Confronting DDoS Attacks in Software-Defined Wireless Sensor Networks based on Evidence Theory

Reyhane Hoseini
Computer Engineering Department, Imam Reza International University, Assar St., Daneshgah St., Mashhad, IRAN,
reyhane_hoseini71@yahoo.com
Nazbanoo Farzaneh*
Computer Engineering Department, Imam Reza International University, Assar St., Daneshgah St., Mashhad, IRAN
Nazbanou.farzaneh@imamreza.ac.ir

Received: 30/May/2020 Revised: 26/Dec/2020 Accepted: 05/Apr/2021

Abstract
DDoS attacks aim at making the authorized users unable to access the network resources. In the present paper, an evidence theory based security method has been proposed to confront DDoS attacks in software-defined wireless sensor networks. The security model, as a security unit, is placed on the control plane of the software-defined wireless sensor network aiming at detecting the suspicious traffic. The main purpose of this paper is detection of the DDoS attack using the central controller of the software-defined network and entropy approach as an effective light-weight and quick solution in the early stages of the detection and, also, Dempster-Shafer theory in order to do a more exact detection with longer time. Evaluation of the attacks including integration of data from the evidence obtained using Dempster-Shafer and entropy modules has been done with the purpose of increasing the rate of detection of the DDoS attack, maximizing the true positive, decreasing the false negative, and confronting the attack. The results of the paper show that providing a security unit on the control plane in a software-defined wireless sensor network is an efficient method for detecting and evaluating the probability of DDoS attacks and increasing the rate of detection of an attacker.

Keywords: Software- Defined Wireless Sensor Networks; Distributed Denial of Service; Entropy; Dempster-Shafer Theory; Evidence Theory.

1- Introduction

A wireless sensor network is made of several wireless nodes able to collect data in non-accessible areas in which human interference could be impossible. In wireless sensor network, there exist numerous limitations because sensor nodes have limited processing power, energy, storing, and bandwidth in wireless links, potential of failure [1, 2].
Software-defined network has been developed as a promising and modern mechanism in improvement of the network. A central software program, called controller, generally manages the network behavior. The software-defined network controller is able to add, update, and remove a flow. Every reaction is actively done in response to information packets using pre-defined rules. Therefore, the software-defined network would be able to quickly react to security threats and traffic filtering, and determine dynamic security policies [3-7].
The software-defined network has emerged as a solution and has been integrated with the wireless sensor network offering it as a modern technology called SDWN (software-defined wireless network). SDWN is not resistant against new attacks due to the physical separation between control plane and data plane. SDWN model uses the software-defined network technique in order to solve many basic problems in the wireless sensor networks; SDWN is facing many challenges though [2-4, 6, 8-18]. In SDWN, the controller sends policies and orders the transportation devices how to face the flows [3, 4, 8-10, 14, 16, 17].
Security is vital for any network; SDWN is not an exception. However, security in SDWN is still in its early stages. Some security solutions can be applicable to SDWN, some cannot. Hence, solving the SDWN security problems is a challenge. Moreover, DDoS attacks are among common attacks in software-defined networks based on wireless sensor networks. DDoS attacks grow in the SDWN environment considering the characteristics of such networks and remain one of the greatest security concerns [3, 15, 16, 18-26].
The present paper, with the help of a security framework on the controller, detects the DDoS attacks which could occur on switches or different nodes. In the proposed method, by a security structure, the suspicious traffic is detected, DDoS attacks are detected, and attacks are

* Corresponding Author

stopped from entering the network main part. The present paper proposes a method for dealing with DDoS attacks in the controller plane using an entropy-based method as a quick and light-weight detection mechanism in its early stages, the history of the flows, and Dempster-Shafer theory for a more exact detection.

The present paper is organized as follows: section 2 offers a literature review. Section 3 provides a brief explanation of the basic concepts of the proposed method. Section 4 is about the proposed method and its importance in the DDoS attacks detection mechanism and the process of detecting such attacks. Section 5 deals with the simulation and evaluation of the proposed method for detecting the DDoS attacks in SDWN environment. It also discusses the simulation results. Finally, section 6 offers the conclusion and the future works areas.

2- Research Background

[15] has offered some solutions to distinguish the slow DDoS attack traffic from lawful traffic. However, nobody has distinguished flash attacks from quick DDoS attacks which are more common. They have used the information theory based on general entropy.

[18] has offered a systematic investigation of various kinds of DoS attacks in software-defined network, and has offered MLFQ to deal with the attack; it allows the queue to develop dynamically and integrate regarding the controller load. [21] has proposed Bohatei method which is a defense method against DDoS using software-based network and NFV. In designing Bohatei, the key takes the address into consideration which is related to scalability, responding, and resisting against attacker. Bohatei simply uses its resources management section to stand against various DDoS attacks; using a few network resources, it can effectively resist attacks.

[26] has offered a defense mechanism against DDoS attacks called CoFence which facilitates the domain-helps-domain cooperation. CoFence is a collaborative network for resistance against DDoS attacks based on network functions virtualization in which the under-attack domain can send the excessive traffic to other collaborative domains for filtering. Specifically, this paper focuses on resource allocation mechanism. It, to allocate the resources, uses the multi-leader-follower Stackelberg game in order to collaborate fairly and share the resources.

Few authors use statistical methods for detecting attacks in software-defined networks. [22] proposed a solution based on entropy. The main purpose of this method is to enable itself to detect the attack up to 500 traffic packets. To do so, it uses the central controller of the software-defined network to detect the attacks. A suggestion for detecting such attacks based on entropy changes is using the destination IP. [24] offers solutions for dealing with DDoS attacks. It uses the flow entropy combined with average entropy technique for detecting attacks. [25] has proposed a mechanism to track DDoS attacks which, according to entropy changes, is between normal traffic and DDoS traffic, basically different from normal traffic. Tracking is done through information theoretical parameters. [23] offers a method similar to techniques like entropy-based system and system anomaly detection for detecting DDoS attacks and preventing them. To investigate a lot of data flow in such environment, a multi-thread IDS approach is proposed. Calculation of entropy for the packets is done using IP address, ports, and flow size. The method presented in [27] used delivery ratio and control packets overhead to detect DDoS attacks in SDWNs. The change point (CP) detection algorithm is used to detect an attack. Method [28] uses several modules to detect the attack and counter the attack in SDIoT networks. If the number of traffic flows exceeds a threshold, the algorithm detects the attack and tries to identify the source of the traffic. If traffic flows are sent from one source, then the attack is definite and that source is blocked. Proposed method in [29] is able to detect an attack in either centralized or distributed mode. The centralized detector has great recognition rates and can differentiate the type of the attacks. The distributed detector offers information that lets to recognize the nodes beginning the attack.

In [30], the method of common intrusion detection is determined for detecting attacks in a cloud. It is a preventive model in which the responsibility of the cloud elements management is distributed among several supervision nodes. In order to detect the common intrusion, Depmster-Shafer evidence theory has been used through the cloud broker role. [31] has proposed a new light-weight trust mechanism called TEDS to detect and increase the effects of Blackhole attacks. The new idea is a combination of entropy function and Dempster-Shafer theory to gain validity for a node. If the validity of a node is less than the threshold amount, it is put on the black list and separated from the network. In [32] Dempster-Shafer theory and combined evidences are used to detect the internal attacker by detection mechanism with the help of neighboring node parameters in wireless sensor network.

[19] focuses on the detection and analysis of DDoS attacks in the environment of cloud calculations. The proposed solution is combining the evidences gained from intrusion detection systems. In the virtual machines, cloud systems are used along with data fusion method. A method has been proposed using a quantitative solution for detecting and analyzing the DDoS attacks in the environment of cloud calculations with the help of Dempster-Shafer theory.

3- Basic Concepts

In this section, the concept of Entropy and Dempster-Shafer theory used in this paper are explained

3-1- Entropy

Entropy or Shannon-Wiener is a significant concept in information theory which measures the amount of uncertainty in the network by an accidental variable or data. The amount of hidden entropy is $[0,\log_m]$ where m is the number of accidental elements [33].
Considering an accidental processing, entropy rate H (X) is calculated from two accidental processes using eq. 1.

$$H(x) = -\sum_{i=1}^{n} Pi \log Pi \quad (1)$$

Entropy H (X) takes an accidental variable X with possible amounts {X1, X2, …, Xn} with the probability of {P1, P2, …, Pn} [33]. And the conditions of eq. 2 are correct for Pi s'.

$$0 \le P_i \le 1 \quad ; \quad \sum_{i=1}^{n} P_i = 1 \quad (2)$$

3-2- Dempster-Shafer Theory

Dempster-Shafer theory is a popular theory used in modeling and reasoning at the time of uncertainty or lack of precision in smart systems. Dempster combination rule is a powerful device used in combining evidences from different information resources. It is used in evaluating the risk and trust capability in engineering issues when the exact measurement of experiments and gaining knowledge from experts' inferences is impossible. A significant aspect of this theory is the combination of evidences gained from various resources and modeling the contrast among them which allows us to reach a belief degree (which is known via a mathematical object called belief function) [32, 34].
Suppose θ is a finite set of elements; an element can be a hypothesis, an aim, or a situation of a system. θ is called frame of discernment. The power set of θ is determined by Ω(θ). For example, if $\theta = \{a,b,c\}$, the amount of Ω(θ) is defined in eq. 3 [35].

$$\Omega(\theta)=\{\{\emptyset\},\{a\},\{b\},\{c\},\{a,b\},\{a,c\},\{b,c\},\{a,b,c\}\} \quad (3)$$

Empty set that shows the flawless system situation $A = \{a,b\}$is a subset of θ : A□θ . A states the system flaw in a or b. θ states the system flaw in a, b, or c.
There are three important functions in this theory which are the base of calculations and equations. They are

- Likelihood function (m)
- Belief function (Bel)
- Likelihood function (Pl)

The probability of the occurrence of the predicate is shown by mass function which is briefly called m and defined as eq. 4 [35].
The mass m(A) of A, a given member of the power set, expresses the proportion of all relevant and available evidences that support the claim that the actual state belongs to A and to no particular subset of A. The value of m(A) pertains only to the set A and makes no additional claims about any subsets of A, each of which has, by addition its own mass.

$$m: \Omega(\theta) \to [0,1], \quad \sum_{i}^{n} m(A) = 1 \quad , \quad m(\theta) = 0 \quad (4)$$

Bel(A) function measures the amount of all probability that should be in elements of A which means certainty about A belief and is the lower bounds on the A probability. The belief function is defined as eq. 5 [35].

$$Bel\ (A): \Omega(\Theta) \to [0,1], \quad Bel\ (A) = \sum m(B) \quad (5)$$

Pl(A) function measures the maximum amount of probability that can be distributed among the elements of A which describes the total belief degree related to A and is the upper bound on A probability. It is defined as eq. 6 [35].

$$Pl\ (A): \Omega(\Theta) \to [0,1], \quad Pl\ (A) = 1 - Bel(A) = \sum_{B \cap A \neq \emptyset} m(B) \quad (6)$$

Where A is the intersection of subsets B and C.

4- The Proposed Method

In this paper, a defense mechanism against DDoS attacks is proposed for the controller. The proposed security model is placed on the controller plane. The purpose is to offer a platform with the help of software-defined network technology as a mechanism for detecting suspicious traffic and attacks and confronting them. Also, the other purpose is to offer a method for providing security on the control plane in order to stop the DDoS attacks from entering the network and, also, to take care of the whole network and the controller as its main part due to being the single point of failure

4-1- The Network Model

The network model of the proposed method is shown in Fig. 1. More explanation about the network model will follow.
SDWN is made of numerous sensor nodes and a sink node. The proposed architecture includes data plane and control

plane. Data plane is made of some wireless sensor nodes; the main part of the network is on the control plane. A set of these nodes connect to the surrounding environment and send the collected information to the controller. Therefore, in the proposed architecture, the connection among sets made of nodes is taken into consideration in which the controller affected by DDoS attack receives malicious traffic flow from these nodes.

For connecting the nodes to the controller, a secure channel is considered in a way that has the smallest delay and the highest trust capability. Secure channel is used for sending control messages and transportation rules from the controller to the sensor nodes, info alert, and changing the message topology from sensor nodes to controller.

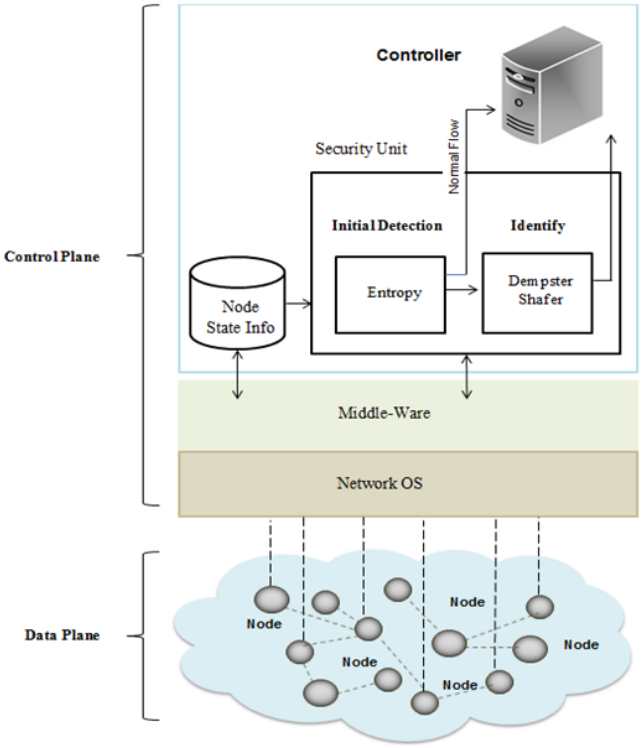

Fig. 1: the network model in the proposed method

Every controller is made of the following components:

Middle-ware

This module is responsible for analyzing and extracting data and updating the database.

Information storage unit

This unit stores the information about the nodes. Such information includes the node number, the node kind, situation, the energy left, etc. When the packets enter the middle-ware of the network, the middle-ware extracts the node information and updates or registers it in the database.

Security unit

Sensor nodes send the flows with various data to the controller. Every traffic flow, after arriving at the control plane and before arriving in the controller, enters the security module to be processed and investigated by this module. Security module is made of the two following parts:

- **First Module:** early detection of suspicious traffic (via entropy module)
- **Second Module:** detecting and confronting the attack (via Dempster-Shafer module)

4-2- The Hypotheses

Bearing in the mind that the main purpose of this research is detecting malicious flows and offering a security model for detecting DDoS attacks, the following hypotheses exist:

- The logical unit of the control plane is considered a part of the sink.
- It is supposed that in this network (software-defined wireless sensor network) the network nodes are fixed.
- The channel is supposed to be secure.
- The controller has to manage the nodes; in fact, the controller can generally view the whole network via flow tables.
- The controller is supposed to be secure, and the flows sent from sensor nodes could be malicious.

4-3- The Details of the Proposed Method

The security unit is made of two main parts. The aim of devising two parts is a more exact detection of attacks. In the first module, considering the input flow and usage of entropy, the traffic flows are classified. Then, suspicious flows are sent to the second module for more investigation. Fig. 2 shows the security unit functioning.

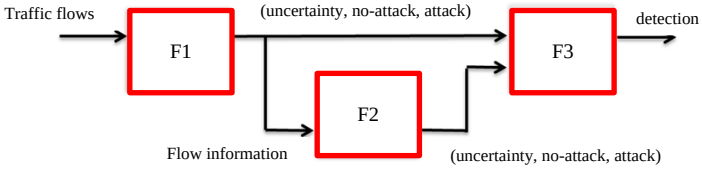

Fig. 2: security unit datagram

As seen,

- F1 is Entropy function:
  - ✓ Input: various traffic flows from certain nodes
  - ✓ Security first-level function: the entropy module measures the entropy value for each input flow

- ✓ Output: a percentage of probability in a certain range (attack, no-attack, uncertainty)
- F2 function (the history of flows)
  - ✓ Input: entropy module output
  - ✓ Function: the information of the flows and the sender of each flow are kept in a memory.
  - ✓ Output: a set of information in n number of various rounds with a percentage of probability (attack, no-attack, uncertainty)
- F3 Dempster-Shafer function:
  - ✓ Input: the information gained from the entropy output and the history of flows
  - ✓ Security second-level function: combination of views and offering the final view about the attack flows
  - ✓ Output: determining the lawfulness or maliciousness of the flow (a percentage of the attack probability, no-attack, uncertainty)

In the following, these two levels of security will be explained in more details.

4-3-1- Entropy: The first level of security

In the first level, the entropy method has been used in the software-based wireless sensor network to detect DDoS attacks. Before the controller is completely attacked, a quick and effective method is needed to work at the control plane. The main reason of choosing entropy used for detecting DDoS attacks is its ability to do accidental measurement in the flows entering the network. However, to stop the overuse of the processing power, the attack detection method has to be light-weight while the attack is occurring.

The entropy measures the probability of an event considering the total number of the events. To detect DDoS attacks, three phases are considered: flow input, traffic analysis, determining whether the flow is malicious or not. The entropy receives the input flows from different nodes and investigates them in a short time. In the simulation process, the flow entropy is used as a measurement tool for detecting attacks because it is efficient and trustworthy. The flow entropy defines the traffic distribution based on certain characteristics such as resource address, destination address, flow ID, etc. It is a significant parameter in traffic analysis. It is important to distinguish lawful and unlawful flows for detecting attacks. To track malicious flows, single flows are calculated and analyzed.

The entropy can, by processing the header of each flow and collecting the flow statistics, quickly detect the suspicious traffic. If the entropy finds a suspicious attack, it can be measured by the entropy.

In DDoS attacks, a great number of fake packets are sent from a group of hosts to the controller. These fake packets can occupy the controller resources and ruin them due to constant processing. Entropy can measure the received packets.

According to the formula, entropy in eq. 1, measures the amount of the flows disorder. In this regard, the entropy keeps 3 amounts for each flow including attack probability, no-attack probability, and uncertainty. The probability amounts in the entropy output determine whether the flow is malicious or lawful.

After the system's early investigation and according to the simulations done in it, a threshold amount is chosen for the entropy. The suitable threshold is chosen through simulation. If the entropy amount is less than the threshold, we consider it as attack flow; if more than the threshold, as lawful flow and normal input flow. Also, the numbers between these two intervals are considered as the probability of uncertainty. Programming is one of the main advantages of software-defined network so when the network construction changes, threshold could be readjusted.

For uncertainty, weighted average has been used as shown in eq. 7.

$$AL(t) = \alpha \times L(t) + \sum_{j=1}^{T} (1-\alpha)^{j} \times L(t-j) \qquad t \geq 1 \quad , \quad 0 \leq \alpha \leq 1 \tag{7}$$

Where AL(t) is the uncertainty weighted average at the time of t, and L is the uncertainty value for various flows. α is the effect and weight of the L values in the past rounds; the older (less) and closer to zero L is, the less effective on the present AL it would be. t is the present time. T is the number of the rounds based on which the weighted average is considered.

After coming to the final conclusion by the entropy plane, the normal flows that have been considered as no-attack enter the main part of the controller, and the first level of security sends the flows considered as attack and uncertain to the second level for more investigation in order to be exactly analyzed by Dempster-Shafer theory and make sure that the suspicious flows detected by entropy have been attacked or not.

In addition to investigation of the flow by the entropy, a history of the flows in n number of the rounds is kept in a memory which is called M2. We calculate the history of the flows for investigating each flow and each sender. In this history, the nodes function and the flows sent by each node in the recent rounds are kept. Hence, the information of various nodes and the amount of malicious node detection and the flows produced by that are registered.

The received information is sent to the second module (Dempster-Shafer) as the output of this section.

Using the history of the flows, as a method in data combination, increases the precision. Entropy measurements and flows history help find the attack resources and stop the DDoS attacks from entering.

4-3-2- Dempster-Shafer: The second level of security

Dempster-Shafer theory is an effective solution for evaluating the probability of DDoS attacks. The purpose of offering this module is having more time for traffic analysis and a more exact detection of attack and confronting it. Based on Dempster-Shafer theory, the information received is combined using different combination rules coming to a final conclusion.

Dempster-Shafer combination rules are shown in the following equations. The combination amount of m(a) is gained from integration of two basic assignment probabilities (m1, m2) and could be calculated according to eq. 8.

$$m(A) = \frac{1}{1-K} \sum_{B \cap C = A} m1\,(B) \times m2(C)$$
$$K = \sum_{B \cap C = A} m1\,(B).\ m2\,(C) > 0 \qquad (8)$$
$$m\,(\emptyset) = 0$$

Where K represents the basic probability mass associated with conflict, and (1-K) denominator in eq. 8 is the normalizing factor.

According to Dempster-Shafer combination method (Monte Carlo), different views will be investigated and, finally, three amounts will be received as output (attack probability, no-attack, and uncertainty). The module results indicate that DDoS attack has been applied onto the input flow, and the malicious flow and lawful flow are detected.

The controller, using the Dempster-Shafer results, decides upon the input flow. If the flow is detected as malicious, it puts it on the black list and, after investigating its function, removes the malicious node from the network.

The proposed method is made of the following stages:

- ✓ **Stage one:** the flows sent from certain senders enter the entropy plane and are calculated and measured by this plane. If the entropy is less than the minimum threshold, the flow enters stage two. Otherwise, the flow is considered lawful and sent to the controller.
- ✓ **Stage two:** the flows history is calculated.
- ✓ **Stage three:** the information taken from the entropy output based on attack and uncertainty is combined with history of flows by the second level of security (Dempster-Shafer) in order to gain a total probability of attack and uncertainty.
- ✓ **Stage four:** the malicious flow is detected and blocked.

The proposed algorithm attempts to be applied with optimum design without nested loop or complicated functions. The application time of the proposed algorithm is O(n) where n is the number of input flows.

The present paper, using a combination of Dempster-Shafer theory and entropy approach, has proposed a method for finding the uncertainty interval for various situations in systems. One of the reasons of using Dempster-Shafer is its theoretical development among non-traditional theories of investigating uncertainty. Also, the combination of different evidences gained from various resources is another advantage.

On the other hand, entropy is one of the ways of measuring uncertainty in finite sets of evidences using their probability distribution function; it can state the incompatibility among evidences resources probable distributions. Therefore, we expect these two methods along with the flows history to be able to detect the changes in traffic behavior of such events and affect the increase of uncertainty.

Using the mentioned levels of security, in addition to detection of the attack, the malicious node could, also, be detected ending in the prevention of the attack. In fact, in the first level, the traffic flows are quickly investigated on-line; in the second, the suspicious flows are analyzed more exactly, in a longer time, and off-line.

4-4- An Example of the Proposed Method

In this section, an example is provided to understand the proposed method more. For instance, the traffic flows including 20 various flows exist from wireless sensor nodes (ex: there are 4 sensor nodes called X1, X2, X3, and X4). We suppose that X2 and X3 are malicious and they send flows containing DDoS attacks sent to the controller. These flows are investigated in the security unit.

In the first level of security, the entropy module, measures the disorder amount of the input flows according to eq. 1 formula; attack probability is 0.7, no-attack is 0.2, and uncertainty is 0.02. The flows which are detected as normal by the entropy are sent to the controller; the flows which are detected as suspicious (attack and uncertainty) are sent to the second level of security for more investigation.

The flows history registers and saves the information and the flows senders, as follows, in a memory.

$$\begin{bmatrix} X3 & 0\,1\,0 \\ X2 & 0\,1\,0 \\ X2 & 0\,0\,1 \\ X3 & 0\,1\,0 \\ X1 & 1\,0\,0 \end{bmatrix}$$ Round 1

$$\begin{bmatrix} X3 & 0\,1\,0 \\ X4 & 1\,0\,0 \\ X2 & 0\,1\,0 \\ X3 & 0\,1\,0 \\ X1 & 1\,0\,0 \end{bmatrix}$$ Round 2

$$\begin{bmatrix} X3 & 0\,1\,0 \\ X2 & 0\,1\,0 \\ X2 & 0\,1\,0 \\ X4 & 1\,0\,0 \\ X4 & 1\,0\,0 \end{bmatrix}$$ Round 3

$$\begin{bmatrix} X3 & 0\,1\,0 \\ X2 & 0\,1\,0 \\ X1 & 1\,0\,0 \\ X3 & 0\,1\,0 \\ X1 & 1\,0\,0 \end{bmatrix}$$ Round 4

As you can observe, for each flow and each sender, the information is registered in the flows history. The attack result is [0 1 0], no-attack is [1 0 0], and uncertainty is [0 0 1]. The flows history keeps 5 stages of input flows in a memory, and when a new flow enters, according to circular shift, replaces the first input flow; finally, the view average of these 4 stages with the a amount of 0.7 (it means that the past flows had less effect on the final results of flows history) as the probability (attack, no-attack, and uncertainty) is considered as output; here, the attack probability is 0.8, and no-attack is 0.2; according to the registered information, X2 and X3 are the nodes attacked by DDoS and send malicious flows. This detection could be the result of over-population or other factors, so these flows are sent to the second level of security for more exact detection and results.

Attack and no-attack results in entropy and the flows history are considered as Dempster-Shafer module input. Dempster-Shafer combines the various views according to combination rules and comes to a complete and final conclusion about malicious flows.

As shown in Table 1, according to Dempster-Shafer theory, after combining the views, attack probability is 0.93, no attack probability is 0.065, and uncertainty is 0. Therefore, DDoS attack caused by these flows is detected. The controller can decide upon these flows and block them.

Table 1. Dempster- Shafer results in the example of the proposed method

	Likelihood amount pl_i		Belief amount bel_i		Mass functionality	
	Ent*	His**	Ent	His	Ent	His
Attack	0.781	0.8	0.77	0.8	0.77	0.8
No-attack	0.219	0.2	0.21	0.2	0.21	0.2
Uncertainty	0.998	1	0.99	1	0.002	0
Attack probability: 0.93 No-attack probability: 0.065 Uncertainty probability: 0 * Ent = Entropy ** His = History						

5- Simulation and Evaluation of the Proposed Method

To simulate, MATLAB has been used. Controller's characteristics is Intel(R) Core(TM) i7 CPU 1.80 GHz, 12GB RAM, windows 10, 64 bit. KDD CUP 99 has been used to send data from various nodes to the controller in the evaluation environment. This database includes a standard set of data investigated by us; it includes numerous simulated intrusions.

In this database 9998 flows exist. Normal flows include 7793 flows and malicious ones include 2205 in this dataset. To each flow, one sender (among 50 wireless nodes) is attributed. Traffic flow from different security unit nodes on the control plane enters the entropy module in order to be processed and investigated by this module in a short time and on-line. Entropy measures the attack probability, no-attack probability, and uncertainty for each flow at a high speed and in 1.56 ms. Table. 2 shows the practical parameters in various functions in the proposed method.

Table 2: Introduction of practical parameters

Parameter name	***Description***
M1	Entropy output
M2	Flow history
DS	Dempster_shafer output
W	Features
a	Suitable features
A	Attack probability
N	No-attack probability
AN	Uncertainty probability
H	Weighted average
Bel_i	Belief amount
Pl_i	Likelihood amount

Each input flow contains 40 features. We, here, have considered 4 of them as more weighted in comparison with others as shown in Fig. 3.

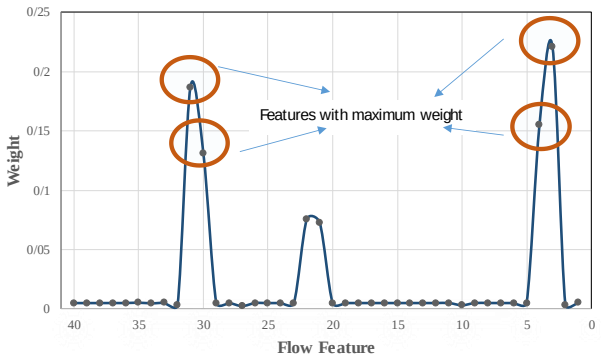

Fig. 3: Flow features weight

As observed in the figure, 40 features are shown on the horizontal pivot. The vertical pivot shows the weight of each feature. The amounts 31, 30, 4, and 3 which have more weight in comparison with others are considered the best features. Every flow containing the above 4 features enters the entropy module for investigation from certain senders (wireless sensor nodes). The entropy amount for each flow is calculated by eq. 1.

A threshold has been considered for the entropy. The minimum is considered 0.2; the maximum is considered 300. This number has been calculated according to the attacks detection rate via trial and error. Amounts less than min threshold are considered as attack probability and amounts more than max threshold as no-attack. The numbers between these two intervals which are less than 300 and more than 0.2 are considered as uncertainty probability. Fig. 4 and Fig.5 show the results of calculating different amounts of threshold.

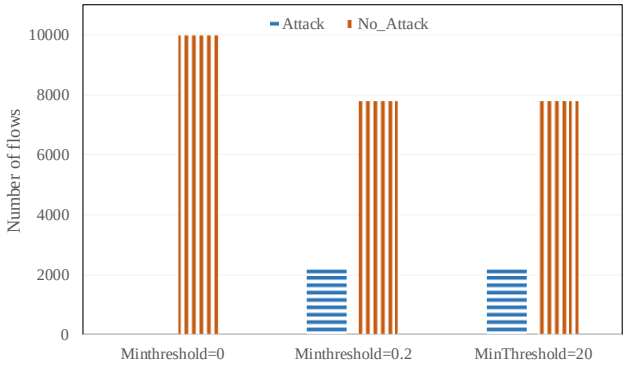

Fig. 4: Calculation of Min Threshold

As observed in Fig. 4, the horizontal pivot is considered as attack, no-attack, and uncertainty. To find the min threshold, 0, 0.2, and 20 are considered. 0 is not suitable for threshold because it cannot correctly detect the attacks. 20 is also not good because it may show more attack probability and more false positive. 0.2 is the best because the number of the attacks it has correctly detected is mainly true positive with less false positive.

Moreover, for max threshold, 3 different amounts are investigated as shown in Fig. 5. Among these amounts, 300 is the best for max threshold because of having less false positive.

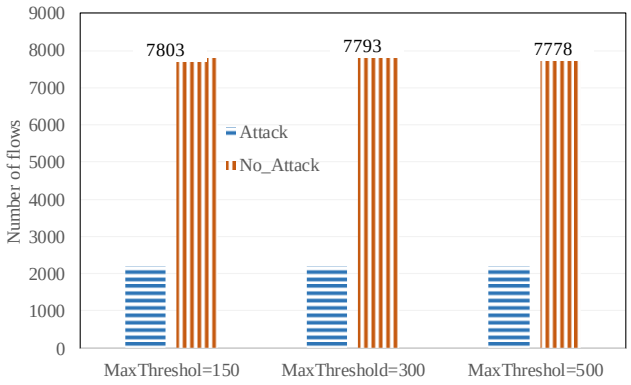

Fig. 5: Calculation of Max Threshold

To calculate the uncertainty eq. 7 is used as shown in Fig. 6. Weighted average calculates the uncertainty for the recent 10 rounds. The weighted average rate (α) is considered 0.5, that is, numbers gained in the past and present are equal. Amounts more than 0.5 mean that the past flows are less effective on final results. In Fig. 6, the amounts of weighted average for different α amounts are shown.

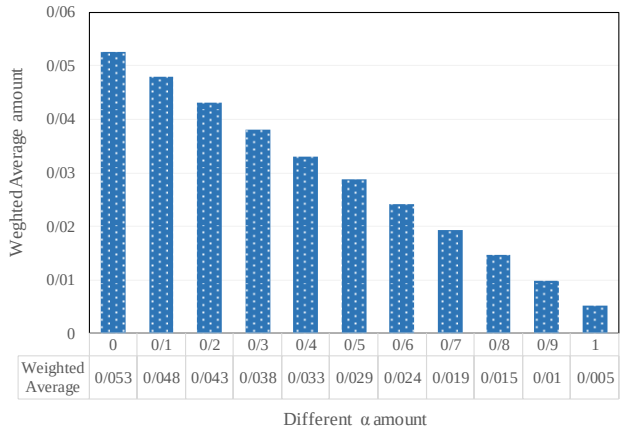

Fig. 6: The effect of α to weighted average formula

The first level of security output includes the calculation of entropy for each flow which is considered as a triple of attack, no-attack, and uncertainty; this output is kept in an array. The results are transformed according to probability

of [0, 1]. The entropy output is shown in Tables 3 and Fig. 7.

Table 3: Entropy module output

	Entropy module output		
	Attack	***No_Attack***	***Uncertainty***
Real amount	2205	7793	0
Real amount (probability)	0.22	0.77	0
Simulation output	2179	7796	23
Simulation output(probability)	0.21	0.787	0.002

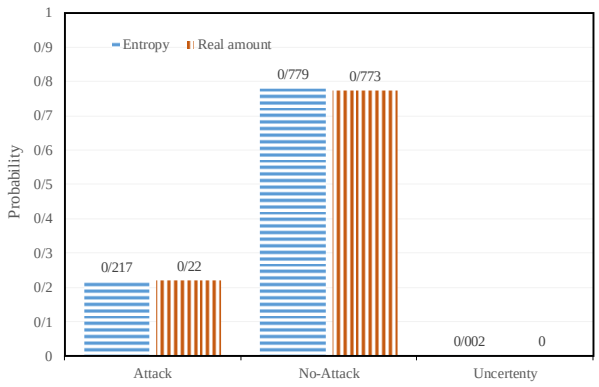

Fig.7: Entropy results

As observed, entropy has been able to detect the amount of normal flows with a high probability percentage. Also, the suspicious flows detected in this module are detected very precisely.

The traffic rate detected by entropy is shown as Fig. 8. As depicted in Fig. 8, in 7000 to 9998 interval, the traffic rate has increased. It means that the attack probability exists in this interval.

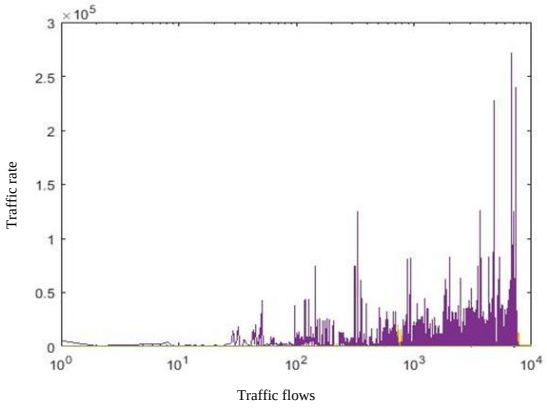

Fig. 8: The traffic rate detected by Entropy

Doing a test on the whole dataset, in the first module, 4 parameters are calculated in terms of efficiency.

- ✓ True positive: the number of the attacks detected correctly
- ✓ False negative: the number of the attacks not detected correctly
- ✓ True negative: the number of flows not attacked and not detected as attack
- ✓ False positive: the number of flows detected as attack while they were not so

Table 4 shows the amount of 4 parameters in terms of efficiency in Entropy

Table 4: Calculation of 4 parameters in terms of efficiency in Entropy

True positive	***False negative***	***True negative***	***False positive***
0.99	0.002	0.96	0

The amounts gained from these parameters are shown in Table 4 which entropy %98.82 has correctly detected the attacks; only 0.2 % of attacks were not correctly determined. Moreover, it has detected the normal flows correctly with %99.6.

The amounts gained from the entropy module output show that entropy can detect the amount of the flows produced by DDoS attacks in an acceptable way. To make sure about the flows considered suspicious by entropy, the suspicious traffic is sent from flows including attack and uncertainty to Dempster-Shafer plane for more analysis.

Moreover, we take the flows history into consideration. A history of flows including 7 rounds is kept in a memory. This history is considered for investigating each flow and, also, each sender which can save the information of the last 7 rounds relative to an event; finally, the views average in these 7 rounds is considered as probability (MT). Table 4 shows the flow history in 7 rounds in simulation test. At each round, it has been assumed that there are 5 input traffic flows and the diagnosis associated with these flows is written in the Table 5. The output amount for each flow from a certain node is considered [0 1 0] for the attack, [1 0 0] for no-attack, and [0 0 1] for uncertainty.

Table 5: The amount of flows history in 7 rounds

	Attack	**No_attack**	**Uncertainty**
Round 1	0	1	0
	1	0	0
	0	1	0
	1	0	0
	1	0	0
Round 2	0	1	0
	1	0	0
	0	1	0
	1	0	0

	1	0	0
Round 3	1	0	0
	1	0	0
	1	0	0
	0	1	0
	1	0	0
Round 4	0	1	0
	1	0	0
	1	0	0
	0	1	0
	1	0	0
Round 5	1	0	0
	1	0	0
	0	1	0
	1	0	0
	0	0	1
Round 6	0	1	0
	1	0	0
	1	0	0
	1	0	0
	0	1	0
Round 7	1	0	0
	1	0	0
	1	0	0
	0	1	0
	1	0	0

As shown in Table 5, in the first round, the first flow has been registered in the memory with the attack probability of [0 1 0]; the second flow with [0 1 0]; the third flow with [0 1 0]; the fourth flow with [1 0 0]; the fifth flow with [0 1 0]. Next rounds are shown in the same way.
When a new flow enters, according to circular shift, it is placed in the first input flow; finally, the average of all 7 rounds of views, with α=0.7, is determined as the triple of attack, no-attack, and uncertainty. Fig. 9 shows the probability values for 7 recent rounds.

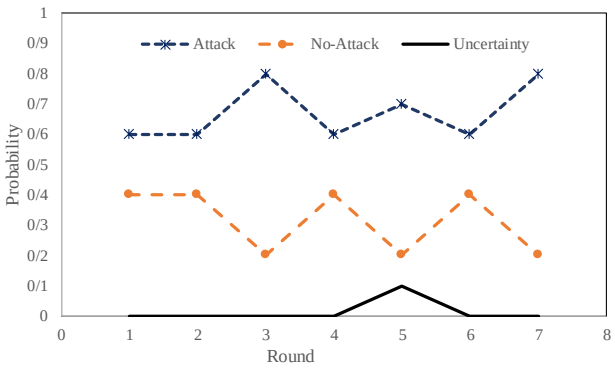

Fig. 9: The flows history in recent 7 rounds

Considering an amount for α, we try to make the numbers gained from the previous rounds affect the present responses. Therefore, we took different amounts of α into consideration; here, 0.9 and 0.7 are considered as the representative of maximum and 0.2 as minimum as shown in Fig. 10.

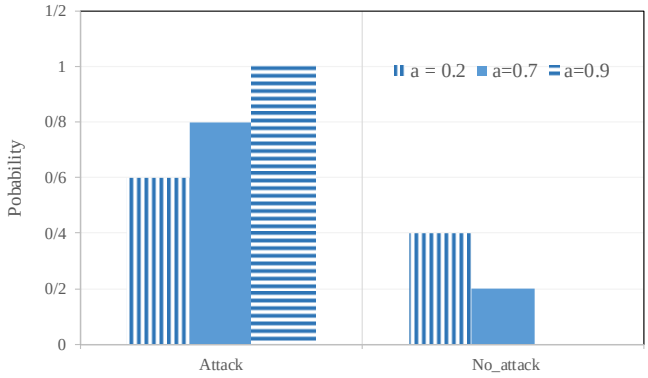

Fig 10: The effect of α amount on flows history formula

As observed, the horizontal levels of the diagram show the attack, no-attack, and uncertainty. For instance, Table 5 shows the results obtained from Dempster-Shafer based on input parameters. Considering α equal to 0.7, this amount shows more attack probability relative to 0.2, that is, the past numbers have less effect on the new responses as shown in Fig. 10. If α equals 0.9, the attack probability is 1. Therefore, the entropy module output and the flows history are sent as second module input (Dempster-Shafer) for a more exact detection and decision-making according to this theory and, finally, coming to a complete conclusion.
For instance, Table 6 shows the results gained by Dempster-Shafer based on input parameters.

Table 6: Demster_Shafer results

	Likelihood amount pl_i		**Belief amount bel_i**		**Mass functionality**	
	Entropy	History	Entropy	History	Entropy	History
Attack	0.22	0.6	0.22	0.6	0.22	0.6
No-attack	0.78	0.4	0.78	0.4	0.78	0.4
Uncertainty	0.99	1	0.99	1	0.002	0

According to the Monte-Carlo combination method, Combination m_1 &m_2
Attack probability: 0.29
No-attack probability: 0.7
Uncertainty probability: 0

The details of combining the results of Dempster-Shafer functions are depicted in Table 7. The results combination is done by Monte-Carlo.

Table 7: Combination of the results of Dempster_Shefer function

	Dempster_Shefer	*M_2 (Flow history)*	*M1 (Entropy)*
Attack	0.22	0.6	0.29
No_Attack	0.78	0.4	0.7
Uncertainty	0.002	0	0

As seen in Table 7, Dempster-Shafer, after combining M1 and M2 belief degrees, has detected the attack amount as being equal to 0.29, that is, it has attributed the DDoS attacks to the malicious flows with higher probability. As the Dempster-Shafer results show, no-attack has decreased relative to entropy results. It has increased the attack probability in the final results.
The simulation results and each function's output result are shown in Fig. 11.

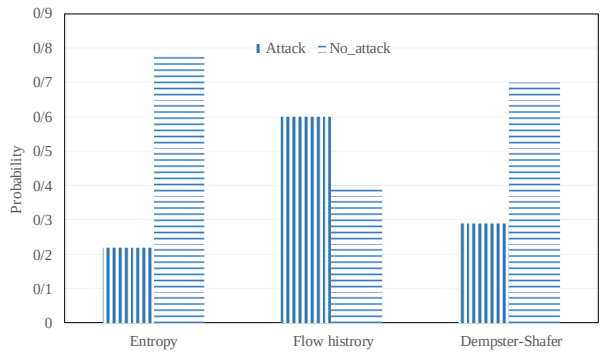

Fig. 11: Comparison of Output Entropy and Dempster_Shafer

As observed, Dempster-Shafer theory, after combining M1 and M2 belief degrees, comes to a certain conclusion about the event. In this example, the second level of security detects the suspicious flow as DDoS attack with greater certainty. According to these results, the controller blocks the detected malicious flow and, by detecting the malicious nodes, stops the malicious sent flow from entering the controller.

6- Conclusions

In SDWN, software-defined network technology is used for solving many basic problems of the wireless sensor network. DDoS attacks are serious threats to modern networks. The main purpose of the present paper is detecting DDoS attacks in their early stages. In the proposed method, to detect DDoS attacks, the central controller of the software-defined network is used, and the entropy approach, as an affective, truly light-weight, and quick solution is also made use of. Also, to confront DDoS attacks, Dempster-Shafer theory is used; it is an effective device for detecting DDoS attacks and finding the attacker. In this paper, in addition to attack detection, the malicious node is also detected ending in the prevention of the attack. By calculating the criteria based on evidence theories like entropy and Dempster-Shafer, the changes differences in the traffic behavior of such events could be detected. The present paper has accomplished the evaluation of attacks aiming at increasing the attack detection rate, maximizing the true positive, decreasing the false negative, and detecting the attack.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, pp. 393-422, 2002/03/15/ 2002.
[2] F. Losilla, C. Vicente-Chicote, B. Álvarez, A. Iborra, and P. Sánchez, "Wireless Sensor Network Application Development: An Architecture-Centric MDE Approach," in Software Architecture, Berlin, Heidelberg, 2007, pp. 179-194.
[3] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in Software Defined Networks: A Survey," IEEE Communications Surveys & Tutorials, vol. 17, pp. 2317-2346, 2015.
[4] A. Akhunzada, E. Ahmed, A. Gani, M. K. Khan, M. Imran, and S. Guizani, "Securing software defined networks: taxonomy, requirements, and open issues," IEEE Communications Magazine, vol. 53, pp. 36-44, 2015.
[5] I. T. Haque and N. Abu-Ghazaleh, "Wireless Software Defined Networking: A Survey and Taxonomy," IEEE Communications Surveys & Tutorials, vol. 18, pp. 2713-2737, 2016.
[6] D. He, S. Chan, and M. Guizani, "Securing software defined wireless networks," IEEE Communications Magazine, vol. 54, pp. 20-25, 2016.
[7] M. Karakus and A. Durresi, "Quality of Service (QoS) in Software Defined Networking (SDN)," J. Netw. Comput. Appl., vol. 80, pp. 200-218, 2017.
[8] Z.-j. Han and W. Ren, "A Novel Wireless Sensor Networks Structure Based on the SDN," International Journal of Distributed Sensor Networks, vol. 10, p. 874047, 2014/03/01 2014.
[9] T. Kgogo, B. Isong, and A. M. Abu-Mahfouz, "Software defined wireless sensor networks security challenges," in 2017 IEEE AFRICON, 2017, pp. 1508-1513.
[10] F. Olivier, G. Carlos, and N. Florent, "SDN Based Architecture for Clustered WSN," in 2015 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2015, pp. 342-347.
[11] C. Ioannou, V. Vassiliou, and C. Sergiou, "An Intrusion Detection System for Wireless Sensor Networks," in 2017 24th International Conference on Telecommunications (ICT), 2017, pp. 1-5.
[12] A. D. Gante, M. Aslan, and A. Matrawy, "Smart wireless sensor network management based on software-defined networking," in 2014 27th Biennial Symposium on Communications (QBSC), 2014, pp. 71-75.
[13] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A Survey on Software-Defined Wireless Sensor Networks:

Challenges and Design Requirements," IEEE Access, vol. 5, pp. 1872-1899, 2017.
[14] S. W. Pritchard, G. P. Hancke, and A. M. Abu-Mahfouz, "Security in software-defined wireless sensor networks: Threats, challenges and potential solutions," in 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), 2017, pp. 168-173.
[15] K. S. Sahoo, M. Tiwary, and B. Sahoo, "Detection of high rate DDoS attack from flash events using information metrics in software defined networks," in 2018 10th International Conference on Communication Systems & Networks (COMSNETS), 2018, pp. 421-424.
[16] S. Shin, L. Xu, S. Hong, and G. Gu, "Enhancing Network Security through Software Defined Networking (SDN)," in 2016 25th International Conference on Computer Communication and Networks (ICCCN), 2016, pp. 1-9.
[17] J. Wu, K. Ota, M. Dong, and C. Li, "A Hierarchical Security Framework for Defending Against Sophisticated Attacks on Wireless Sensor Networks in Smart Cities," IEEE Access, vol. 4, pp. 416-424, 2016.
[18] P. Zhang, H. Wang, C. Hu, and C. Lin, "On denial of service attacks in software defined networks," IEEE Network, vol. 30, pp. 28-33, 2016.
[19] D. E. P. Alina Madalina Lonea, Huaglory Tianfield, "Detecting DDoS Attacks in Cloud Computing Environment," International Journal of Computers Communications & Control, vol. 8, 2013.
[20] Y. Ashok Khimabhai and V. Rohokale, SDN Control Plane Security in Cloud Computing Against DDoS Attack, 2016.
[21] S. K. Fayaz, Y. Tobioka, V. Sekar, and M. Bailey, "Bohatei: Flexible and elastic ddos defense," in 24th {USENIX} Security Symposium ({USENIX} Security 15), 2015, pp. 817-832.
[22] S. M. Mousavi and M. St-Hilaire, "Early detection of DDoS attacks against SDN controllers," in 2015 International Conference on Computing, Networking and Communications (ICNC), 2015, pp. 77-81.
[23] A. Navaz, V. Sangeetha, and C. Prabhadevi, "Entropy based anomaly detection system to prevent DDoS attacks in cloud," arXiv preprint arXiv:1308.6745, 2013.
[24] R. Vadehra, M. Singh, B. Singh, and N. Chowdhary, "Evaluation of Flow and Average Entropy Based Detection Mecha-nism for DDoS Attacks using NS-2," International Journal of Security and Its Applications, vol. 10, pp. 139-146, 2016.
[25] S. Yu, W. Zhou, R. Doss, and W. Jia, "Traceback of DDoS attacks using entropy variations," IEEE Transactions on Parallel and Distributed Systems, vol. 22, pp. 412-425, 2011.
[26] B. Rashidi, C. Fung, and E. Bertino, "A collaborative DDoS defence framework using network function virtualization," IEEE Transactions on Information Forensics and Security, vol. 12, pp. 2483-2497, 2017.
[27] G. A. N. Segura, S. Skaperas, A. Chorti, L. Mamatas, and C. B. Margi, "Denial of Service Attacks Detection in Software-Defined Wireless Sensor Networks," in 2020 IEEE International Conference on Communications Workshops (ICC Workshops), 2020, pp. 1-7.
[28] A. Wani and S. Revathi, "DDoS Detection and Alleviation in IoT using SDN (SDIoT-DDoS-DA)," Journal of The Institution of Engineers (India): Series B, vol. 101, pp. 117-128, 2020/04/01 2020.
[29] G. A. Nunez Segura, A. Chorti, and C. Borges Margi, "Centralized and Distributed Intrusion Detection for Resource Constrained Wireless SDN Networks," arXiv e-prints, p. arXiv: 2103.01262, 2021.
[30] Á. MacDermott, Q. Shi, and K. Kifayat, "Distributed Attack Prevention Using Dempster-Shafer Theory of Evidence," in Intelligent Computing Methodologies, Cham, 2017, pp. 203-212.
[31] h. tan, M. Ma, H. Labiod, and P. H. J. Chong, "TEDS: A Trusted Entropy and Dempster Shafer Mechanism for Routing in Wireless Mesh Networks," presented at the MOBILITY 2014 The Fourth International Conference on Mobile Services, Resources, and Users, Paris, France, 2014.
[32] M. Ahmed, X. Huang, and D. Sharma, "Dempster-Shafer Theory to Identify Insider Attacker in Wireless Sensor Network," in Network and Parallel Computing, Berlin, Heidelberg, 2012, pp. 94-100.
[33] A. Vassilev and T. A. Hall, "The Importance of Entropy to Information Security," Computer, vol. 47, pp. 78-81, 2014.
[34] R. R. Y. Liu, Classic Works of the Dempster-Shafer Theory of Belief Functions: Springer, Berlin, Heidelberg, 2008.
[35] J. H. Ying-Jin Lu, "Dempster-Shafer Evidence Theory and Study of Some Key Problems," Journal of Electronic Science and vol. 15, pp. 106-112, 2017.

Reyhane Hoseini was born in 1993. She received the BSc degree from Payame Noor University, IRAN, in 2016 and a Master's degree from Imam Reza International University in 2018. Hers research topics include Security, trust management, wireless sensor network security, Software defined Network

Nazbanoo Farzaneh received her B.S degree in 2002 and her M.S degree in computer engineering from the Ferdowsi University, Mashhad, Iran in 2006. She received his Ph.D. degree in computer engineering from Ferdowsi University in 2014. She is currently an assistant of computer engineering department of Imam Reza International University, Mashhad, Iran. Her main research interests include Wireless Sensor Network, Next Generation Network (NGN), Vehicular Ad hoc Network, Congestion Control, Quality of Services, Fuzzy Logic Control, Reinforcement Learning, Game theory and Queuing Theory