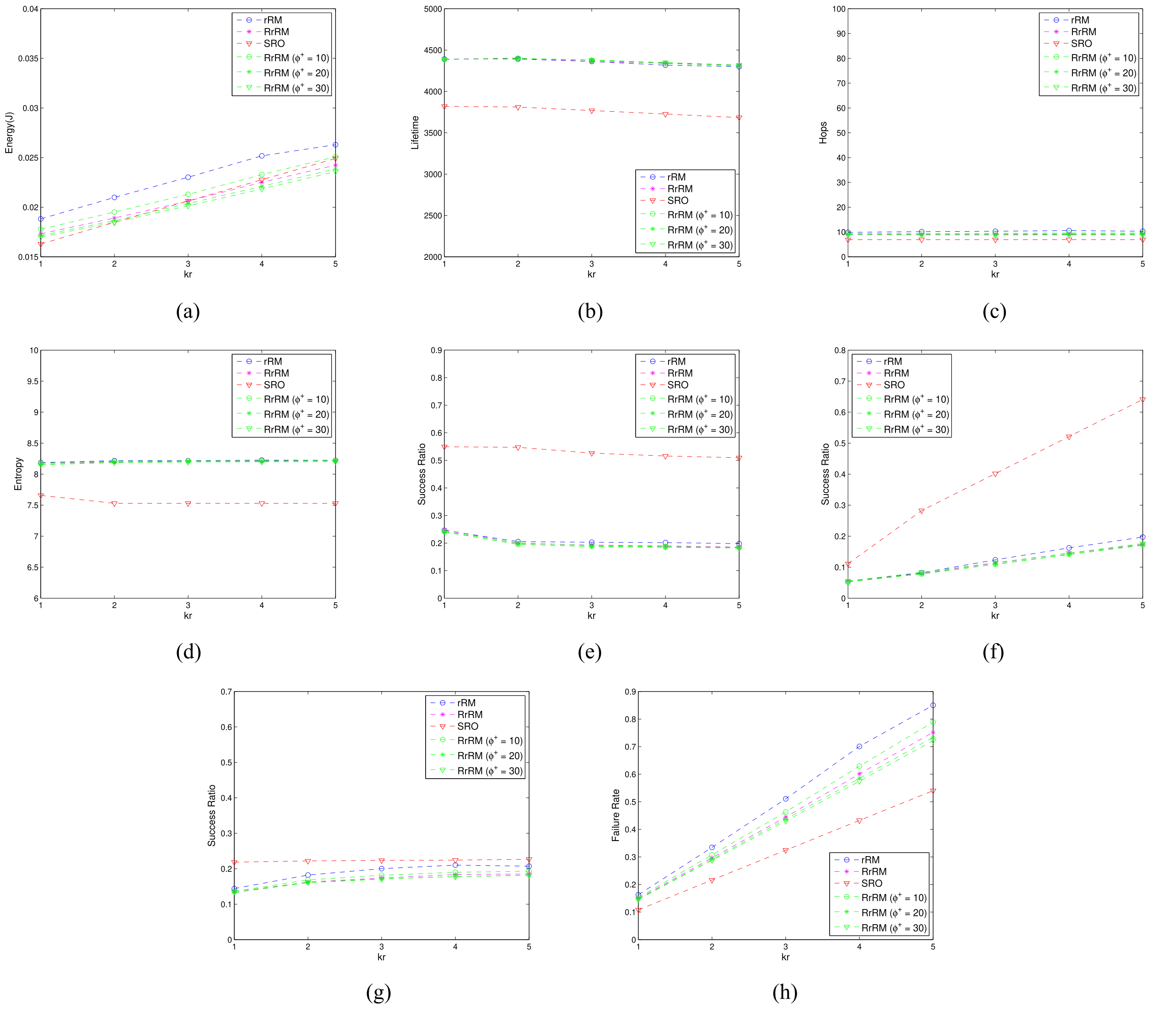
(g)

(h)

Fig. 8. Comparison between the proposed route obfuscation solutions and the state of the art. (a) Energy consumption. (b) Network lifetime. (c) Path length. (d) Entropy. (e) Sniffer attack. (f) Link-flooding attack. (g) Cross path attack. (h) Heuristic attack.

the optimal similarity s and history h scores. This parameter determines the increased steps of $\Phi_{\max}$ when constructed paths are invalid due to the s and h scores. ϕ^+ parameter determines how much the selection is strict to find the optimal path. The entropy slightly degrades when ϕ^+ increases [Fig. 9(d)]; hence, there is an increase in the success rate of the studied attacks. Higher ϕ^+ means expanding the candidate list of nodes to be selected for the next hop, hence, a higher consideration to the node/link cost. A lower ϕ^+ leads to fewer candidates due to the strict selection. Increasing the ϕ^+ parameter degrades the defense performance, but it improves the network performance. In the end, this research aims to balance network protection and functionality.

4) Sink Obfuscation: Fig. 10 provides a comparison between the proposed mechanisms (rRM, 4krRM, 4ksRM, and 4ks4krRM) and the state of the art (SRO [14] and PLAUDIT [15]) in fully centric WSNs. In this figure, all the network traffic (data and control flows) happens between the sink node and the sensor nodes. In Fig. 10(d), there is a greater increase of entropy with four fake sink nodes (4ksRM and 4ks4krRM). However, with four mutated routes, the entropy is the highest because the traffic is more evenly distributed around the real and fake sink nodes (4ks4krRM). This shows that the idea of generating multiple routes and multiple fake sink nodes in a controlled manner does aid in making the network traffic pattern more random. Fig. 10(h) shows the average heuristic attack steps to reach the sink node. Having four fake sink nodes dramatically increases the attack steps due to the local maxima. Moreover, integrating route obfuscation and sink obfuscation (4ks4krRM) results in a higher number of attack steps. In Fig. 10(a), having four fake sink nodes increases the energy consumption as extra traffic is generated. 4krRM and 4ksRM have smaller energy consumption than 4ks4krRM, while PLAUDIT uses a higher energy consumption due to the higher traffic volume. However, 4ks4krRM provides better performance in terms of entropy



(a) (b) (c)

(d) (e) (f)

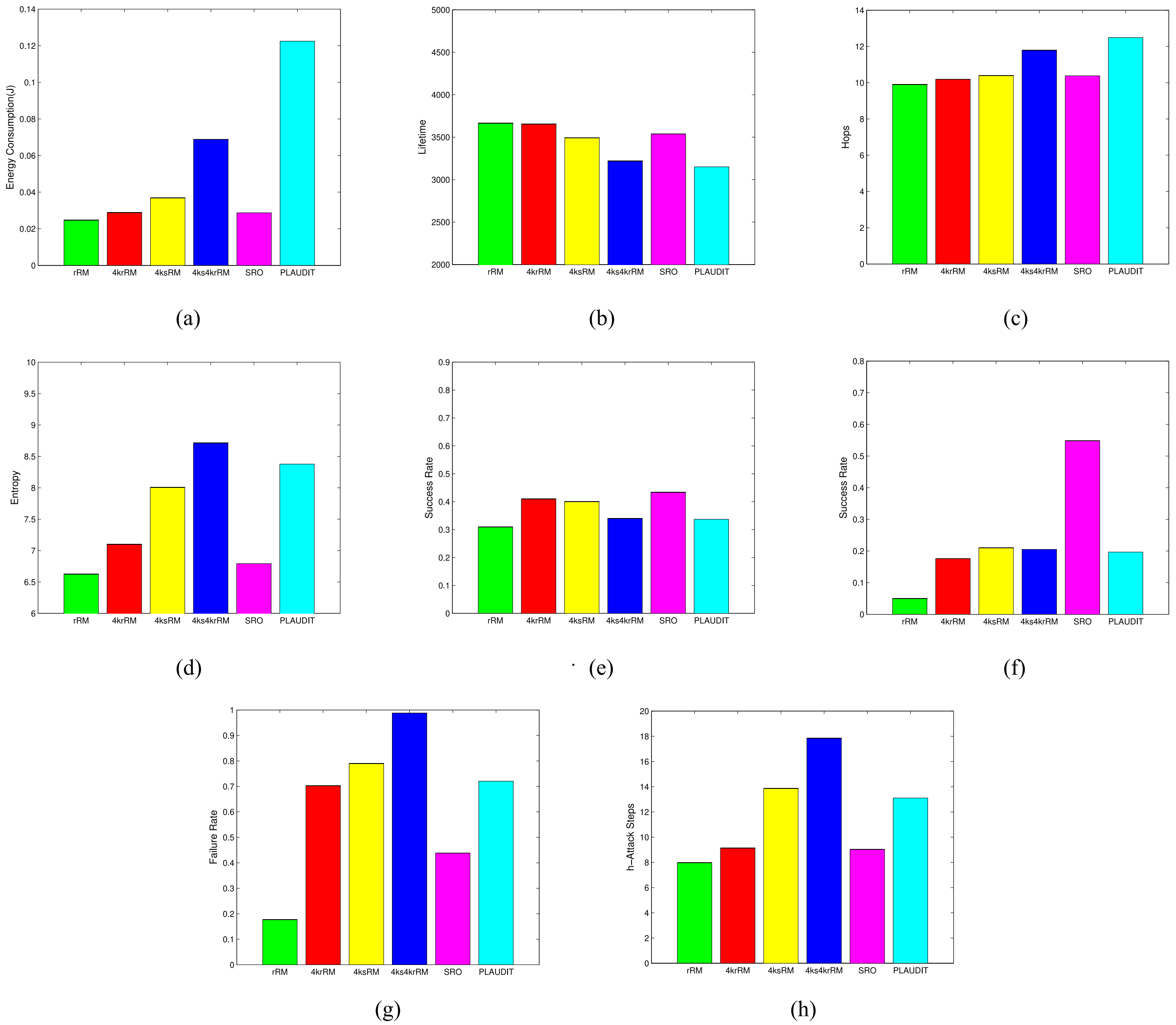
(g) (h)

Fig. 9. Influence of the number of mutated routes k_r. (a) Energy consumption. (b) Network lifetime. (c) Path length. (d) Entropy. (e) Sniffer attack. (f) Link-flooding attack. (g) CrossPath attack. (h) Heuristic attack.

and heuristic attack defense. Its effect on network parameters can be considered as the cost of better performance. Fig. 10 shows that SRO fails to defend the network that is fully centric WSNs. However, the deceptive traffic of PLAUDIT provides good defensive performance but not better than 4ks4krRM. Moreover, PLAUDIT has poor network performance, such as very high energy consumption and low network lifetime.

E. Discussion

Algorithm 1 solves a composite routing problem by finding paths for the given flow set using several network performance parameters and security metrics. In other words, a path is selected under similarity and history constraints with the least path cost that is computed by (6). The path cost is determined based on the residual energy of nodes and the expected energy consumption, as well as the node capacity and reliability. Moreover, the selected paths are restricted to a maximum path length to ensure QoS requirements. Consequently, the results of the proposed mechanism show better network performance, such as lower energy consumption, lower lifetime, and lesser path length associated with higher network protection. The security performance is investigated under several types of traffic analysis attacks, namely, node-based attack (sniffer), edge-based attack (link-flooding), and control attack (CrossPath). The proposed mechanism protects the network by hiding the network topology and obfuscating the traffic. The network traffic does not accumulate on the high-profile nodes as the network lessens relying on specific nodes without harming the network functionality. The heuristic attack is more complex; thus, a sink obfuscation algorithm is proposed. The results show that having multiple fake sink nodes degrades the network performance. Thus, Algorithm 2 balances the performance degradation and security gain by carefully selecting the fake sink nodes based on several criteria, including the distance between fake and real



(a) (b) (c)

(d) (e) (f)

(g) (h)

Fig. 10. Comparison between the proposed mechanism and the state of the art in fully centric WSNs. (a) Energy consumption. (b) Network lifetime. (c) Path length. (d) Entropy. (e) Sniffer attack. (f) Link-flooding attack. (g) Heuristic attack. (h) Heuristic attack steps.

sink nodes and the energy level of fake sink nodes and their neighbors.

VII. Conclusion

In this article, two mechanisms of NTO have been proposed to protect WSNs from traffic analysis attacks. In addition, they provide practical and scalable solutions for resource-constrained WSNs. First, an rRM mechanism that considers several route criteria to offer reliable and energy-efficient routing for route obfuscation has been developed. Second, sink node obfuscation can minimize the observability of a sink node by an adversary, especially for fully centric WSNs. For future work, advanced adversaries, such as global adversary and intelligent heuristic adversary, will be investigated. In addition, a learning-based RM approach that utilizes the historical topological data and the current network state will be developed.

Acknowledgment

The authors would like to thank the Hadhramout Foundation for their support during this research. The authors also would like to thank our colleagues from Broadband Communications Research (BBCR) Lab for their insightful comments.

References

[1] M. Charfi, A. Mouradian, and V. Vèque, “Networking functions for wireless sensor network applications: An SDN-based approach,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–7.

[2] S. Li, Q. Ni, Y. Sun, G. Min, and S. Al-Rubaye, “Energy-efficient resource allocation for industrial cyber-physical IoT systems in 5G era,” *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2618–2628, Jun. 2018.

[3] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, “Fragmentation-based distributed control system for software-defined wireless sensor networks,” *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 901–910, Feb. 2019.

[4] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K. R. Choo, “An energy-efficient SDN controller architecture for IoT networks with blockchain-based security,” *IEEE Trans. Services Comput.*, vol. 13, no. 4, pp. 625–638, Jul./Aug. 2020.

[5] D. Zeng, P. Li, S. Guo, T. Miyazaki, J. Hu, and Y. Xiang, "Energy minimization in multi-task software-defined sensor networks," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3128–3139, Nov. 2015.

[6] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks," in *Proc. Comput. Commun. IEEE Conf. (INFOCOM)*, 2015, pp. 513–521.

[7] C. Xenofontos, I. Zografopoulos, C. Konstantinou, A. Jolfaei, M. K. Khan, and K.-K. R. Choo, "Consumer, commercial, and industrial IoT (in)security: Attack taxonomy and case studies," *IEEE Internet Things J.*, vol. 19, no. 1, pp. 199–221, Jan. 2022.

[8] J. R. Ward and M. Younis, "Cross-layer traffic analysis countermeasures against adaptive attackers of wireless sensor networks," *Wireless Netw.*, vol. 25, no. 5, pp. 2869–2887, 2019.

[9] J. Jiang, G. Han, H. Wang, and M. Guizani, "A survey on location privacy protection in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 125, pp. 93–114, Jan. 2019.

[10] A. Liu, Z. Zheng, C. Zhang, Z. Chen, and X. Shen, "Secure and energy-efficient disjoint multipath routing for WSNs," *IEEE Trans. Veh. Technol.*, vol. 61, no. 7, pp. 3255–3265, Sep. 2012.

[11] Y. Zhou, W. Ni, K. Zheng, R. P. Liu, and Y. Yang, "Scalable node-centric route mutation for defense of large-scale software-defined networks," *Secur. Commun. Netw.*, vol. 2017, Dec. 2017, Art. no. 4651395, doi: 10.1155/2017/4651395.

[12] Q. Duan, E. Al-Shaer, and J. H. Jafarian, "Efficient random route mutation considering flow and network constraints," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, 2013, pp. 260–268.

[13] J. Wang, F. Wang, Z. Cao, F. Lin, and J. Wu, "Sink location privacy protection under direction attack in wireless sensor networks," *Wireless Netw.*, vol. 23, no. 2, pp. 579–591, 2017.

[14] A. Rauf, Z. Wang, H. Sajid, and M. A. Tahir, "Secure route-obfuscation mechanism with information-theoretic security for Internet of Things," *Sensors*, vol. 20, no. 15, p. 4221, 2020.

[15] N. Baroutis and M. Younis, "Load-conscious maximization of base-station location privacy in wireless sensor networks," *Comput. Netw.*, vol. 124, pp. 126–139, Sep. 2017.

[16] J. Cao *et al.*, "The crosspath attack: Disrupting the SDN control channel via shared links," in *Proc. 28th USENIX Secur. Symp.*, 2019, pp. 19–36.

[17] J. Deng, R. Han, and S. Mishra, "Countermeasures against traffic analysis attacks in wireless sensor networks," in *Proc. 1st Int. Conf. Secur. Privacy Emerg. Areas Commun. Netw. (SECURECOMM)*, 2005, pp. 113–126.

[18] S. Yoon, J.-H. Cho, D. S. Kim, T. J. Moore, F. Free-Nelson, and H. Lim, "Attack graph-based moving target defense in software-defined networks," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 3, pp. 1653–1668, Sep. 2020.

[19] J. Y. Koh, D. Leong, G. W. Peters, I. Nevat, and W.-C. Wong, "Optimal privacy-preserving probabilistic routing for wireless networks," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 9, pp. 2105–2114, Sep. 2017.

[20] J. Y. Koh, G. W. Peters, I. Nevat, and D. Leong, "Probabilistic routing in wireless networks with privacy guarantees," *Comput. Commun.*, vol. 151, no. 6, pp. 228–237, 2020.

[21] M. M. E. A. Mahmoud and X. Shen, "A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 10, pp. 1805–1818, Oct. 2012.

[22] B. D. Ying, D. Makrakis, and H. T. Mouftah, "Anti-traffic analysis attack for location privacy in WSNs," *EURASIP J. Wireless Commun. Netw.*, vol. 2014, no. 131, pp. 1–15, 2014.

[23] A. Liu, X. Liu, T. Wei, L. T. Yang, S. Rho, and A. Paul, "Distributed multi-representative re-fusion approach for heterogeneous sensing data collection," *ACM Trans. Embedded Comput. Syst.*, vol. 16, no. 3, pp. 1–25, 2017.

[24] M. M. E. A. Mahmoud, X. Lin, and X. Shen, "Secure and reliable routing protocols for heterogeneous multihop wireless networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 1140–1153, Apr. 2015.

[25] G. Chai, M. Xu, W. Xu, and Z. Lin, "Enhancing sink-location privacy in wireless sensor networks through k-anonymity," *Int. J. Distrib. Sensor Netw.*, vol. 8, no. 4, 2012, Art. no. 648058.

[26] R. E. Navas, F. Cuppens, N. B. Cuppens, L. Toutain, and G. Z. Papadopoulos, "MTD, where art thou? A systematic review of moving target defense techniques for IoT," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7818–7832, May 2021.

[27] Y. Wang, Q. Chen, J. Yi, and J. Guo, "U-TRI: Unlinkability through random identifier for SDN network," in *Proc. Workshop Moving Target Defense*, 2017, pp. 3–15.

[28] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 660–670, Oct. 2002.

[29] R. Sawilla and D. Skillicorn, "Partial cuts in attack graphs for cost effective network defence," in *Proc. IEEE Conf. Technol. Homeland Secur. (HST)*, 2012, pp. 291–297.

[30] M. Bin-Yahya, O. Alhussein, and X. Shen, "Securing software-defined WSNs Communication via Trust Management," *IEEE Internet Things J.*, early access, Aug. 5, 2021, doi: 10.1109/JIOT.2021.3102578.

[31] M. Conti, F. De Gaspari, and L. V. Mancini, "A novel stealthy attack to gather SDN configuration-information," *IEEE Trans. Emerg. Topics Comput.*, vol. 8, no. 2, pp. 328–340, Apr.-Jun. 2002.

[32] Q.-S. Hua, M. Ai, H. Jin, D. Yu, and X. Shi, "Distributively computing random walk betweenness centrality in linear time," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2017, pp. 764–774.

Manaf (Ben Yahya) Bin-Yahya received the B.Sc. degree in computer science and engineering from Aden University, Aden, Yemen, in 2010, and the M.A.Sc. degree in computer engineering from the King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, in 2016. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Waterloo, Waterloo, ON, Canada.

His current research interests lie in the area of network security, wireless sensors network, software-defined network, distributed systems, and digital forensics.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, social networks, 5G and beyond, and vehicular ad hoc and sensor networks.

Prof. Shen received the R. A. Fessenden Award in 2019 from IEEE, Canada, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015, and the Education Award in 2017 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006 and the Outstanding Performance Award five times from the University of Waterloo and the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada. He has served as the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, and the IEEE Globecom'07, the Symposia Chair for the IEEE ICC'10, the Tutorial Chair for the IEEE VTC'11 Spring, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He was an Editor-in-Chief of the IEEE INTERNET OF THINGS and the Vice President on Publications of the IEEE Communications Society. He is also a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Fellow, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.