Securing Software-Defined WSNs Communication via Trust Management

Manaf Bin-Yahya, Omar Alhussein, *Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Software-defined wireless sensor networks (SDWSNs) can be functionally affected by malicious sensor nodes that perform arbitrary actions, e.g., message dropping or flooding. The malicious nodes can degrade the availability of the network due to in-band communications and the inherent lack of secure channels in SDWSNs. In this article, we design a hierarchical trust management scheme for SDWSNs (namely, TSW) to detect potential threats inside SDWSNs while promoting node cooperation and supporting decision making in the forwarding process. TSW evaluates the trustworthiness of involved nodes and enables the detection of malicious behavior at various levels of the SDWSN architecture. We develop sensitive trust computational models to detect several malicious attacks. Furthermore, we propose separate trust scores and parameters for control and data traffic, respectively, to enhance the detection performance against attacks directed at the crucial traffic of the control plane. Furthermore, we develop an acknowledgment-based trust recording mechanism by exploiting some built-in SDN control messages. To ensure the resilience and honesty of the trust scores, a weighted averaging approach is adopted, and a reliability trust metric is defined. Through extensive analyses and numerical simulations, we demonstrate that TSW is efficient in detecting malicious nodes that launch several communications and trust management threats, such as black-hole, selective forwarding, denial of service, bad mouthing, and ON–OFF attacks.**

***Index Terms*—Internet of Things (IoT), security, software-defined wireless sensor network (SDWSN), trust management, wireless sensor network (WSN).**

I. Introduction

IN RECENT years, the Internet-of-Things (IoT) paradigm has gained considerable attention from academia and industry. Wireless sensor networks (WSNs) are considered a key enabler of the IoT paradigm [1]. In a WSN-enabled IoT environment, a large number of sensor devices are connected to the Internet. Therefore, conventional architecture and solutions for WSNs should be subjected to further research and introspection [2]. WSNs are dynamic networks that encounter several challenges due to potentially limited communication range and resources, e.g., network management, QoS routing, and security challenges [3], [4]. Software-defined networking (SDN) is gaining prominence as an alternative architecture to address the aforementioned challenges in WSNs for the era of IoT [5]. The SDN is a centralized architecture that separates the control plane from the data plane and introduces programmability to the network elements in the data plane [6]. Introducing SDN to WSNs provides several benefits, such as flexible and centralized network management, efficient routing, enhanced mobility and localization management, and improved security [7], [8]. For routing, SDN makes the major decisions through the controller in the control plane, thereby rendering a more efficient use of power resources.

Furthermore, SDN can provide security solutions (e.g., routing isolation and obfuscation) while handling the heterogeneity, scalability, and limited resources of WSN for IoT applications [9]. However, such a centralized architecture, broughtforth by SDN, comes with new challenges, which are mainly due to having nondedicated channels for the control plane. In software-defined WSNs (SDWSNs), there is no secure channel due to limited resources since the control plane uses in-band and multihop communication [10]. In contrast, in other scenarios, an SDN controller connects with devices through a secure channel, e.g., transport layer security (TLS)/secure sockets layer (SSL)-based communication [11]. The robustness of SDWSNs is heavily dependent on the control plane and the intermediate nodes between the controller and the rest of the network [12]. However, identifying the security issues of SDWSNs has received little attention. Analyzing the attacks and defense aspects in SDWSNs is highly essential at this stage. In particular, SDWSNs have some unique properties, which differ from typical distributed WSNs. These unique properties range from the central controller to the inherited SDN vulnerabilities.

Having insecure in-band and multihop communication between the controller and sensor nodes increases the security attack surface, thereby making the network vulnerable to several communication threats that affect the network availability [13]. Message dropping and flooding behaviors are examples of communication threats as illustrated in Fig. 1. Crypto-based authorization and authentication techniques on their own are insufficient in dealing with such threats [14].

Therefore, it is imperative to develop trust management among the participating nodes to provide a secure reflection of the network. Trust management aims to secure WSNs against attacks launched by insider malicious nodes. It establishes trust relationships among network nodes based on their

Manuscript received 17 January 2021; revised 7 July 2021; accepted 21 July 2021. Date of publication 5 August 2021; date of current version 7 November 2022. This work was supported by the Hadhramout Establishment for Human Development (HEHD). *(Corresponding author: Manaf Bin-Yahya.)*

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

Omar Alhussein was with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada. He is now with the Wireless Advanced System and Competency Centre, Huawei Technologies Canada Inc., Ottawa, ON K2K 3J1, Canada, (e-mail: omar.alhussein@huawei.com).

Digital Object Identifier 10.1109/JIOT.2021.3102578

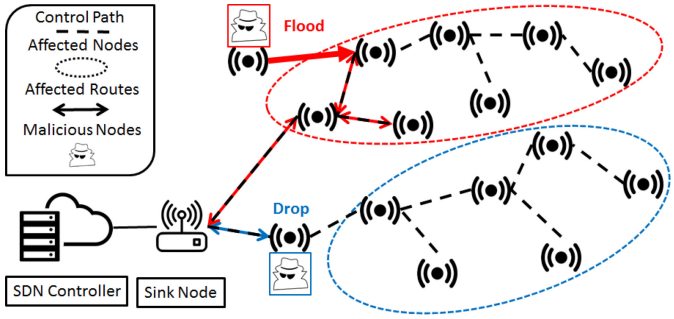


Fig. 1. Examples of SDWSNs communication threats.

experiences with others [15]. Besides defending against communication threats, trust management schemes need to be robust against good/bad mouthing and ON–OFF attacks. In this article, we consider that all nodes (including malicious ones) are authorized and authenticated, and we focus on trust management.

Extensive research efforts have been directed toward designing trust management schemes for conventional WSNs [16]–[21]. Recall that SDN enables network programmability via the controller node and provides a decoupled separation between control and data planes. Therefore, trust management schemes for conventional WSNs cannot be directly deployed for SDWSNs, and they do not harness of the introduced centralized and programmable capabilities. Moreover, due to the decoupled nature of SDWSNs (i.e., decoupled control and data planes), and due to the existence of a single point of failure, the effect of some threats can become aggravated, which calls for more careful consideration.

Only few works address trust-based routing and trust management for SDWSNs [22], [23]. An energy-efficient trust management and routing mechanism (ETMRM) was proposed in [23]. ETMRM utilizes the node's trust score and residual energy to detect malicious behavior and ensure secure yet energy-efficient routing. The authors provide a flow-table extension to achieve lightweight trust monitoring at the node level. However, the inherent *decoupled* and *hierarchical* architecture of SDWSNs is not fully considered. That is, the aforementioned trust management scheme does not consider a separation between the control and data planes, which is very crucial in such networks. Moreover, a trust management scheme that addresses the several hierarchical levels of SDWSNs is not yet studied. Moreover, we provide a thorough investigation and new careful design to trust evaluation models. See Section II-B for a more detailed discussion.

The goal of this article is to secure SDWSNs from nodes that behave maliciously to perform several attacks, e.g., black hole, selective forwarding, and DoS (new-flow [23]). While considering the particular characteristics of SDWSNs, we propose a trust management scheme for SDWSNs (namely, TSW scheme). We address key design issues and provide analysis of trust management for SDWSNs, including trust recording (i.e., how and what information to record about the nodes), trust evaluation (i.e., how to evaluate the trust score of each node), and trust propagation (i.e., how trust scores are aggregated at each architectural level). We want to point out that a preliminary version of this article appeared in [24]. Specifically, the main contributions of this research are summarized as follows.

1) We propose a hierarchical trust management scheme in which trustworthiness is evaluated at each level of the SDWSN architecture. This enables the architecture to have finer detection granularity and early response to malicious behavior. Furthermore, to avoid biased trust scores, trust metrics are computed from reliable scores based on a weighted averaging approach with a defined reliability trust metric.
2) To address the decoupled nature of SDWSNs, we separate the computation of trust scores to control plane and data plane traffic, respectively. The separation can provide more critical treatment to the control plane. Also, we utilize specific SDN-based control message types to design an acknowledgment-based trust recording mechanism, resulting in an enhanced detection rate of control dropping attacks.
3) We design trust evaluation models that have more sensitivity to the change of abnormal behavior for an enhanced detection time and accuracy. Moreover, we propose a new Bayesian method with reward and penalty factors (PFs) to increase sensitivity to bad behavior. Furthermore, we design a trust updating mechanism that can dynamically learn from past trust evaluation by giving past bad behavior more weight in the next time-window evaluation.
4) Finally, through both theoretical analysis and extensive numerical simulations, we show that our proposed scheme provides robust and efficient trust management.

The remainder of this article is organized as follows. A review of related works is discussed in Section II. In Section III, we introduce the system and communication threats models and define the design goals. Then, we provide an overview of the TSW scheme in Section IV. The details of the three phases of the TSW scheme are provided in Section V. Next, we provide theoretical analysis and performance evaluation for the TSW scheme in Sections VI and VII. Finally, we conclude our article in Section VIII.

II. Related Works

A. WSN and Trust Management

Securing WSNs through reputation-based schemes is widely proposed in the literature and has proven to be an effective approach for guarding the network against several attacks [3], [14], [15], [25]. Many research studies are proposed to build trust-based schemes to secure WSNs. We discuss some of the recently proposed schemes as follows. Jiang *et al.* proposed a distributed trust model scheme [16]. In this scheme, communication trust is determined through direct (subjective logic framework) and indirect (trust chain) approaches. Data trust is calculated based on the deviation between the received sensing data from a certain node from the average of other nodes' data in the same area. Energy

trust is calculated based on the residual energy of the node under the assumption that a node knows its neighbors' initial energy. Trust reliability and familiarity are used to ensure the precision of recommendation trust. Meng [17] proposed a trust-based scheme with traffic sampling implemented with the intrusion detection system (IDS) for IoT-driven WSN. A fixed or random sampling approach is used while considering the computational and energy capability of a limited resources network with a high data traffic rate. To detect malicious behaviors inside the network, a Bayesian-based intrusion detection method is used in which all data flows are expected to be independent. Zhang *et al.* [18] proposed a trust evaluation approach for clustered WSNs based on the cloud model. The model takes into consideration multiple factors. The approach can be used to meet security requirements according to WSN applications. However, this approach is constrained by the limited power resources of WSNs with lower computational power and incompatible recording approaches. Chen *et al.* [19] proposed a trust management scheme for IoT networks. The scheme provides adaptive management that is capable of evaluating relationships among IoT node owners to better assess social trust for those devices. Three social trust metrics are presented, namely, a cooperativeness metric based on social ties, an honesty metric based on node reliability, and a community-interest metric to assess co-relationships. Li *et al.* [20] proposed a lightweight and dependable trust management system (LDTS). In this hierarchical scheme, there are different levels of trust evaluation. The trust scores are calculated based on self-evaluation and cluster head (CH) recommendation at the cluster member level. Moreover, each CH evaluates other CHs while introducing the base station as a source of indirect trust evaluation. Sahoo *et al.* [21] proposed a bio-inspired trust management scheme to provide a secure clustering approach for WSNs. The trust scheme uses a honey bee mating algorithm for trust-based clustering. This approach avoids selecting a malicious node as a CH by creating a list of the nominee nodes associated with trust scores. The algorithm selects the appropriate CH according to this list.

Distributed non-SDN-based trust management solutions do not conflict with SDWSNs. However, such approaches do not naturally consider the decoupled nature of the network, nor can they harness the introduced centralized and programmable capabilities. For example, we develop an acknowledgment-based trust recording mechanism that is compatible with SDN-based control message types.

B. SDWSN and Trust Management

The research on trust management schemes for SDWSNs is still in a nascent stage [22], [23]. Vishnu and Manjunath [22] proposed a trust-based and QoS-aware routing mechanism for SDWSNs (namely, SeC-SDWSN). The authors define a three-Řtier architecture of clusters, switches, and SDN controllers. Sensors are clustered using a secure hash tree-based clustering algorithm. Then, an encryption approach is proposed for data security. After that, a fuzzy weighted technique is used to transmit data to the appropriate switch.

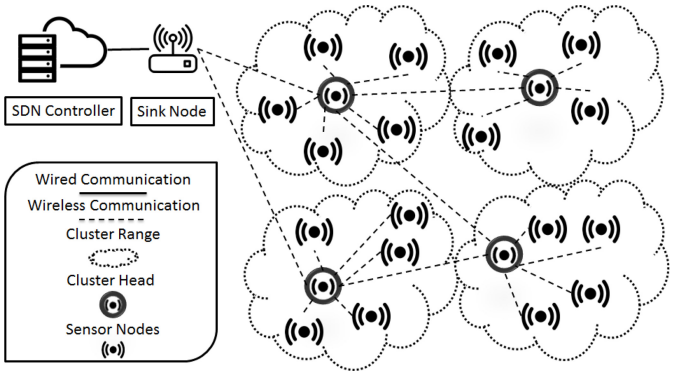


Fig. 2. SDWSN system model.

A more relevant work was proposed by Wang *et al.* where ETMRM considers both the node's trust score and residual energy to detect malicious behavior and ensure secure routing [23]. The authors provide a flow table extension to achieve lightweight trust monitoring at the node level. A Bayesian model is used as a trust computational model. In addition, trust scores are collected from sensor nodes at the controller level to detect and isolate malicious nodes in a centralized manner. ETMRM's trust scheme does not consider separating data and control traffic evaluation, which is an important inherent property when considering SDN-based networks. Moreover, a hierarchical scheme that addresses the inherent architectural levels in SDWSNs is not considered.

To this end, we consider and address several research gaps and design factors pertinent to the architecture and properties of SDWSNs, which have not been addressed yet. We design a hierarchical trust management scheme on the node, CH, and controller levels. Moreover, we design a trust management scheme that treats the control and data planes separately, which by design can provide higher sensitivity and protection to the crucial traffic of the control channels. Furthermore, we design trust scores that account and preserve historical behavior in the trust updating process. Also, we improve on the existing trust computational models by introducing a new Bayesian factor model with penalty and reward factors to increase the sensitivity to bad behavior, thereby improving the detection rate. We also make sure our trust management scheme is robust to bad/good mouthing (GM) and ON–OFF behavior.

III. Problem Formulation

In this section, we present the SDWSN system model as well as network and security assumptions. Then, we describe several potential threats in SDWSNs. Finally, we discuss the design goals of the TSW scheme.

A. System Model

The system model consists of several components, namely, sensor nodes, CHs, sink nodes, and the SDN controller, as shown in Fig. 2. Three hierarchical levels exist in this model, namely, the node level, CH level, and controller level. We assume that network nodes are deployed randomly and are

homogeneous (i.e., every sensor node has the same communication range). Also, each node in the network has a fixed position. Each sensor node must have at least two one-hop neighbors in its communication range. Thus, the network can be very dense. Nodes in the same communication range can detect, communicate, and overhear each other. For each cluster, the CH aggregates and relays messages inside the network. The controller has a global view of the network. It has a high computational capacity and unlimited communication resources compared to other components in the system. The controller receives statistical updates from the underlying network components to build comprehensive network maps. The controller can provide several services efficiently, such as routing, network management, and security based on these maps. Each node has a flow table, and the SDN controller is responsible for updating the flow rules. The flow table consists of matching rules, actions, and statistics fields.

B. Security Assumptions

We assume that each node can be identified by a unique legal ID. The network applies defense mechanisms to deal with replay, Sybil, and identity-based attacks [26]. Every node (including malicious nodes) is authorized and authenticated. The communication inside the SDWSN is encrypted using a shared key. Thus, unauthorized nodes cannot eavesdrop nor be an active part of the network. Also, every network node has another shared key with the SDN controller. Thus, no one is able to modify the message content between a component and the controller, such as to forge statistical updates, flow rule control messages, or trust update messages. Moreover, we assume that sink nodes and the controller are fully trusted. Sensor nodes and CHs can be malicious. We assume that there is no attack at the initial network setup.

C. Threat Models

SDWSNs face several security challenges and vulnerabilities due to their open environment and limited resources [7]. Thus, sensor nodes can be compromised physically or remotely and become malicious by an outsider attacker. Also, some nodes, such as selfish nodes, can act maliciously to preserve their limited resources, such as energy.

1) Black-Hole Attack: A malicious node drops all received messages received from its neighbors.

2) Selective Forwarding Attack: A malicious node drops received messages partially either at random or deliberately. A malicious node can launch this attack to block certain types of messages or certain node's messages, or degrade the network's message delivery ratio. On the other hand, if a selfish node launches the attack, dropped messages can be random.

Fig. 1 demonstrates an aggravated effect of dropping messages in SDWSNs. The bottom (blue) attacker drops the received control messages from/to the bottom part of the network, isolating it from connecting with the controller. In addition, both black-hole and selective forwarding (SF) attacks can be launched on the data plane.

3) DoS and New-Flow Attacks: The new-flow attack is a flooding DoS attack that targets the control plane of SDN-based networks. However, in SDWSNs, this attack can target both the control and data plane due to the large number of packet-in messages, which can degrade the network's availability. For example, in Fig. 1, the upper (red) malicious node floods a node in the control plane with packet-in messages. The control path starting from this node will be degraded in terms of bandwidth, overhead, and nodes' energy. As a result, the availability of the upper (red) part of the network will be affected. This attack can also be directed at the availability of the controller.

4) ON–OFF Attack: A malicious node launches the communication attacks irregularly. In so doing, a malicious node attempts to deceive trustworthiness metrics by behaving normally following a bad behavior.

5) Good/Bad Mouthing Attack: A malicious node attempts to falsify the aggregated trust for a particular node by providing biased scores. A malicious node sends a low trust score for a nonmalicious node in a bad mouthing attack, which would affect the nonmalicious node's trustworthiness at higher levels in the SDWSN. On the other hand, in a GM attack, malicious nodes try to raise the trust of other malicious nodes by sending good feedback. Potentially, a number of malicious nodes can also launch a collaborative mouthing attack.

D. Design Goals

In TSW, we aim to provide a security mechanism to defend against the aforementioned attacks. The main design goals of this work are listed as follows.

Simplicity and Efficiency: The trust scheme must have lightweight operations due to the limited resources of sensor devices. At the same time, the trust scheme needs to accurately detect and isolate malicious nodes from the network services. It can be challenging to develop lightweight models that are sufficiently sensitive to malicious behavior. Trust computational models are to ensure that the trustworthiness of a node must fall quickly following bad behavior yet rise slowly following good behavior.

Compatibility With SDWSNs: There are three hierarchical levels in SDWSN, thereby the need for a hierarchical trust management scheme, and there should be separate data and control trust scores. Besides, it is instructive to utilize built-in SDN messages for the recording phase, while statistical information can be collected from existing flow-table constructs.

Dynamicity and Timeliness: The computed trust score for a particular node at each level must be dynamic. Moreover, node reliability must be reflected correctly in the change of its trust score over time. Furthermore, the past behavior of a node needs to be considered when continuously updating its respective trust score.

Resilience and Honesty: Unreliable scores need to be avoided from the trust evaluation process in higher levels to reduce the effect of biased scores.

IV. TSW Scheme Overview

Fig. 3 presents an overview of the TSW scheme, which consists of three main phases, namely, recording, evaluation, and

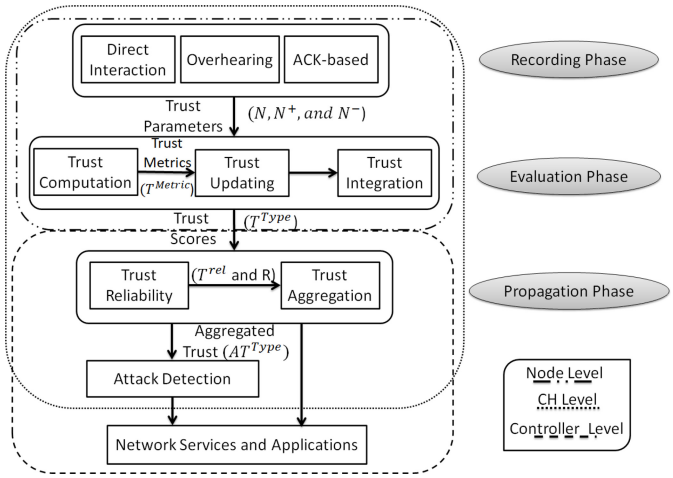


Fig. 3. TSW scheme overview.

TABLE I
TABLE OF NOTATIONS

$T_{x,y}$	Trust score computed by node x for node y.
T^{Metric}	Trust metric defined in section IV-A.
T^{Type}	Trust score computed through integration (V-B4).
N_y	Number of messages initiated by node y.
N_y^+	Number of successful messages forwarded by node y.
N_y^-	Number of unsuccessful messages forwarded by node y
th_H	Trust score threshold of being trusted ($T \geq th_H$).
th_L	Trust score threshold of being untrusted ($T \leq th_L$).
η_{max}	Maximum number of messages that a node can initiate.
η_{min}	Minimum number of messages that a node can initiate.
AT_{avg}	Average of received trust scores.
AT	Aggregated trust score.
R	Score reliability.
T^{rel}	Node reliability.

propagation. Trust parameters (in Section V-A) are collected and recorded in the recording phase through direct interaction, overhearing, and ACK-based approaches. Next, based on these parameters, trust metrics are computed through trust computational models (in Section V-B). The trust scores are updated with the consideration of past scores (in Section V-B3), and integrated together throughout every time window (Δt) (in Section V-B4). After that, the computed trust scores are aggregated at the upper layers of the SDN architecture (in Section V-C2), where the trust reliability is provided, and biased scores are excluded in the computation of the aggregated trust score (in Section V-C1). A list of notations is given in Table I.

A. Trust Metrics

In the TSW scheme, we define six trust metrics to evaluate the trustworthiness of nodes. These metrics are computed based on the sending and forwarding parameters from the recording phase Section V-A. The trustworthiness evaluation is conducted to determine the participation and cooperation of every node in the forwarding process. These metrics can differ based on the application of trust management [14]. Our scheme aims to secure network communications. We assume that node x is the evaluating node and node y is the evaluated node.

1) *Forwarding Trust:* Denote the forwarding trust metric for data traffic and control traffic by $T_{x,y}^{DF}$ and $T_{x,y}^{CF}$, respectively. These trust metrics evaluate how a node cooperates in the forwarding process of data and control messages, respectively. Such metrics are needed to detect message dropping attacks. They are computed by considering successful and unsuccessful experiences of participation through forwarding counters (Section V-A1).
2) *Sending-Rate Trust:* Denote the sending-rate trust metric for data traffic and control traffic by $T_{x,y}^{DS}$ and $T_{x,y}^{CS}$, respectively. These trust metrics evaluate whether or not the sending rate of data and control messages of a node is within the limit, respectively. Such metrics are needed to detect flooding attacks. If the number of initiated messages exceeds the maximum limit, the trust score must decrease to penalize node y. However, $T_{x,y}^{CS}$ also has a lower limit as the number of expected control messages can be determined (Section V-B2). $T_{x,y}^{DS}$ and $T_{x,y}^{CS}$ are computed based on sending rate parameters (Section V-A2).
3) *New-Flow Trust:* This trust metric $T_{x,y}^{NF}$ evaluates if the sending rate of packet-in control messages from a node is under the maximum limit. The packet-in messages are crucial because they can be used to launch a DoS attack in the SDN-based architecture. If the number of initiated messages exceeds the limit, the trust score must decrease. Similar to $T_{x,y}^{CS}$ and $T_{x,y}^{DS}$, the new-flow trust metric can computed through direct interaction and overhearing approaches.
4) *Node Reliability:* We need to determine a node trustworthiness when it sends or propagates trust scores to the upper layer. That is, we need a metric to defend against mouthing attacks, which can be computed by CHs and the controller. The node reliability metric T_y^{rel} is determined based on the aggregated trust scores from node y about its neighbors. A node becomes trusted when it provides reliable trust scores (Section V-C1).

B. TSW Architectural Levels

Next, we describe the trust management process at every SDWSN architectural level, namely, node, CH, and controller levels.

1) Node Level: Algorithm 1 illustrates the trust management process at the node level. Each node builds a record for its neighbors based on the recording approaches. The recorded parameters (counters) are used in the computation of trust metrics. These trust metrics are combined to build specific direct trust scores. Thus, for each time window Δt, nodes store the calculated trust scores for future use in the trust updating process. As well, these scores are sent to the upper level. According to [16], Δt should not be very small because this would mean frequent trust computation and updating, which would lead to power consumption. Furthermore, the trust evaluation will be affected by nonmalicious causes, such as congestion and delay. On the other hand, the trust score should not be computed over a very long period. As a result, the trust evaluation would not reflect the most current state of the nodes' trustworthiness.

Algorithm 1 Trust Evaluation Algorithm at Node Level

while t **during** Δt **do**
  $\forall E \in$ Events ▷ E is any interaction event experienced by the current node x.
  if E triggered for node y **then** ▷ y can be any node in the same range.
    $N^{E}_{x,y} = N^{E}_{x,y} + 1$;
    if E is positive **then**
      $N^{E+}_{x,y} = N^{E+}_{x,y} + 1$;
    else
      $N^{E-}_{x,y} = N^{E-}_{x,y} + 1$;
    end if
  end if
end while
at the end of Δt:
  for all $y \in Y$
    $T^{m}_{x,y}$ = **metric_model**$(N^{E}_{x,y}[, N^{E+}_{x,y}, N^{E-}_{x,y}])$; $\forall m \in$ *Metrics*
    $T^{m}_{x,y}$ = **update**$(T^{m}_{x,y}(t), T^{m}_{x,y}(t - \Delta t))$; ▷ Eq. 3
    $T^{ty}_{x,y} = \sum_m w_m T^{m}_{x,y}$; $\forall ty \in Types$
  end for
  send$(T^{ty}_{x,Y}, CH)$; ▷ CH is the CH of nodes' cluster.

Algorithm 2 Trust Evaluation Algorithm at CH Level

node_level_function();
for all $x \in C$ (Cluster)
  $T^{ty}_{x,Y}$ = **receive**(x); $\forall ty \in Types$
end for
$[R_{X,Y}, T^{\text{rel}}_{X}]$ = **reliability**$(T^{ty}_{X,Y})$; ▷ Eq.s 5 and 6
$AT^{ty}_{C,Y}$ = **aggregate**$(T^{ty}_{X,Y}, R_{X,Y})$; ▷ Eq. 8
forward$(T^{ty}_{X,Y}, controller)$;
if $AT^{ty}_{C,y} \leq th_L$ **then**
  report$(AT^{ty}_{C,y}, controller)$;
end if

2) CH Level: Algorithm 2 illustrates the trust management process at the CH level. At this level, CHs evaluate the trust of nodes in two directions. The first direction is the trust evaluation of their neighbors in the same communication range (node level). Additionally, every CH has to record and evaluate the trustworthiness of other CHs and report it to the controller. The second direction is that CH aggregates the trust update messages from cluster nodes and computes cluster-based trust aggregation and trust reliability of cluster nodes. These computed trust scores are sent to the controller. CHs can provide a quick response when malicious nodes are detected via aggregated trust score $AT^{ty}_{C,Y}$. Here, the CH can stop forwarding packets from/to the malicious node and report such events to the controller.

3) Controller Level: Algorithm 3 illustrates the trust management process at the SDN controller level. The controller, which has a supervisory view of the network, computes each sensor's overall trust scores. The trust scores, which are computed at the node level, are received by the controller. Based on these trust scores, the controller analyzes and combines

Algorithm 3 Trust Evaluation Algorithm at Controller Level

for all $x \in All$ nodes of the network
  $T^{ty}_{x,Y}$ = **receive**(x); $\forall ty \in Types$
end for
for all $c \in Cs(Clusters)$
  $T^{ty}_{c,Y}$ = **receive**(c);
end for
$[R_{X,Y}, T^{\text{rel}}_{X}]$ = **reliability**$(T^{ty}_{X,Y})$; ▷ Eq.s 5 and 6
$AT^{ty}_{All,Y}$ = **aggregate**$(T^{ty}_{X,Y}, R_{X,Y})$; ▷ Eq. 9
network_services$(AT^{ty}_{All,Y})$;

these scores for each node. To this end, the controller computes the global trust score for each node from the collected trust scores. Again, an outlier detection mechanism is used to ensure that the global aggregated trust is not affected by the mouthing attacks. The trust matrices at the controller have a broad sight because it includes the trust scores evaluated for nodes that are not from the same cluster. As a result, the controller determines the trust level of every node and decides which nodes can be trusted to provide the different network services. At the same time, the untrusted nodes are prevented from participating in the network operations. Therefore, the flow table updates depend on the collected trust scores.

V. TSW Scheme Phases

In this section, we provide details about the three phases of the TSW scheme. First, we describe the trust recording approaches and the recorded trust parameters. Next, we define the trust computational models for trust evaluation in addition to the trust updating and integration mechanisms. Finally, we give details about the propagation phase where the trust aggregation takes place at the CH and controller levels while considering the reliability of these aggregated trust scores.

A. Trust Recording Phase

Each node monitors and records other nodes' behavior in the same communication range in the recording phase. Thus, the recording process of trust information is accomplished by overhearing neighboring nodes during the message interactions and transmissions. The recorded information is learned from the interactions between the evaluated node and the evaluating node and the interactions between the evaluated node and other nodes in the same communication range. Moreover, we use direct interaction counters in the statistics fields of the flow table as part of the recording process. As well, some SDWSN control messages can be used to evaluate intermediate nodes' behavior. For example, receiving a packet-out message means that the intermediate nodes have delivered the packet-in message successfully.

1) Forwarding Parameters (N^{+}_{y}, N^{-}_{y}): Using these parameters, the number of successful and unsuccessful forwarding messages by a particular node y is recorded. Three cases of recording these parameters are defined in the TSW scheme.

Overhearing Approach: In the first overhearing case, as shown in Fig. 4(a), when the node x sends a message to node

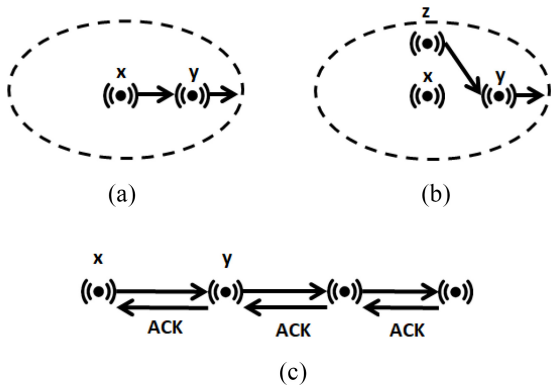


Fig. 4. Recording approaches of forwarding parameters. (a) and (b) are the two overhearing cases and (c) is the ACK-based case.

y that is not the destination, node x overhears the transmissions of node y to check if node y forwarded the message. If the node successfully forwarded the message, then the counter N_y^+ is updated by adding 1. If not, then the other parameter counter N_y^- is updated by adding 1. The second overhearing case is shown in Fig. 4(b). Hence, when node z sends a message to node y (node y is not the destination) and node x is a watchdog for both nodes, then node x overhears node y's transmissions to check whether or not node y forwarded the message successfully. Accordingly, N_y^+ and N_y^- parameters are updated as in the previous case.

ACK-Based Approach: In SDN-based networks, nodes send updates between the controller and sensor nodes in the control plane. In the TSW scheme, we utilize these types of messages to enrich the trust recording for the control traffic. For example, node x sends a message to the controller through node y, as shown in Fig. 4(c). If node x received an ACK or any response confirming that the message is forwarded successfully by node y, the counter N_y^+ is updated by adding 1. Otherwise, if no ACK or response is received, the counter N_y^- is updated by adding 1. The packet-out message is an instance of a response for successfully forwarding a packet-in message. Thus, if node x sends a packet-in message to the controller through node y, and the packet-out message is received by node x, node y performed a successful forwarding, and the success counter is increased by 1.

2) Sending Parameter (N_y): This parameter counts the number of initiated messages by a particular node y (N_y). Two approaches are used to record these counters.

Direct Interaction Approach: The first case is the direct interaction approach, where a node receives messages from its neighbors directly. From Fig. 5(a), when node x receives a message from node y, the counter N_y is increased by 1. If node y is not the source of the message and the source is node z, and it has not already been recorded through overhearing, then the counter N_z is increased by 1. In SDWSNs, the statistics of direct interaction can be retrieved directly from the statistical part of the flow table.

Overhearing Approach: In this case, as shown in Fig. 5(b), when node x overhears node y's transmissions, if node y sends a message, the counter N_y is increased by 1; If node z is the source of the message, and the message has not been recorded,

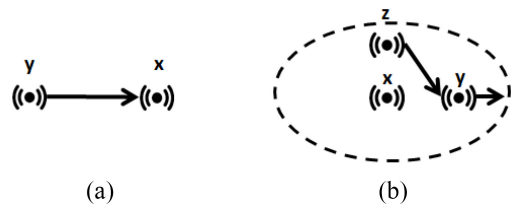


Fig. 5. Recording approaches of sending parameters: (a) direct interaction and (b) overhearing cases.

the counter N_z is increased by 1. However, if node x is a watchdog for nodes y and z, and it records that node z sent that message to node y, then the counter N_y is decreased by 1. Moreover, we define a specific counter of this type to count the number of forwarded packet-in messages.

B. Trust Evaluation Phase

In the trust evaluation phase, each node executes trust computation models based on the parameters resulting from the recording phase. Each type of trust metric has its features, therefore, the evaluation model. As a result, each of these metrics must be computed using different trust computational models. We divide the trust score $[1, 0]$ into three zones, namely, trusted $[1, th_H]$, uncertain $[th_H, th_L]$, and untrusted $[th_L, 0]$. These trust score boundaries can be adaptive based on the two sets of trusted and untrusted nodes and the total number of nodes that contain trusted, trusted, and uncertain nodes [27].

1) Success/Fail Model: Trust evaluation of data forwarding ($T_{x,y}^{DF}$) and control forwarding ($T_{x,y}^{CF}$) metrics depends on successful and unsuccessful hits of the measured property. The Bayesian method is used to compute these types of trust metrics [28]. Additionally, we consider rewarding the successful hits and penalizing the unsuccessful hits. Therefore, rewarding and PFs are defined as shown in (1), which is inspired by [21]. The rewarding term is used to ensure a slow rising of the trust scores and reward credit for the number of successful hits. The PF is used to ensure a fast falling of the trust scores when there are unsuccessful hits and punish the failure. As a result, the falling and rising behavior are less sensitive to the uncertain area. Equation (1) combines the Bayesian method with the rewarding and PFs in the trust computation as follows:

$$T_{x,y}^{\text{Metric}} = \frac{N_{x,y}^{+} + 1}{N_{x,y}^{\text{total}} + 2} \cdot \frac{N_{x,y}^{+}}{N_{x,y}^{+} + 1} \cdot \frac{1}{\sqrt{N_{x,y}^{-} + 1}} \tag{1}$$

where x is the evaluating node, which calculates the trust scores and counts the number of success and failure hits, and y is the evaluated node by node x. $N_{x,y}^{+}$ is the successful count that node x has recorded about node y. $N_{x,y}^{-}$ is the number of failure hits that node x experiences with node y. $T_{x,y}^{\text{Metric}}$ is the trust metric computed by x for y. In (1), the first term is the Bayesian factor and then the rewarding and the penalty terms, respectively.

2) Threshold-Limit Model: The trust evaluation of data sending-rate ($T_{x,y}^{DS}$), control sending-rate ($T_{x,y}^{CS}$), and new flow ($T_{x,y}^{NF}$) trust metrics is done based on a threshold limit. For

example, sending a large number of packet-in messages attacks the control line between nodes and the SDN controller. In this case, each node counts the number of received parameters, e.g., the number of initiated packet-in messages from a particular node. Thus, a defined threshold η for this parameter should not exceed a specific limit. Exceeding the threshold limit must degrade the trust score. Threshold η can be defined as the maximum number of expected messages from the sending node or the maximum capacity of received messages by the receiving node. We set this to be the initiated number of data and packet-in messages.

In the proposed threshold-limit model, the trust score starts degrading at η_1 when N_y approaches the threshold value where $\eta > \eta_1$. The trust score degrades linearly from full trust value at η_1 to $(1/2)$ when N_y is equal to η. Another drop starts from η to η_0, at which point the trust score reaches the zero value. Thus, the proposed threshold equation is as follows:

$$T_{x,y}^{\text{Metric}} = \begin{cases} 1, & N_y \leq \eta_1 \\ 1 - \frac{\eta_1 - N_y}{2(\eta_1 - \eta)}, & \eta_1 < N_y \leq \eta \\ \frac{N_y - \eta_0}{2(\eta - \eta_0)}, & \eta < N_y \leq \eta_0 \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Basically, the distance between η and η_1 must be greater than the distance between η and η_0 because the trust score must drop faster and penalize more when N_y exceeds the threshold value. Hence, the threshold points η_1 and η_0 are calculated as follows: $\eta_1 = \eta/2$ and $\eta_0 = \eta + \eta/4$. Note that η is set by the network manager based on the network statistics or prior knowledge of the expected maximum limit of the number of initiated messages inside the network [29]. This threshold must be lower for a network with a limited bandwidth than for a higher bandwidth network.

Moreover, the trust evaluation of control sending-rate trust metric ($T_{x,y}^{CS}$) depends on lower and higher limit thresholds as it should not exceed a specific limit $\eta_{\min}$ or fall below another specific limit $\eta_{\max}$. The trust score of this node must be degraded when going above or below these two threshold limits. The lower limit threshold $\eta_{\min}$ is defined for control messages due to the properties of SDWSNs where nodes in this network have to send symmetric messages (e.g., statistical information) periodically. Thus, the node is expected to send a certain number of this type of message. A selfish node, which tries to save its residual energy, may refrain from sending these messages. Similarly, thresholds $\eta_{\min}$ and $\eta_{\max}$ are set by the network manager based on the network statistics. The same threshold-limit model is used to determine the trust score based on the lower threshold with few modifications of inequalities and threshold points $\eta_1 = 3\eta/2$ and $\eta_0 = \eta - \eta/4$.

3) Trust Updating: To consider the historical trust scores, the new trust score at time t is calculated by combining the current trust score during the time window (Δt) and the previously calculated trust score at time $(t - \Delta t)$. Therefore, we use an improved trust updating mechanism to compute the trust as follows:

$$T(t) = \begin{cases} (1-\alpha)T(t) + \alpha T(t-\Delta t) \\ \quad \text{if } T(t) \leq T(t-\Delta t) \\ (1-(\alpha+\beta))T(t) + (\alpha+\beta)T(t-\Delta t) \\ \quad \text{if } T(t) > T(t-\Delta t) \text{ and } \alpha+\beta < 1 \end{cases} \tag{3}$$

where $T(t)$ is the new trust score computed for the current window Δt at time t, while $T(t - \Delta t)$ is the previous trust score calculated in the previous window at time $(t - \Delta t)$. T is $T_{x,y}^{\text{Metric}}$ for all symbols. α is a decay factor determining the balance between the current and previous calculations of trust. β is the newly defined decay factor, which gives more weight to the old trust score when the current trust score is greater than the old score. In other words, the trust scheme relies more on the old trust score when the node has behaved maliciously in the past.

4) Trust Integration: Each node computes the trust score of other nodes in every Δt. Each separate trust score depends on a different number of trust metrics. For example, control trust is computed based on control sending rate and control forwarding trust metrics. Having a separate control trust score from the data trust score enhances the ability to detect attacks that target the control plane and network availability. The equation below gives the weighted linear approach to compute the trust scores by combining trust metrics

$$T_{x,Y}^{\text{Type}}(t) = \sum_{m \in \text{Metrics}} w_m T_{x,Y}^{m}(t) \tag{4}$$

where Type represents the separate trust scores for specific applications, such as data and control trust scores. Metrics are the trust metrics used to calculate the specific trust score. However, for each trust score type, not all trust metrics are combined or have the same impact. Therefore, a weighting parameter is used for every trust metric to determine its weight. In the equation, $0 < w_m < 1$ and $\sum_{m \in \text{Metrics}} w_m = 1$. For example, if a metric m is not related to a specific trust type, its weight w_m will be zero; hence, it will not be included in that trust type evaluation. On the other hand, it is possible to have a trust type computed through only one trust metric. Assigning the weighting parameters depends on a specific application to utilize the trust management performance.

C. Trust Propagation Phase

In this section, we describe our defense mechanism against mouthing attacks by computing trust reliability. Furthermore, we describe how the trust aggregation takes place at CH and controller levels.

1) Trust Reliability: The CHs and the controller must validate the aggregated trust scores as some nodes can perform the mouthing attacks. In the TSW scheme, two reliability scores are defined, namely, score reliability and node reliability. The node reliability (T_x^{rel}) is similar to the recommendation trust metric, which evaluates the trustworthiness of a node's positive or negative recommendation of other nodes. However, the score reliability ($R_{x,y}$) is used to detect an outlier from the aggregated trust scores at CH and controller levels.

To compute these reliability scores, first, the average trust score ($AT_{\text{avg},y}$) of collected trust scores of a certain node y is

computed as a base point. Then, the trust reliability of each trust score of node y received by the CH or controller from a node x (score reliability) is computed as follows:

$$R_{x,y}(t) = 1 - \left|T_{x,y}(t) - AT_{\text{avg},y}(t)\right| \tag{5}$$

where the resulting value of $R_{x,y}$ is used to compute the aggregated trust score in Section V-C2. Again, score reliability only determines the weight that must be considered for a single trust score. The score reliability of each trust score is determined separately. However, these values do not determine the evaluation trustworthy of node x. Thus, the node reliability can be calculated by using the score reliability as follows:

$$T_x^{\text{rel}}(t) = \frac{\sum_{y\in Y} R_{x,y}(t)}{N_Y} \tag{6}$$

where N_Y is the number of neighbor nodes. If the calculated node reliability (T_x^{rel}) of node x is in the trusted zone as defined in Section V-B, then this node collects scores at that level, which can be used toward computing trust aggregation. The node reliability has the exact updating mechanism described in (3). To conclude, the node reliability score is considered a special trust score type. This score evaluates the node's trustworthiness when evaluating others. Thus, this trust score is used to detect mouthing attacks and avoid biased trust scores. Consequently, the controller excludes trust scores submitted by a node that has a low reliability score. This isolation guarantees that bias nodes are not capable of affecting the detection efficiency of the TSW scheme.

2) Trust Aggregation: Due to the hierarchical architecture, the trust scores are aggregated at CH and controller levels. At the CH level, the aggregated trust score value is calculated for each node in the cluster. Then, the controller also collects trust scores aggregated by CHs to compute the node's trust evaluation. Usually, to combine the collected trust scores, the average is computed to represent the final trust as follows:

$$AT_{\text{avg},y}(t) = \frac{\sum_{x\in X} T_{x,y}(t)}{N_X} \tag{7}$$

where $AT_{\text{avg},y}$ is the average aggregated trust score for the node y, which is calculated by averaging the received trust scores of node y from different nodes in X. X is a 1-D array of nodes that are able to evaluate the trust of node y. N_X is the total number of x's in X. However, both CH and controller must validate these collected trust scores from the sensor nodes as some nodes can be biased and perform mouthing attacks. To detect any outliers in the aggregated trust scores, we use a modified averaged difference algorithm from spatial weighted outlier detection presented in [30].

Therefore, CH evaluates the trustworthiness of nodes in its cluster, and each cluster node sends the trust scores it computes about its neighbors to the controller. CH computes the cluster-based aggregated trust of each cluster node by averaging the receiving trust scores with weights, where the weight of each trust score is computed by trust reliability

$$AT_{C,y}(t) = \frac{\sum_{x\in C} T_{x,y}(t).R_{x,y}(t)}{\sum_{x\in C} R_{x,y}(t)} \tag{8}$$

where $R_{x,y}$ is a value between [0, 1] called score reliability, which is defined in the next Section V-C1. Using this approach in (8), the aggregated trust score is calculated mainly from the reliable nodes by giving the outlier evaluation less weight to be included in the calculation. In (7), the exact weight is given for all collected trust scores. This leads to the trust score being easily affected by the mouthing attacks. This makes our approach a more realistic approach than the averaging approach in (7).

However, at the CH level, the trust scores are collected from the cluster nodes only. By comparison, the controller collects trust scores from every node in the network. This gives the controller the advantage to evaluate the trustworthiness of any node from a broad sight and increases its ability to detect any malicious behavior. Therefore, the global aggregated trust score is computed as follows at the controller level:

$$AT_{\text{All},y}(t) = \frac{\sum_{x\in \text{All}} T_{x,y}(t).R_{x,y}(t)}{\sum_{x\in \text{All}} R_{x,y}(t)} \tag{9}$$

where $AT_{\text{All},y}$ is the global trust score for node y and is computed by the SDN controller. This score is used by the controller to detect any malicious behavior of node y. Also, it determines the credibility of this node's participation in different network services and applications.

VI. Trust Model Analysis

In this section, we provide an analysis of the trust evaluation models presented in the previous Section V. The analysis is conducted to study the behavior of the trust score under these models. Note that all evaluation is implemented using MATLAB. We study the behavior of the success/fail model, threshold-limit model, updating mechanism, and aggregation approach with some existing models.

A. Analysis of Success/Fail Model

The trust score does not only represent the probability of having a successful experience. However, it is defined as the probability of having a future successful experience given the probability of successful events of past experiences. The Bayesian method is the best approach to determine the posterior probability. Thus, the Bayesian method is used to compute the trust score of success/fail type [17], [23]

$$T_{x,y}^{\text{Metric}} = \frac{N_{x,y}^{+} + 1}{\left(N_{x,y}^{+} + 1\right) + \left(N_{x,y}^{-} + 1\right)}. \tag{10}$$

In the TSW scheme, we use an improved Bayesian method (Section V-B1). We consider a rewarding factor and a PF to make our model more sensitive to the bad behavior in the uncertain area of the trust score. The rewarding factor approaches *one* gradually as the number of successful hits increases, while the PF approaches zero gradually as the number of unsuccessful hits increases. Hence, we consider the density of communication experiences and the density of successful and unsuccessful experiences.

Definition 1: In the TSW scheme, node x considers node y malicious; i.e., $T_{x,y} \leq th_L$ only if $N_{x,y} > 0$ and $N_{x,y}^{-} > N_{x,y}^{+}$, where $N_{x,y}^{-}$ and $N_{x,y}^{+}$ are positive integers.

Lemma 1: TSW is able to detect malicious behavior at node level if the number of successful experiences ($N_{x,y}^{+}$) is less than

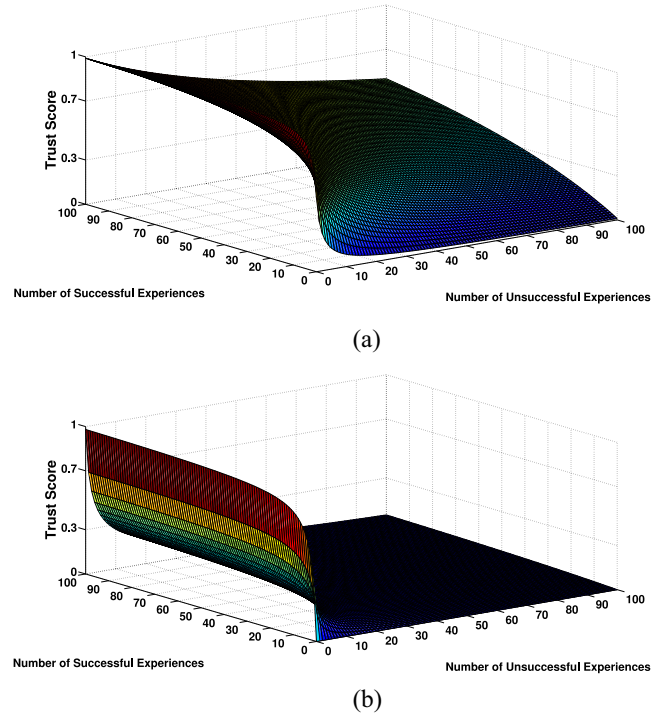


Fig. 6. Success/fail trust score behavior when using three different models with the variation of successful and unsuccessful experiences from 0 to 100. (a) Bayesian model. (b) Proposed model.

the number of unsuccessful experiences ($N^-_{x,y}$). Also, $T_{x,y}$ is sensitive to the bad behavior of y.

Proof: If node y is considered bad, then the computed trust score by node x is $T_{x,y} < th_L$

$$T_{x,y} = \frac{N^+_{x,y}+1}{N^{\text{total}}_{x,y}+2} \cdot \frac{N^+_{x,y}}{N^+_{x,y}+1} \cdot \frac{1}{\sqrt{N^-_{x,y}+1}} \leq th_L. \quad (11)$$

Case 1: When the PF $> th_L$, where $PF = (1/[\sqrt{N^-_{x,y}+1}])$, $N^-_{x,y} < (1/th^2_L)$. So, whatever $N^+_{x,y}$ is, if $N^-_{x,y} > (1/th^2_L)$, node y is considered a malicious node.

Case 2: When PF $< th_L$, $N^-_{x,y} < (1/th^2_L) - 1$. To prove this, we need to prove that $([N^+_{x,y}+1]/[N^{\text{total}}_{x,y}+2])([N^+_{x,y}]/[N^+_{x,y}+1]) < th_L$ when $N^-_{x,y} < N^+_{x,y}$. Then, $N^+_{x,y} < th_L N^+_{x,y} + th_L N^-_{x,y} + 2th_L$. Finally, $N^+_{x,y} < (th_L/1-th_L)N^-_{x,y} + (2th_L/1-th_L)$. As $th_L > 0$, $N^+_{x,y} < N^-_{x,y}$. ■

Lemma 2: When there are no successful experiences, $T_{x,y} = 0$.

Proof: When $N^+_{x,y} = 0$, the rewarding factor is equal to zero ($N^+_{x,y}/N^+_{x,y}+1 = 0$), which leads to $T_{x,y} = 0$. ■

Lemma 3: When there are no unsuccessful experiences, then the node must not be considered a malicious node as $N^+_{x,y} \geq 1$. Also, when $N^+_{x,y} \to \infty$, $T_{x,y} \to 1$.

Proof: When $N^-_{x,y} = 0$, then the PF is equal to 1. This leads to $T_{x,y} = ([N^+_{x,y}]/[N^+_{x,y}+2]) > th_L$, then, $N^+_{x,y} > (2th_L/1-th_L)$ where $th_L \in [0, 1]$ and $N^+_{x,y} \geq 1$. ■

Fig. 6(a) and (b) shows the effect of successful and unsuccessful interactions on the trust score. We vary the number of successful and unsuccessful interactions from 0 to 100 to cover all possible values. Fig. 6(a) is obtained from the Bayesian model in (10). It shows that the Bayesian model behaves similar to a ratio-based model (N^+/N_{total}) when N^{total} is large. However, when N^{total} is small, the Bayesian model tends to not rely very much on the few successful experience counts. Fig. 6(b) shows the behavior of the proposed model. With the rewarding and PFs, the trust score is more sensitive to the unsuccessful experience and drops to low scores exponentially.

B. Analysis of Threshold-Limit Model

Our threshold-limit model in (2) is used to detect flooding behavior. A threshold-limit model is presented in [23] as follows:

$$T^{\text{Metric}}_{x,y} = \frac{1}{\left\lceil \frac{N_y}{\eta} \right\rceil}. \quad (12)$$

However, there are some issues/limitations in using this model to compute the trust score. As shown in Fig. 7(a), the model in (12) sets the trust score to be 1 when N_y is less than the threshold η, even when N_y approaches the η value. Furthermore, the trust score jumps to (1/2) value after the recorded parameter count N_y goes beyond the threshold η. However, N_y does not degrade for some values of $N_y < 2 * \eta$ and the trust score remains the same when N_y is between $[\eta, 2 * \eta]$. This is because the trust score changes with $N_y = k * \eta$, where $k = 1, 2, 3, \ldots$ Finally, we can observe that the trust score does not reach the zero value even if $N_y >> \eta$. In the proposed model, we consider the mentioned issues of (12) when computing this type of trust metrics. Also, we take into consideration the fact that the sensor nodes are resource-limited devices. Thus, the proposed approach must be simple and have lightweight computational operations. Fig. 7(b) shows the outcomes of the proposed model (Section V-B2) with the same threshold values used in Fig. 7(a). The proposed model is sensitive when the number of interactions approaches the threshold value. Moreover, the proposed model becomes more sensitive when the number of interactions surpasses the threshold value by rapidly approaching the zero trust score.

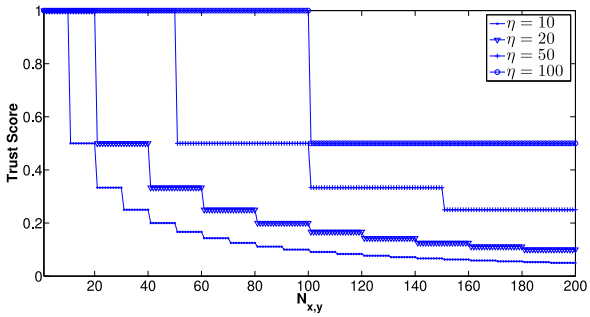
C. Analysis of Trust Updating Mechanism

The trust updating mechanism in [31] is widely used [16], [19], [32] in the trust updating process as follows:

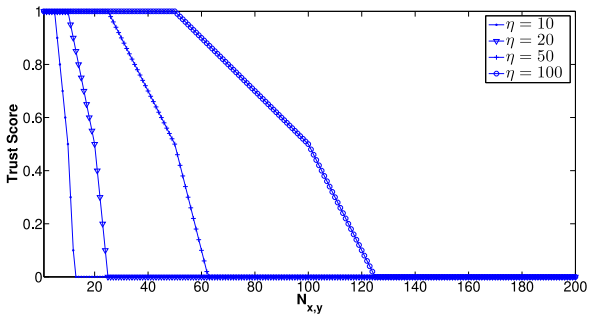
$$T^{\text{Metric}}_{x,y}(t) = (1-\alpha)T^{\text{Metric}}_{x,y}(t) + \alpha T^{\text{Metric}}_{x,y}(t-\Delta t). \quad (13)$$

In the trust updating process, there are two cases, one is falling when $T(t) \leq T(t-\Delta t)$, and the other one is rising when $T(t) > T(t-\Delta t)$. Equation (13) provides similar behavior for rising and falling cases. However, we need to have a slow rising of the trust score to ensure that the premisbehaving nodes are no longer malicious and the system can trust them again. Our proposed mechanism in (3) behaves the same as (13) when $\beta = 0$. However, the trust score rises at a slower rate when $\beta > 0$.

Fig. 8(a) and (b) shows the trust score behavior from the trust updating in (13) and (3), respectively. In these figures, the total number of counts N^{total} is 50. The success counter N^+

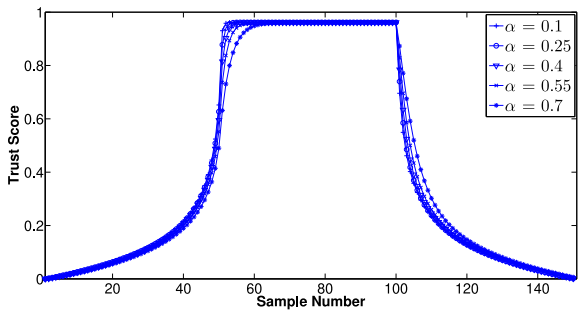


(a)

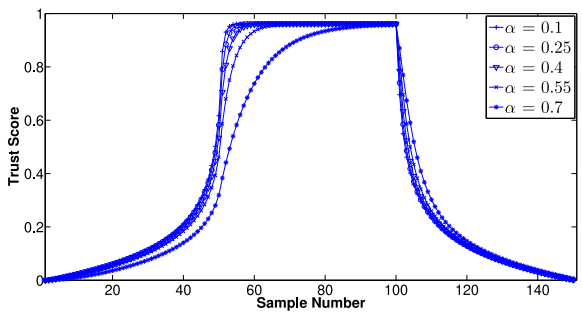


(b)

Fig. 7. Influence of η in the threshold-limit models. (a) ETMRM threshold-limit model. (b) Proposed threshold-limit model.



(a)



(b)

Fig. 8. Influence of α factor in the trust updating mechanisms for success/fail model. (a) Without β factor. (b) With β factor ($\beta = 0.2$).

goes from zero (sample number 1) to N^{total} (sample number 51), and it remains with zero unsuccessful count until (sample number 100). Then, the success count goes to zero again at sample number 150. Similarly, N^- is equal to (N^{total}–N^+). In Fig. 8(b), the trust score rises at a slower rate compared to Fig. 8(a) due to the effect of β factor. That is, it takes longer for a node to be trusted after behaving maliciously. In this figure, the value of β is 0.2.

Definition 2: $|\Delta t|_R$ is the required number of time windows (Δt) of regularly acting by the malicious node to raise its trust score T_y to the trusted zone. The number of Δt, in which malicious behavior is presented, is $|\Delta t|_M$.

Definition 3: The malicious node regularly acts for $|\Delta t|_R > |\Delta t|_M$ to restore its reputation and deceive the scheme.

Lemma 4: The TSW scheme is resilient against the deception of the malicious node. Also, it ensures a slower rising and faster falling of the trust score for each time window based on past behavior.

Proof: As we mentioned, there are falling and rising cases.

Case 1: Assume that there is a continuous falling, i.e., $T_i > T_j$, where $i < j\ \forall i, j \in [1, n]$

$$\begin{aligned}
T &= (1-\alpha)T_n + \alpha T_{n-1} \\
&= (1-\alpha)T_n + \alpha\big[(1-\alpha)T_{n-1} + \alpha T_{n-2}\big] \\
&= (1-\alpha)T_n + \alpha\big[(1-\alpha)T_{n-1} + \alpha\big[(1-\alpha)T_{n-2} + \alpha T_{n-3}\big]\big] \\
&= (1-\alpha)T_n + \alpha(1-\alpha)T_{n-1} + \alpha^2(1-\alpha)T_{n-2} \\
&\quad + \cdots + \alpha^n(1-\alpha)T_0 \\
&= (1-\alpha)\sum_{k=0}^{n}\alpha^k T_{n-k} \\
&= (1-\alpha)T_n + (1-\alpha)\sum_{k=1}^{n}\alpha^k T_{n-k}.
\end{aligned}$$

The first part is the most recent evaluated trust, while the second part determines the effect of old trust scores. As $k \to n$, T_k becomes less effective on the new trust score. Thus, if $n \to \infty$, T_k will be neglected when $k \to 0$.

Case 2: Assume that there is a continuous rising, i.e., $T_i < T_j$, where $i < j\ \forall i, j \in [1, n]$, and $\alpha + \beta < 1$, ($\gamma = \alpha + \beta$ where $\gamma < 1$)

$$\begin{aligned}
T &= (1-\gamma)T_n + \gamma T_{n-1} \\
&= (1-\gamma)\sum_{k=0}^{n}\gamma^k T_{n-k} \\
&= (1-\gamma)T_n + (1-\gamma)\sum_{k=1}^{n}\gamma^k T_{n-k}.
\end{aligned}$$

■

Comparing cases 1 and 2, the model ensures that older trust scores have less weight in the falling case than the rising case as $\gamma > \alpha$, while the newer and larger trust scores have less weight when rising.

Using decay factors, the proposed updating mechanism effectively defends against irregular behavior. The malicious node needs to act nonmaliciously for a long period of time to avoid detection. In other words, to raise its trust score, the node must regularly act for a considerable number of time windows (Δt). The β factor is only used when the recorded trust level of a node is at the low level and the current level is higher, which indicates that this node might be trying to launch an ON–OFF attack. To summarize, a higher α value

means that the new computed trust score relies less on the trust evaluation of the current window than the old trust score that is computed in the previous Δt. A higher β value means that the new computed trust score relies less on the current trust evaluation than the old trust score, which is computed in the previous Δt when the current computed score is higher than the old one. The values of α and β decay factors depend on the environmental and operational conditions of each trust metric [31].

D. Analysis of Trust Aggregation Model

In the TSW scheme, the trust scores are aggregated at CH and controller levels. Moreover, two reliability scores are computed, which are score and node reliability, as discussed in Section V-C.

Definition 4: A reliable evaluator gives $T > th_H$ if a node is considered a good node and $T < th_L$ for bad nodes.

Definition 5: An unreliable evaluator gives $T < th_H$ for good nodes and $T > th_L$ for bad nodes.

Lemma 5: The TSW scheme is robust against up to 54% of unreliable evaluators.

Proof: Assume that K is the total number of evaluators, I is the number of reliable evaluators, and J is the number of unreliable evaluators, where $K = I + J$. Then, (9) can be written as

$$AT_y = \frac{\sum_{k=1}^{K} T_{k,y}.R_{k,y}}{\sum_{k=1}^{K} R_{k,y}} = \frac{\sum_{i=1}^{I} T_{i,y}.R_{i,y} + \sum_{j=1}^{J} T_{j,y}.R_{j,y}}{\sum_{i=1}^{I} R_{i,y} + \sum_{j=1}^{J} R_{j,y}}. \tag{14}$$

Case 1: Assume that a group of unreliable evaluators (J) tries to deceive the system about a nonmalicious node y by providing zero trust, i.e., $T_{i,y} = 1\ \forall i \in I$, and $T_{j,y} = 0\ \forall j \in J$. Then

$$AT_y = \frac{\sum_{i=1}^{I} R_{i,y}}{\sum_{i=1}^{I} R_{i,y} + \sum_{j=1}^{J} R_{j,y}} \tag{15}$$

when $I = J = K/2$ and $AT_{\text{avg},y} = 0.5$. Using (5), $R_{k,y} = 0.5\ \forall k \in K$. Thus, from (14), $AT_y = 0.5$, where $R_{i,y}$ and $R_{j,y}$ are equal to 0.5. This is also applied when a group of unreliable evaluators (J) tries to deceive the system about a malicious node y by providing $T_{j,y} = 1\ \forall j \in J$, while $T_{i,y}$=1 $\forall i \in I$.

Case 2: In this case, we consider the worse case of case 1, in which the reliable evaluators give the nonmalicious node (y) $T_{i,y} = th_H\ \forall i \in I$, while the group of unreliable evaluators tries their best to deceive the system by giving a zero trust score for node y, i.e., $T_{j,y} = 0\ \forall j \in J$.

Assume that $th_H = 0.7$ and $th_L = 0.3$. We have

$$AT_{\text{avg},y} = \frac{\sum_{i \in I} T_{i,y} + \sum_{j \in J} T_{j,y}}{I + J} = \frac{7I}{10N}. \tag{16}$$

Thus, $R_{i,y} = 3/10 - 7I/10N\ \forall i \in I$, and $R_{j,y} = 1 - 7I/10N$ $\forall j \in J$

$$AT_y = \frac{\sum_{i=1}^{I} 0.7.R_{i,y}}{\sum_{i=1}^{I} R_{i,y} + \sum_{j=1}^{J} R_{j,y}} < 0.3. \tag{17}$$

Solving this inequality will lead to $I < 0.45335K$ and $J > 0.54664K$. ■

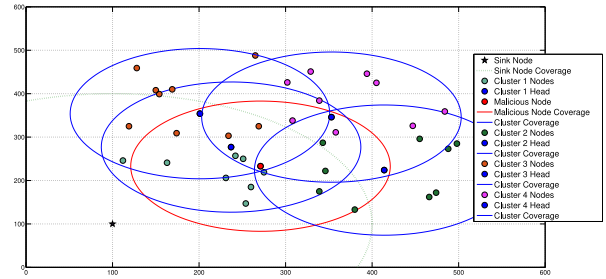


Fig. 9. Example of simulation model.

VII. Performance Evaluation

In this section, we first discuss the simulation setup. Then, we numerically analyze the performance of the TSW scheme in terms of the trust score and the detection rate against several attack scenarios, namely, black-hole, selective forwarding, DoS, good mouthing, and ON–OFF attacks. Finally, we provide an analysis of communication and storage overhead.

A. Simulation Setup

We have conducted simulations with MATLAB to evaluate the performance of the TSW scheme. We consider a network with 40 randomly deployed sensors over an area of 400×400 with one sink node as an SDN controller. We further divide the network into equal-sized fixed clusters. The cluster size is set to be ten nodes, resulting in a total of four clusters. All nodes have a communication range of 150, while the sink node range is 300. The location of the sink is at the extremity of the network (100, 100). The network area with the clusters and sensor placement used in the simulations is shown in Fig. 9. Moreover, we consider two types of nodes: 1) normal nodes, which have good cooperation in forwarding messages and providing trust scores for others and 2) malicious nodes that perform one of the malicious behaviors defined in Section III-C. The simulation proceeds in rounds, where various aspects related to sensor communication and trust evaluation are updated, and each round is equal to a time window (Δt). The simulation ends if it reaches round 1000. The default parameter values for the trust scheme are defined as follows: $th_H = 0.7$, $th_L = 0.3$, w_m is equally distributed for all $m \in$ Metrics, $\alpha = 0.5$, and $\beta = 0.2$.

B. Performance Analysis

First, we study the detection performance of TSW whereby a malicious node initiates several attack scenarios. Fig. 10 demonstrates the trust score as a function of the simulation rounds. In each scenario, the malicious node launches an attack at the 100th round and terminates its malicious behavior at the 200th round.

1) Black-Hole Attack: Fig. 10(a) analyzes the detection performance against the malicious node performing a black-hole attack. The figure presents the trust score computed the SDN controller, the CH of the cluster containing the malicious node, and another cluster member that is a neighbor of the malicious node. Once the attack is initiated, the trust score

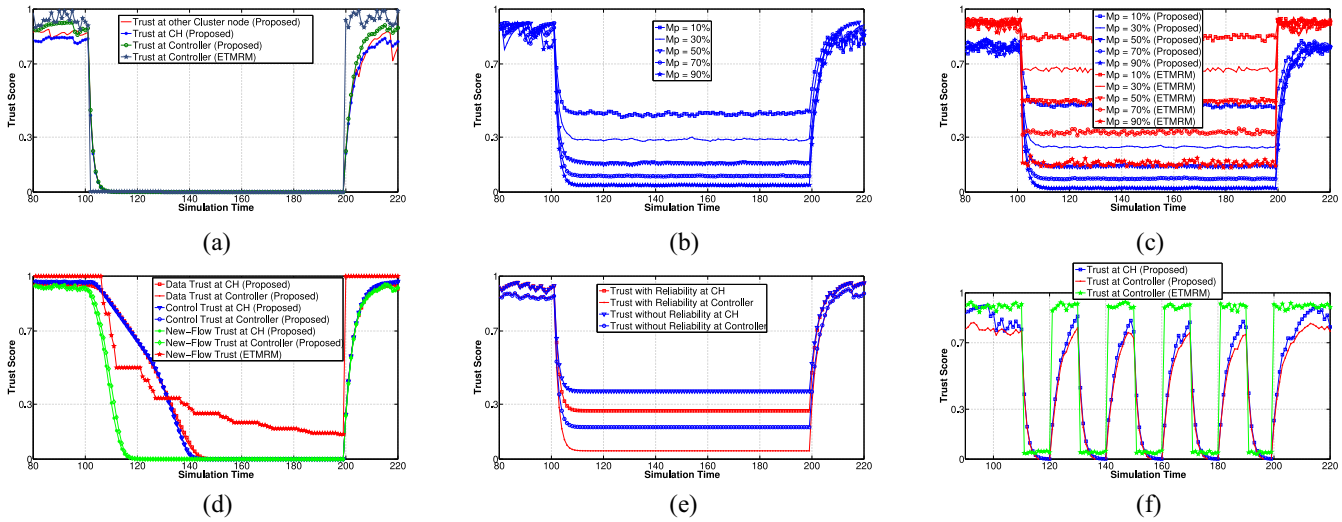


Fig. 10. Detection performance for black-hole, selective forwarding, DoS, good mouthing, and ON–OFF attacks. (a) BH attack detection. (b) SF attack detection at CH level. (c) SF attack detection at controller level. (d) DoS attack detection. (e) BH attack detection with GM attack. (f) Defending against ON–OFF attack.

of the malicious node drops significantly at all SDWSN levels and reaches zero. Note that if the malicious node stops dropping messages, its trust score does not immediately jump from the untrusted to the trusted zone, where the node requires more rounds to gain trust thanks to the proposed updating mechanism.

2) Selective Forwarding Attack: Here, the malicious node randomly drops received messages by a specific drop percentage (M_p). This scenario repeats with different values of M_p. Fig. 10(b) presents the aggregated trust scores computed for the malicious node by the CH. When M_p equals 50%, 70%, and 90%, the TSW scheme is able to classify the malicious node as an untrusted node. However, when M_p is 10%, the CH is uncertain about determining the trustworthiness of the malicious node. When M_p is 30%, the trust score of the malicious node is very close to the uncertain zone.

Fig. 10(c) presents the global trust scores computed by the SDN controller. When M_p is larger than 30%, the controller deems the malicious node as an untrusted. Only when M_p is 10% or less that the controller is uncertain about determining the trustworthiness of the malicious node. As a benchmark, we also compare with the detection performance of ETMRM. TSW detects the malicious node when M_p is larger than 30%, whereas ETMRM detects the malicious node when M_p is larger than 90%. Due to the incorporation of the rewarding and PFs when computing the forwarding trust metrics, TSW outperforms ETMRM.

3) DoS Attack: Here, to launch the DoS attack on several fronts, the sending rate of the malicious node increases gradually for data, control, and packet-in messages. Therefore, we need to observe the data trust metric, control trust metric, and new-flow trust metric. Fig. 10(d) shows that all trust metrics starts dropping once the malicious node initiates the attack. Although the malicious node behavior returns to normal at round 200, the trustworthiness of the malicious node does not enter the trusted zone instantly due to the use of the trust updating mechanism. In contrast, in ETMRM, the trust score jumps to the trusted zone immediately after the attack stops as it does not consider past scores in the evaluation updating mechanism. In addition, the ETMRM scheme does not provide detection functionality at other levels besides the controller level. This prevents the scheme from responding quickly to malicious behaviors.

4) Good Mouthing Attack: We need to determine the effect of biased trust scores due to GM attack. In Fig. 10(e), a malicious node launches a black-hole (BH) attack by dropping received messages. At the same time, other malicious nodes try to cover for this node by submitting a full trust score to the network. Malicious nodes make up for 30% of the total number of nodes. Fig. 10(e) shows the results of the attack detection when the bias score detection is applied and not applied, respectively. Fig. 10(e) shows the effectiveness of trust reliability in avoiding bias trust scores and computing the aggregated trust score from reliable nodes. Therefore, TSW can successfully detect the malicious node that drops messages even when 30% of nodes perform a GM attack.

5) ON–OFF Attack: We evaluate the TSW scheme against the ON–OFF attack scenario, whereby the malicious node performs a BH attack periodically. That is, the malicious node launches the attack for a certain period of time, and then it behaves regularly for another period of time. The period of being ON (behaving maliciously) or OFF (behaving normally) is called the ON–OFF period (ε). For example, if $\varepsilon = 5$, the malicious node launches a BH attack for five time rounds, and it behaves normally for the next five time rounds, and so on.

Fig. 11 shows the detection rate of a malicious node during the attack period with different ON–OFF periods ($\varepsilon = [1, 20]$). We modify the ETMRM to protect against ON–OFF attacks by adding an updating mechanism shown in (13) (named ETMRM_U). In Fig. 11, the TSW scheme ($\alpha = 50\%$) detects the malicious node more than 70% of the attack period. However, the modified ETMRM method recognizes the malicious node as untrusted during the ON period only as it does not consider the historical trust level in the next time window.

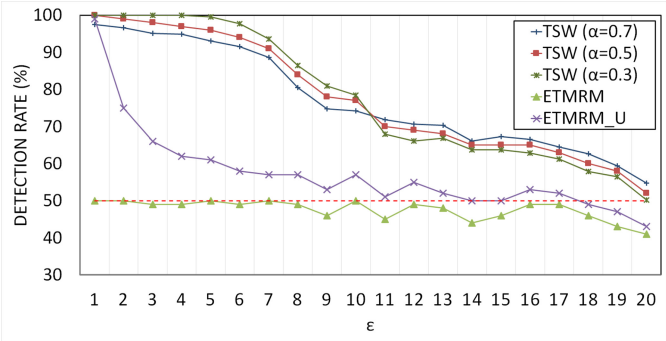


Fig. 11. BH attack detection with ON–OFF attack.

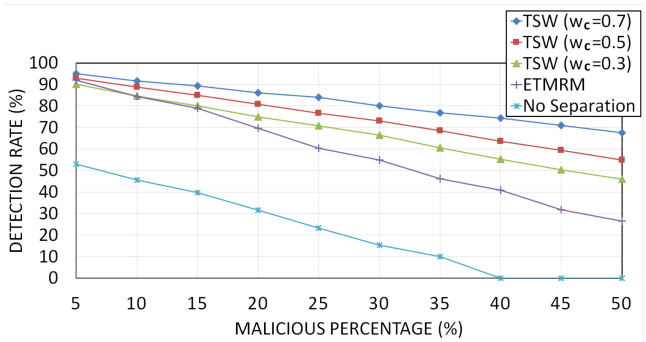


Fig. 12. SF attack detection on the control plane.

In addition, we observe that the detection rate for smaller ε is better when α is low (0.3), while a lower detection rate is observed for longer ε. As shown in Fig. 8, the trust convergence speed is lower when old scores are smaller (slow rising case), while the trust convergence speed is larger when old scores are larger (fast falling case). Therefore, as α increases, it allows the trust scheme to keep the malicious node in the untrusted zone.

In Fig. 10(f), a malicious node launches a BH attack with $\varepsilon = 10$. Due to the use of the trust updating mechanism, the malicious node does not reach the trusted zone immediately. Longer time is needed to reach the trusted zone again compared to the ETMRM scheme. Note that this figure shows only the aggregated trust score computed based on the trust scores collected from the network about the malicious node. However, the controller can have a more restrictive policy for the ON–OFF attacker. For example, the controller can suspend the attacker for a sufficient time period to provide network services such as the forwarding process. Thus, the controller makes the final decision based on its policies on the previously recorded attackers.

6) Attacking the Control Plane: To demonstrate the importance of separating the control and data plane, we simulate an SF attack that deliberately drops control traffic. The number of control messages is less than that of the data traffic. Recall that w_c determines the weight of the trust metrics of the control traffic. Fig. 12 shows the detection rate against the percentage of malicious nodes in the network. Here, we vary w_c from 0.3 to 0.7 and show that our trust management scheme outperforms the no-separation baseline as well as ETMRM. This demonstrates the advantage of a decoupled treatment in our trust management scheme.

7) Collaborative Attack Detection: In this scenario, we evaluate the effect of biased aggregated trust scores on the detection rate of the BH, SF (Mp = 50%), and new-flow attacks. We design a scenario where a group of malicious nodes cooperates to perform one of these attacks while these nodes try to cover each other by performing GM attacks. Specifically, each malicious node sends full trust scores about other malicious nodes to a higher level to avoid their behavior detection. Fig. 13 shows the detection rate of malicious nodes with the increase in the number of malicious nodes in the attacking group (malicious percentage). We observe that the detection rate of BH and DoS attacks is mostly high (above 70%) when the malicious percentage is lower than 40%. A similar trend is shown for SF attack but with a lower detection rate. This is because only half of the messages are dropped. We also observe that with the increase of malicious nodes, the detection rate at the CH level becomes slightly higher than at the controller level. This is because of the increase in the number of collected biased scores at the controller than at the CH level. TSW shows better performance than ETMRM in all cases. ETMRM shows poor performance in detecting the new-flow attack when combined with a GM attack.

C. Overhead Analysis

In TSW, we use the topology discovery protocol for trust reporting [23]. Trust scores are attached to the report messages to minimize the communication overhead. First, we consider the total communication overhead of schemes considering the worse case (when every node wants to connect with every other node). TSW has a communication overhead of $\Pi + \pi(h-1)$, where Π is the total number of nodes, π is the number of clusters (or the number of aggregation points in ETMRM), and h is the average hop count to the controller. Due to the lack of an SDN controller, the communication overhead is larger in distributed protocols since each evaluating node needs to request trust recommendations from other nodes. We calculate the communication overhead for a clustered scheme in [20] to be $2\pi(\Pi^2 + \Pi) + 2\pi^2 + 2\pi$.

The TSW scheme uses lightweight computational models to calculate the trust scores that fit the limited-resource devices in terms of computational overhead. Next, we analyze the memory requirement. Each node must store trust counters and scores for each of its neighbors. The counter is reset every Δt; thus, one byte counter can be sufficient. Assume that τ and p are the total bytes needed for trust scores and counters, respectively, and the storage required for a sensor node in TSW is $(\tau + p)\phi$, where ϕ is the average number of neighbors for a node. For the CH node, additional storage of $\tau(\xi^2)$ is required for the cluster trust matrix where ξ is the average number of cluster members. Table II shows a comparison of the storage overhead. On another note, the overhearing process may cause an issue due to the sensor's power constraint. Many research studies have aimed to optimize the watchdog process and reduce power consumption [33].

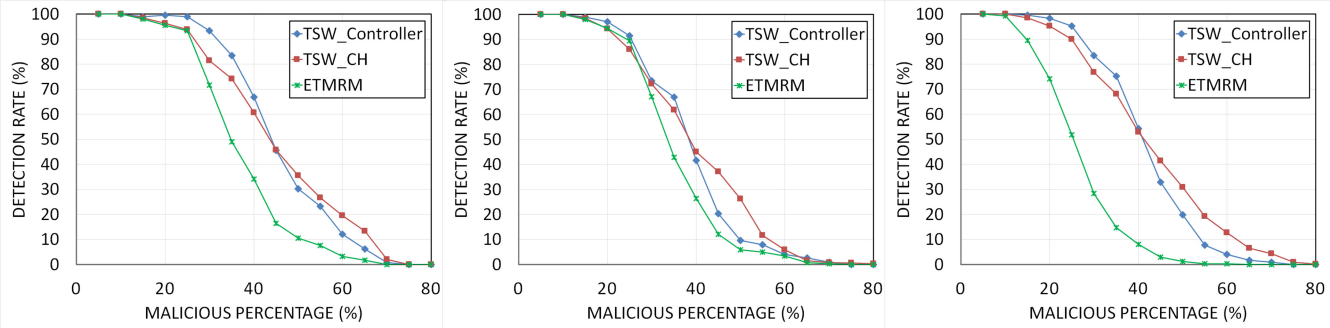


Fig. 13. Collaborative attack detection with GM attack. (a) BH attack detection. (b) SF attack detection (Mp = 50%). (c) New-flow attack detection.

TABLE II
STORAGE AND COMMUNICATION OVERHEAD

Overhead	Schemes	Value
Communication	TSW	$\Pi + \pi(h-1)$
	ETMRM	$\Pi + \pi(h-1)$
	non-SDN	$2\pi(\Pi^2 + \Pi) + 2\pi^2 + 2\pi$
Memory	TSW	$(\tau + p)\phi$, CH: $(\tau + p)\phi + \tau(\xi^2)$
	ETMRM	$(\tau + p)\phi$
	non-SDN	$(\tau + p)\phi$, CH: $(\tau + p)\phi + \tau(\xi^2 + \pi)$

VIII. CONCLUSION

In this article, a hierarchical trust management scheme for SDWSNs called TSW has been designed to secure SDN-enabled WSNss. With TSW, trust scores at each level of the SDWSN architecture can be computed to allow for swift response against malicious nodes to secure network services. Moreover, TSW considers separate trust scores for the data plane and the control plane, respectively, to detect potential elaborate attacks on either plane. Additionally, using outlier detection and weighted averaging mechanisms, TSW can resist the dishonest behavior. The efficacy of the TSW scheme has been demonstrated by simulating and analyzing several communications and trust management threats. For future work, we will design an adaptive and dynamic trust management by considering the energy consumption.

ACKNOWLEDGMENT

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

REFERENCES

[1] J. Ni, X. Lin, and X. S. Shen, "Toward edge-assisted Internet of Things: From security and efficiency perspectives," *IEEE Netw.*, vol. 33, no. 2, pp. 50–57, Mar./Apr. 2019.

[2] N. Vlajic and D. Zhou, "IoT as a land of opportunity for DDoS hackers," *Computer*, vol. 51, no. 7, pp. 26–34, Jul. 2018.

[3] M. Frustaci, P. Pace, G. Aloi, and G. Fortino, "Evaluating critical security issues of the IoT world: Present and future challenges," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2483–2495, Aug. 2018.

[4] U. Baroudi, M. Bin-Yahya, M. Alshammari, and U. Yaqoub, "Ticket-based QoS routing optimization using genetic algorithm for WSN applications in smart grid," *J. Ambient Intell. Hum. Comput.*, vol. 10, no. 4, pp. 1325–1338, 2019.

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

[6] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Comput. Netw.*, vol. 167, Feb. 2020, Art. no. 106984.

[7] H. Mostafaei and M. Menth, "Software-defined wireless sensor networks: A survey," *J. Netw. Comput. Appl.*, vol. 199, pp. 42–56, Oct. 2018.

[8] K. Kalkan and S. Zeadally, "Securing Internet of Things with software defined networking," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 186–192, Sep. 2018.

[9] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging SDN and NFV security mechanisms for IoT systems," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 812–837, 1st Quart., 2019.

[10] S. T. Ali, V. Sivaraman, A. Radford, and S. Jha, "A survey of securing networks using software defined networking," *IEEE Trans. Rel.*, vol. 64, no. 3, pp. 1086–1097, Sep. 2015.

[11] M. Liyanage, A. B. Abro, M. Ylianttila, and A. Gurtov, "Opportunities and challenges of software-defined mobile networks in network security," *IEEE Security Privacy*, vol. 14, no. 4, pp. 34–44, Jul./Aug. 2016.

[12] J. Xie, D. Guo, C. Qian, L. Liu, B. Ren, and H. Chen, "Validation of distributed SDN control plane under uncertain failures," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1234–1247, Jun. 2019.

[13] R. C. A. Alves, D. A. G. Oliveira, G. C. C. F. Pereira, B. C. Albertini, and C. B. Margi, "WS3N: Wireless secure SDN-based communication for sensor networks," *Security Commun. Netw.*, vol. 2018, Aug. 2018, Art. no. 8734389.

[14] F. Ishmanov and Y. B. Zikria, "Trust mechanisms to secure routing in wireless sensor networks: Current state of the research and open research issues," *J. Sens.*, vol. 2017, Feb. 2017, Art. no. 4724852.

[15] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of Things security: A survey," *J. Netw. Comput. Appl.*, vol. 88, pp. 10–28, Jun. 2017.

[16] J. Jiang, G. Han, F. Wang, L. Shu, and M. Guizani, "An efficient distributed trust model for wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1228–1237, May 2015.

[17] W. Meng, "Intrusion detection in the era of IoT: Building trust via traffic filtering and sampling," *Computer*, vol. 51, no. 7, pp. 36–43, Jul. 2018.

[18] T. Zhang, L. Yan, and Y. Yang, "Trust evaluation method for clustered wireless sensor networks based on cloud model," *Wireless Netw.*, vol. 24, no. 3, pp. 777–797, 2018.

[19] R. Chen, F. Bao, and J. Guo, "Trust-based service management for social Internet of Things systems," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 6, pp. 684–696, Nov./Dec. 2016.

[20] X. Li, F. Zhou, and J. Du, "LDTS: A lightweight and dependable trust system for clustered wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, vol. 8, pp. 924–935, 2013.

[21] R. R. Sahoo, A. R. Sardar, M. Singh, S. Ray, and S. K. Sarkar, "A bio inspired and trust based approach for clustering in WSN," *Nat. Comput.*, vol. 15, no. 3, pp. 423–434, 2016.

[22] V. M. Vishnu and P. Manjunath, "SeC-SDWSN: Secure cluster-based SDWSN environment for QoS guaranteed routing in three-tier architecture," *Int. J. Commun. Syst.*, vol. 32, no. 14, p. e4020, 2019.

[23] R. Wang, Z. Zhang, Z. Zhang, and Z. Jia, "ETMRM: An energy-efficient trust management and routing mechanism for SDWSNs," *Comput. Netw.*, vol. 139, pp. 119–135, Jul. 2018.

[24] M. Bin-Yahya and X. Shen, "HTM: Hierarchical trust management for software-defined WSNs," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Waikoloa, HI, USA, 2019, pp. 1–6.
[25] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for IIoT-enabled retail marketing Atop PoS blockchain," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3527–3537, Jun. 2019.
[26] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "Authentication protocols for Internet of Things: A comprehensive survey," *Security Commun. Netw.*, vol. 2017, Nov. 2017, Art. no. 6562953.
[27] R. A. Shaikh, H. Jameel, B. J. d'Auriol, H. Lee, S. Lee, and Y.-J. Song, "Group-based trust management scheme for clustered wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 11, pp. 1698–1712, Nov. 2009.
[28] Y. L. Sun, W. Yu, Z. Han, and K. J. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 2, pp. 305–317, Feb. 2006.
[29] A. Leal, J. F. Botero, and E. Jacob, "Improving early attack detection in networks with sFlow and SDN," in *Proc. Workshop Eng. Appl.*, 2018, pp. 323–335.
[30] Y. Kou, C.-T. Lu, and D. Chen, "Spatial weighted outlier detection," in *Proc. SIAM Int. Conf. Data Min.*, 2006, pp. 614–618.
[31] F. Bao, I.-R. Chen, M. Chang, and J.-H. Cho, "Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection," *IEEE Trans. Netw. Service Manag.*, vol. 9, no. 2, pp. 169–183, Jun. 2012.
[32] B. Sun and D. Li, "A comprehensive trust-aware routing protocol with multi-attributes for WSNs," *IEEE Access*, vol. 6, pp. 4725–4741, 2018.
[33] P. Zhou, S. Jiang, A. Irissappane, J. Zhang, J. Zhou, and J. C. M. Teo, "Toward energy-efficient trust system through watchdog optimization for WSNs," *IEEE Trans. Inf. Forensics Security*, vol. 10, pp. 613–625, 2015.

Manaf Bin-Yahya (Ben 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 Mineral, 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.

Omar Alhussein (Member, IEEE) received the B.Sc. degree in communications engineering from Khalifa University, Abu Dhabi, UAE, in 2013, the M.A.Sc. degree in engineering science from Simon Fraser University, Burnaby, BC, Canada, in 2015, and the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 2020.

He is currently with the Wireless Advanced System and Competency Centre, Huawei Technologies Canada, Ottawa, ON, Canada. His research interests include next generation wireless networks, wireless communications, and machine learning.

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, 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, 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 the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL 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.