Accepted Manuscript

Journal of Circuits, Systems and Computers

Article Title:	SDN Framework for Mitigating Time-based Delay Attack
Author(s):	Sagar V. Ramani, Rutvij H. Jhaveri
DOI:	10.1142/S0218126622502644
Received:	27 February 2022
Accepted:	06 May 2022
To be cited as:	Sagar V. Ramani, Rutvij H. Jhaveri, SDN Framework for Mitigating Time-based Delay Attack, *Journal of Circuits, Systems and Computers*, doi: 10.1142/S0218126622502644
Link to final version:	https://doi.org/10.1142/S0218126622502644

Journal of Circuits, Systems, and Computers
© World Scientific Publishing Company

SDN Framework for Mitigating Time-based Delay Attack*

Sagar V. Ramani[†]
Department of Computer Engineering,
Gujarat Technological University,
Ahmedabad-382424, Gujarat , India
[†]*sagarramani@gmail.com*

Rutvij H. Jhaveri[§]
Department of Computer Science and Engineering,School of Technology,
Pandit Deendayal Energy University,
Gandhinagar-382007, Gujarat, India.
[§]*rutvij.jhaveri@sot.pdpu.ac.in*

In the today's era of Cyber-Physical Systems, the conventional security schemes face difficulties in dealing with various attacks as the attackers nowadays have adopted many intelligent mechanisms involving advanced information and communication technologies to launch attacks. These intelligent attacks have a significant impact on the resiliency of the software defined networks. In this paper, we present a real time delay attack on fault-resilient software defined networks.The main objective of the time delay attack is to reduce the resiliency in the SDN-RM (Software-Defined Networking Resilience Management) by adding a delay in the LLDP packets in the Open-Flow Switch. This addition of delay causes degradation in the network performance resulting in a low success rate in the SDN-RM mechanisms. In this paper, we present a machine learning based detection system for detecting the attack. The integration of machine learning techniques with network resilience solutions can adequately address the issue of classifying and predicting the LLDP packets that are delayed at a particular switch. Moreover the detection of the malicious switch at the controller side also results in the improvement of the resiliency in SDN-RM. We propose a machine learning solution to detect the anomalies present in the network topology at the controller side. We use machine learning classifiers such as k-nearest neighbors, support vector machines(SVM), random forest, naive bayes, and decision trees to detect the poisonous switches. The delay-based time attack detection system (DTA-DS) helps the controller to take a reactive decision to improve resilience by detecting poisonous network switches. With the help of the preventive approach, we achieve high fault resiliency in fault resilient-based software-defined networks.

Keywords: Fault resilience, real-time communication, Time-based delay attack, contract-base methodology, software-defined networking.

*Corresponding Author Details: Rutvij H. Jhaveri, Department of Computer Science and Engineering,School of Technology, Pandit Deendayal Energy University, Gandhinagar, India. Email:rutvij.jhaveri@sot.pdpu.ac.in
†First Author Details:Sagar V. Ramani, Department of Computer Engineering, Gujarat Technological University, Ahmedabad, Gujarat, India.
Email:sagarramani@gmail.com

1. Introduction

In Cyber-Physical Systems (CPS), the integration of the software-defined network with critical systems or Information and Communication Technologies (ICT) has drastically transformed technological advancement. However, this type of technological transformation and advancement improves the system performance but at the same time, it poses many security risks. The security of these types of systems is predominantly dependent on the resiliency provided by the software-defined network [1]. By considering the real-time characteristic of the network traffic in CPS, it is often observed that the packets are delivered with a high delay[2]. As a result, network resilience appears as a critical component in such types of systems in order to enable fault-tolerant communications in real-time even when it is challenged by network failures[3] or by constantly fluctuating flows [4]. Advanced firewall and air gaps can be used to protect these sophisticated systems but insider attacks [5] and Stucknex [6] have raised questions against these methods.

For timely transmission of the data packets across network hosts, time-sensitive systems in industrial CPS demand upper constraints on end-to-end delay[1]. As a result, there is a need to meet the delay requirements in the network flow by providing resiliency against the faults in an efficient manner against sophisticated attacks. Therefore, this paper aims to provide the assessment study and the effects of such type of attacks in fault resilient-based software-defined networks. We craft the time-based delay attack on the fault resilient software-defined network to assess the resiliency. In the attack, the adversary node intentionally delays the LLDP packets without altering the content of the packets.The time-based delay attack is carried out by infecting the network's OpenFlow switches with the malware injection, unlike the traditional false data injection (FDI) attack. For several Cyber Physical Systems, the timely delivery of the LLDP packet is crucial. The delay in packets delivery degrades the system performance and reduces the resiliency due to the unsatisfactory contract [7]. Thus, the real-time state of the network is critical from a safety and resiliency point of view. Therefore, if the delay attack puts the network in an unsafe state, an attack remedying strategy must be instantiated to regain the network's safety and resiliency.

If the magnitude of the delay is predicted accurately by the attack rationale algorithm, the attack's effects can be accurately investigated by the control center and appropriate mitigation methods are applied to mitigate the system's adverse effects[8]. The motivation to adopt this mechanism is that the control plane of SDN can detect the poisonous switches by applying ML classifiers which are helpful to make reactive decisions. To craft the time-based delay attack we use Wireshark, and to analyze the behaviour of the approach we make use of packeth tools. The experiments are carried out with Mininet emulator tool with UDP type of traffic and Ryu type of SDN controller along with different machine learning classifiers. The results show that among all five machine learning classifiers (KNN, SVM, RF, Decision Tree, and Naive Bias) KNN performs better in all situations. The main contributions of our work are: (1) An attack based on a time-based delay attack on OpenFlow switches which is stealthy to detect (2) A time-based delay attack detection system that employs machine learning classifiers to detect the delay attack.

Our prior work [9], presented SDN-based routing techniques that enhance Quality of Services(QoS) in robotic cyber-physical systems that have tight real-time requirements. To address the resiliency we extend our work in [1], which provides the real-time flows to furnish resilience in the fault events and to accommodate the variations in the network state. Based on these two mechanisms, we extend our work and add the following new contributions: 1) we address existing mechanisms under delay-based attack and record the results of success rate and network throughput 2) compare and analyze the results of the existing mechanisms with the results in the normal conditions 3) applying the machine learning techniques on the previously recorded results under normal and attack mode detect the delay based attack at the particular switch.

The remaining part of the paper is organized as follows. Section II demonstrates the review of related existing works. While section III represents the threat model of the attack and the effect of the time-based delay attack on the resiliency of SDN-RM. Section IV presents the delay-based time attack detection system (DTA-DS) framework and the result analysis of the framework. Section V concludes the whole paper.

2. Related Work

This section is about a review of the state-of-the-art delay-based attacks and their detection mechanisms. In [10], a thorough examination of time delay injection attacks on network based control systems is presented. The intruder adds time delay into the forwarding channels and the feedback of the control systems. As an artifact of this attack, the adversary interferes in the control system, causing abnormal operational regimes, creates system instability, and potentially forcing the system to crash. However, the gradually increasing delay may put the system in an unstable state for the given approach which tends to reduce the resiliency of the systems. In [11], the authors present a technique for detecting anomalies in Industrial Control Systems(ICS) that studies the network package contents transacted between ICS nodes with their time-series structure. To store the signature database, a Bloom filter is employed for package substance level anomaly detection. In this approach, Long Short Term Memory (LSTM) network-based softmax classifier is used for time-series besed anomaly detection. Based on the historical traffic package data it predicts the most similar package signatures for the system. However, as in [12] authors demonstrated a simple multi-path routing approach that causes latency peaks in the ICS network which are extremely undesirable in the real-time network traffic in an ICS. Therefore, they represented the priority multi-path routing strategy which prevents this delay. This approach uses OpenFlow rule priority to ensure that the switch always has a matching forwarding rule to reduce the number of table misses which adequately reduces the delay peaks, and ensures that selected pathways find a good balance between disjoints and cost.

In [13], the authors have presented scenarios to capture attacks caused by malicious switches that do not follow the OpenFlow protocol. However, existing approach does not include important QoS matrices like throughput, packet drop rate, and latency during detection. In [14], an SVM classifier is used to classify the power system stability by using the historical measured data. However, due to lack of various types of historical data, SVM is required to be retrained. To decrease the extra computational cost in the real-time CPS systems, machine learning [15] and artificial neural networks [16] are applied to identify system stability and provide safety measures against certain emergencies based on the measured physical condition of the system. However, all these studies do not focus on the contingent concern of cyber-physical systems when an attacker attacks the system. In [17], the authors have addressed the controller placement problem (CPP) to achieve scalability and resiliency. To maximize the network resiliency, the proposed approach presents a solution for CPP problem for a large scale network. To cause network disruption, the attacker chooses the malicious node based on certain metrics. However, the selection criteria to choose the malicious node is based on the predictions which may be different in actual network. In [18], the authors demonstrate that existing cryptographic solutions are not effective to detect the TDS. Thus, authors have demonstrated a TDS recovery protocol that identifies the unstable state of TDS attacks and recovers from it. While in [19], the authors present a flow-based intrusion detection approach which makes the use of machine learning classifiers to detect the intrusion in the SDN. But in both approaches, it has been observed that existing solutions do not address the artifacts of the attack in the context of resiliency.

In this paper, we present an approach to focus on testing the system's resiliency by putting the system under time based delay attack. This paper acknowledges easier and hence further arguably more substantial attacks which maliciously delays the LLDP packets between two communicating OpenFlow switches unlike mechanisms proposed in [14,15,16,17,20]. In addition to this, we have analyzed the effects of delay attacks on the SDN-RM by varying the fault events as well as by varying the numbers of malicious switches in the network unlike approach presented in [21,22,23]. Our proposed detection system detects the delay attack by considering possible four scenarios unlike the detection systems presented in [11,26,27,28].

3. Impact and analysis of time based delay attack

This section presents a delay-based attack model in which we assume that an attacker has compromised at least one OpenFlow switch in the network. The main goal of the attacker is to delay LLDP packets. This type of attack is stealthy because the amount of delay time is small which is difficult to be noticed without any specific detection system [22,23]. In addition to this, we consider that controllers, switches, and applications are well protected to prohibit the communication between controllers and attackers by employing rigorous access control policies in the network.

3.1. *Threat Model*

To determine the end-to-end delay between two nodes in the network, we propose Link-layer Delay Estimator (LLDE) model as discussed in [1]. LLDE uses default SDN packets like LLDP and Echo packets to find connection delays and transmission delays to reduce communication overhead.The data related to timestamp is saved in the LLDP packet while it is being transmitted by the SDN controller for upgrading topology and discovery processes. The SDN controller sends Ethernet frames containing LLDP data units to each of its interfaces regularly to verify that all the switches are connected and available for packet routing.

The time based delay attack is formally describe as follows. Let $\omega[t]$ denote the LLDP packets that are transmitted by the controller in the t^{th}time slot. The attacker intentionally delays the LLDP packets at OpenFlow switch by τ time slot. Thus, in the $[t+\tau]^{th}$time slot, the LLDP packet will arrive at the adjacent OpenFlow switch. The delay attack does not temper the data of the packets. Note that the delay (τ) also includes the communication latency. As mentioned this attack can be launched through compromised OpenFlow switch like other DDoS attacks where the target switch gets overloaded by spam requests. Time based delay attack does reduce the system efficiency due unnecessary outdated demands and results in corruption of the switches. In the actual scenario if we consider time delay $\tau \Rightarrow\infty$ then this attack becomes a type of DDoS attack. It's worth noting that DDoS attacks are easily detectable. While in this attack τ time slot is constant during the whole invasion procedure therefore, the time based delay attack is stealthy.

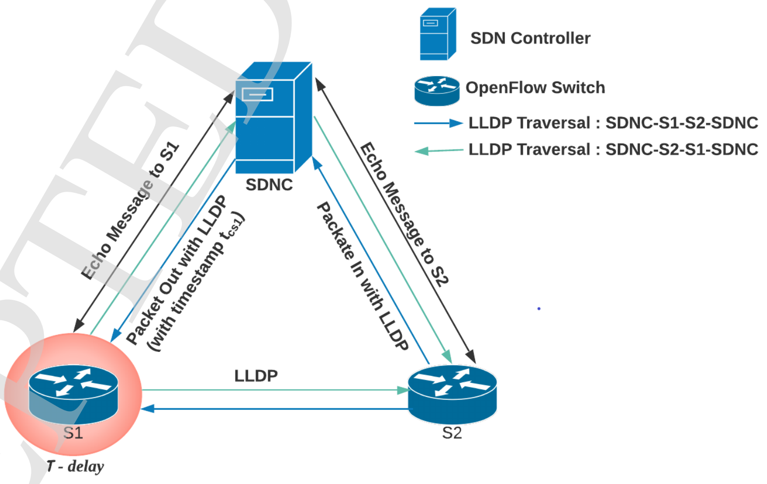
SDN Controller
OpenFlow Switch
LLDP Traversal : SDNC-S1-S2-SDNC
LLDP Traversal : SDNC-S2-S1-SDNC
SDNC
Echo Message to S1
Packet Out with LLDP (with timestamp t_{cs1})
Echo Message to S2
Packate In with LLDP
LLDP
S1
S2
τ - delay

Fig. 1. Delay based attack at switch S1

The controller starts the link finding procedure by sending the Packet-Out message along with the *Datapath-ID*, LLDP time T which contains a *timestamp* ($T_{(c-s1)}$), and *outport-id* of switch *S1* which describe in figure 1. Upon receiving LLDP packets at switch *S1*, the packets will be transmitted to all the output ports of the switch. Switch *S2* advise the controller with the use of *Packet-In* message with LLDP packet which consists of its *timestamp* ($T_{(c-s1)}$) along with the *ingress-port-ID* and *datapath-ID* upon receiving the transmitted packet from switch *S1*. This process is followed by the controller to find the unidirectional link between the two switches *S1* and *S2* at time $T_{(c-s1,s1,s2,c)}$. By sending a *Packet-Out* message with LLDP packet at time T_{c-s2} and receiving a *Packet-In* with LLDP packet at time $T_{(c-s2,s2,s1,c)}$, the controller identifies another unidirectional connection (and hence the entire network topology) from *S2* to *S1*. The controller relay OpenFlow Echo type messages to *S1* and *S2* concurrently to compute the round trip time (RTT) that is $T_{(c-s1,s1,c)}$ (travel time from controller-*S1*-controller)and $T_{(c-s2,s2,c)}$ (travel time from controller-*S2*-controller) respectively. Therefore, the link delay can be calculated by LLDE between *S1* and *S2* by accounting the latency between switches and controller to be bisection of the RTT with the equation 1:

$$\begin{aligned}LinkDelay_{(s1,s2)} = ((T_{(c-s1,s1,s2,c)} - T_{(c,s1)}) + (T_{(c-s2,s2,s1,c)} - T_{(c,s2)}) - \\ (T_{(c-s1,s1,c)}) - (T_{(c-s2,s2,c)}))/2 \quad (1)\end{aligned}$$

Additionally LLDE highlights the estimation of transmission delay, which is assessed regularly throughout the traversal of the LLDP packet across the switches by the ratio of packet length to available bandwidth. As a result, the transmission delay is determined for each switch (for example, transmission delay for switch is *S1*- $TransDelay_{s1}$). Here, with this transmission delay, the attacker will add delay τ at switch *S1*. Thereby, for the compromised switch *S1*, total transmission delay can be realized by equation 2

$$TotalDelay_{s1} = TransDelay_{s1} + \tau \quad (2)$$

In addition, the LLDE recommends the SDN controller to save the link cost in a graph structure. The computation of the link cost from switch *S1* to switch *S2* is done with the help of equation shown in equation (3).

$$LinkCost_{(s1,s2)} = TotalDelay_{s1} + LinkDelay_{(s1,s2)} \quad (3)$$

The end-to-end delay ($ETEDelay_{(s1,sn)}$) for packet travelling from switch *(S1)* to switch *(Sn)* for the linear type network topology of n number of switches is estimated with the equation (4):

$$ETEDelay_{(s1,sn)} = \sum_{i=1}^{n-1} (LinkCost_{(si,s[i+1])}) \quad (4)$$

3.2. *Effect of time delay attack on SDN-RM Resiliency*

To evaluate the performance of SDN-RM, the experiments are carried out on a mininet emulator as shown in figure 2. In the SDN-RM the *resilience manager*(RM) seek to provide the network resiliency to the faults. In order to evaluate the resiliency, the performance metric success rate(percentage of the total number of times that flows delay criteria are fulfilled) is used. Therefore, we have implemented time delay attack on the OpenFlow switches of the mininet tool and have taken the results of the success rate of the SDN-RM. The configuration of mininet parameters for the controller is shown in table 1. The tools and parameters are used for implementation and experimental assessment are listed in Table 1 .

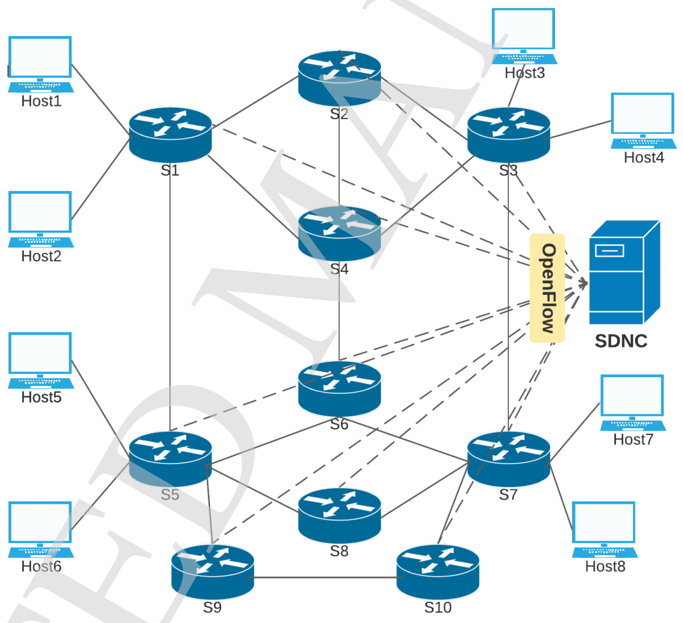

Fig. 2. Mininet 10 switches emulation topology

This section assesses the behavior of SDN-RM by answering the subsequent questions:

- *Q1.*What is the network throughput and the success rate (resilience) of SDN-RM in the case of failure caused by link breakdown ($\varepsilon 1$) and spot variation in delay requirements ($\varepsilon 2$) under the conventional circumstances?
- *Q2.* What is the network throughput and the success rate (resilience) of SDN-RM in the case of failure caused by link breakdown ($\varepsilon 1$) and spot variation in delay requirements ($\varepsilon 2$) under the intrusion?

Table 1. Tools Used for Experiments

Tools/Parameters	**Specification**
Total switches	10
Total hosts	8
Attacker switches	0 to 4%
Link capacity	1 Gbps
Emulation time	300 seconds
Delay assessment period	10 seconds
Traffic Type	UDP
Flow speed	1200Mbps
No of Host Pair	1 (H1-H3)
Traffic generation tool	iperf
Traffic capture file generator	wireshark
Packet delay generation	Packeth

- *Q3.* What is the network throughput and the success rate (resilience) of SDN-RM in the case of failure caused by link breakdown ($\varepsilon 1$) and spot variation in delay requirements ($\varepsilon 2$) with an increasing number of malicious switches in the network?

The following two measures are used to assess the performance of the aforementioned mechanism: 1) success rate (percentage of the total number of times that flows delay criteria are fulfilled) 2) throughput of the network. During the conduction of the experiments, the delay assessment period is chosen to 10 seconds as it is plausible to assume a stable network condition along this period [7].

3.2.1. *Changing number of Events with 20 percent Malicious Switches*

In the set of experiments conducted, the no of events ($\varepsilon 1$ and $\varepsilon 2$) vary from one to five where there are 20% attacks switches in the network. The parameters are set as per table1.

Experiment 1: Link breakdown event($\varepsilon 1$):The link between various switches in the network goes down ($\varepsilon 1$) from 1 to 5 times throughout the simulation. The figure 3 shows the results of the success rate of the networks with 20 percent of the network switches under attack and remaining switches under normal condition. Under this scenario, the comparison of the success rate of the network under normal condition where there is no attack and the network where 20 percent of the network switches are under attack is shown. As the number of link breakdown events increases, the success rate of the network gradually reduces in both scenarios. If we compare both the conditions, as a network under attack and a network in normal condition without any attack, we observe that with every increase in the number of an event under attack condition, the success rate goes down compared to the normal condition. As the number of link breaks increases, the success rate

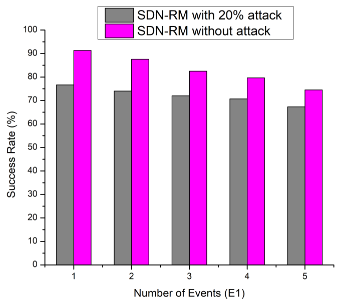

Fig. 3. Success rate along changing number of events ($\varepsilon1$) with 20 % malicious switch

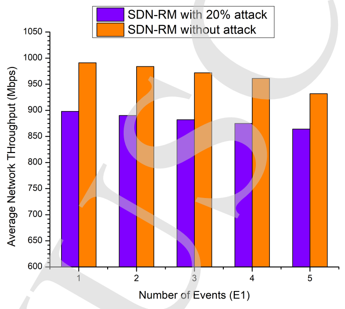

Fig. 4. Throughput along changing number of events ($\varepsilon1$) with with 20 % malicious switch

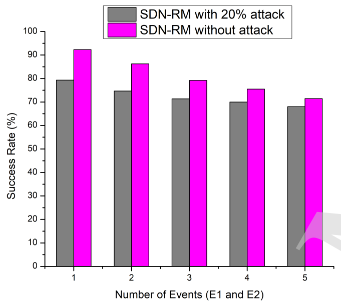

Fig. 5. Success rate along changing number of events ($\varepsilon1$ and $\varepsilon2$) with 20 % malicious switch

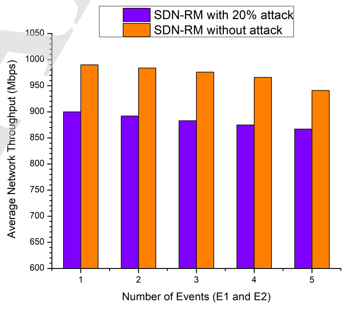

Fig. 6. Throughput along changing number of events ($\varepsilon1$ and $\varepsilon2$) with 20 % malicious switch

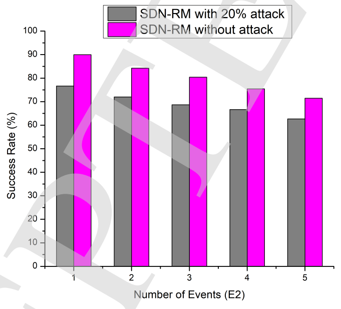

Fig. 7. Success rate along changing number of events ($\varepsilon2$) with 20 % malicious switch

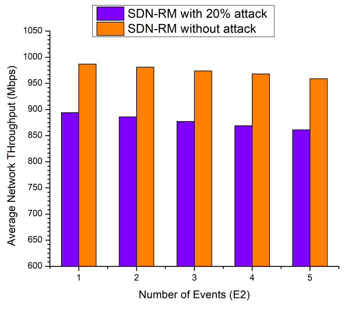

Fig. 8. Throughput along changing number of events ($\varepsilon2$) with 20 % malicious switch

of all the mechanisms decreases due to the increased number of contract violations (faults). Thus, SDN-RM success rate decreases in the case of multiple faults generated in the network. The average success rate observed is 72.13% in the case of 20% (2 malicious switch out of 10) malicious switches in a Mininet topology.

As shown in the figure 4 it is obvious that the network throughput decreases as the number of link breakdown increases. In the 20% (2 malicious switch out of 10) attack condition the average throughput observed is 881.8 Mbps for varying link failure events between 1 to 5. While in the normal condition the average throughput observed is 968 Mbps.

Experiment 2: Link breakdown event and spot variations in delay requirements($\varepsilon 1$ and $\varepsilon 2$): Both events ($\varepsilon 1$ and $\varepsilon 2$) occur with changing frequency (between 1 and 5 times) at different intervals during the execution of the emulation. As shown in the figure 5 with the increasing number of such events, the success rate decreases and the average success rate reported is 72.67% under 20% attack condition in the network. While the success rate of the network under normal conditions observed is 80.95 % which is higher compared to the network under attack condition.

As presented in figure 6 the network throughput decreases with the increase in the no of events($\varepsilon 1$ and $\varepsilon 2$). Average network throughput observed is 883.4 Mbps for the $\varepsilon 1$ and $\varepsilon 2$ events for the network under 20% attack. While for the same network under normal conditions, a throughput of 941.4 Mbps is observed which is high compared to SDN-RM with 20% attack.

Experiment 3: Run-time changes in delay requirements($\varepsilon 2$): The delay requirements are varied ($\varepsilon 2$) from 1 to 5 times at do different points throughout the simulation in this experiment. Figure 7 and figure 8 show that success rate and throughput decreases with the increase in the the no of event ($\varepsilon 2$). The average success rate of the SDN-RM is 69.33% under attack condition compared to success rate of 80.31% under normal conditions. While throughput observed is 877.4 Mbps and 973.8 Mbps for SDN-RM under attack and normal condition respectively. Thus, experiments 1, 2, and 3 address the issue of network throughput and success rate (resilience) of SDN-RM under conventional and normal circumstances. Therefore, it has been observed that Q1 and Q2 are justified by the results of experiments 1,2, and 3.

3.2.2. *Changing Number of Malicious Switches*

In this experiment, we increase the number of malicious switches in the 10 switch mininet topology and analyze the results of the success rate and throughput of the network. The number of poisonous switches varies from 10% to 40% in the mininet topology. During the simulation, we consider fault events($\varepsilon 1$ and $\varepsilon 2$) for 5 times for the results. Other parameters are same as above.

Experiment 4: Link breakdown event($\varepsilon 1$): During the emulation to test the resiliency of the SDN-RM, we have considered 5 times link failure events ($\varepsilon 1$) and present the results for the 10% to 40% poisonous switches in the network.

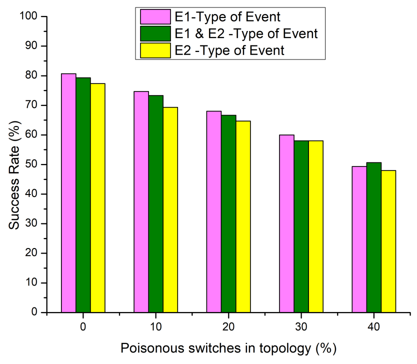

Fig. 9. Success rate with varying number of poisonous switches with faults

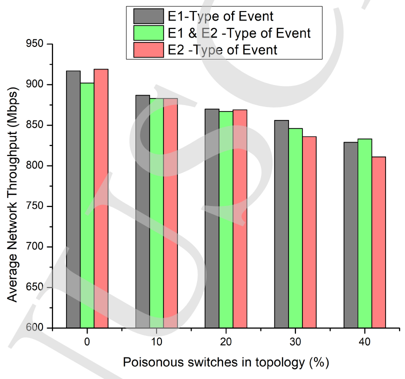

Fig. 10. Throughput with varying number of poisonous switches with faults

Figure 9 shows that when the number of the poisonous switches are increased from 10% to 40% , the success rate of SDN-RM gradually decreases.

Figure 10 shows that throughput decreases with the increase in the number of the poisonous switches which affect directly QoS parameters of the networks. Also in the table 2 we have summarize the results in figure 9 and figure 10 of the Experiment 4.

Table 2. Result Summary of Experiment 4

% of Poisonous Switches in the Network	**Success Rate**	**Throughput**
0%	80.67 %	917 Mbps
10%	74.67 %	887 Mbps
20%	68.00 %	870 Mbps
30%	60.00 %	856 Mbps
40%	49.33 %	829 Mbps

Experiment 5: Link failure and spot variations in delay requirements($\varepsilon1$ and $\varepsilon2$): Both events ($\varepsilon1$ and $\varepsilon2$) occur five times during the emulation at various periods during the execution. For this emulation the results of figure 9 and figure 10 are summarized in table 3.

Experiment 6: spot variations in delay requirements($\varepsilon2$): During the emulation spot variations in delay requirements ($\varepsilon2$) occur 5 times at distinct time instances. For this emulation the results of figure 9 and figure 10 are described in table 4. Thus, experiments 4, 5, and 6 address the issue of the result of network

Table 3. Result Summary of Experiment 5

% of Poisonous Switches in the Network	Success Rate	Throughput
0%	79.33 %	902 Mbps
10%	73.33 %	883 Mbps
20%	66.67 %	867 Mbps
30%	58.00 %	846 Mbps
40%	50.67 %	833 Mbps

throughput and the success rate (resilience) of SDN-RM with an increasing number of malicious switches in the network. Therefore, Q3 is justified by the results of experiments 4, 5, and 6.

Table 4. Result Summary of Experiment 6

% of Poisonous Switches in the Network	Success Rate	Throughput
0%	77.33 %	919 Mbps
10%	69.33 %	883 Mbps
20%	64.67 %	869 Mbps
30%	57.59 %	836 Mbps
40%	48.00 %	811 Mbps

4. Delay base time attack detection System(DTA-DS)

4.1. *Model Architecture*

Our model aims to detect the malicious switches in the network which delay the LLDP packets. To predict the switches which are harmful, the detection module of the controller will examine the network state periodically. The controller can get the LLDP transmission delay of the OpenFlow switch as well as link delay of the network path. To detect the malicious switch, we have developed the Delay-based Time Attack Detection System (DTA-DS) as shown in figure 11. Conventional SDN architecture consists of the controller in the control plane and OpenFlow switches at the data plane. Data plane components follow the instructions from the control plane which was given by the controller in terms of flow rules. In this approach we describe Delay bases Time Attack System (DTA-DS) for the controller which consists of two-modules 1) Contract Observer Module 2) DTA-DS module. The contract observer module is responsible for monitoring the network database during specific time interval. The main purpose of this module is to get the latest

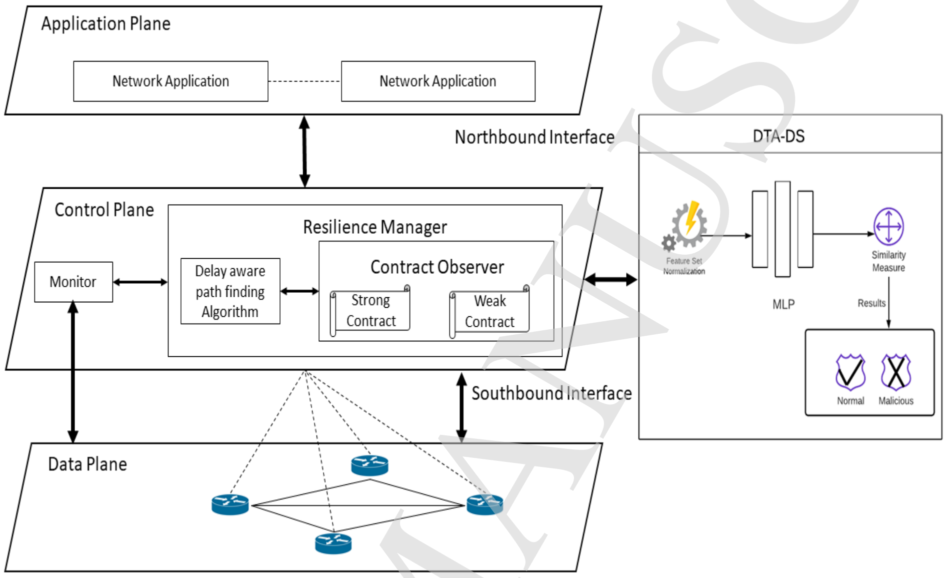

Fig. 11. Architectural design of Delay based Time Attack Detection System(DTA-DS)

data of the network and use it to train the ML model. Before feeding the data sets to train the model, normalization is performed on the data set to get the high accuracy. The machine learning module classifies the malicious switch based on the similarity measure. To calculate the similarity measure, the Euclidean distance is used as a distance metric. The estimated distances are sorted in ascending order, and the top rows are extracted from the sorted items. Then the most frequent class of these rows are fetched and returns the prediction result. Therefore, based on the similarity measure, the OpenFlow switch will be marked as malicious or normal.

4.2. *Model Realization*

We consider classification of the OpenFlow switch as normal or malicious as the binary classification problem. For a switch to be marked as benign or malicious, we consider two separate classes that represent two conditions of the network traffic. To train the ML model we have considered the following scenarios:

- **Scenario 1:** LLDP packets get delayed due to delay based attack
- **Scenario 2:** LLDP packets get delayed due to traffic congestion
- **Scenario 3:** LLDP packets get delayed due to link failure between specific switches
- **Scenario 4:** LLDP packets get delayed due to error at the time of sending or receiving packets at particular switch

To train the model, we have considered the above scenarios to achieve maximum accuracy for the detection. We have created a situation in the mininet network topology as per above mentioned scenarios and generated the datasets to train the ML model. Refer table 5 which describes the feature sets of the network measurements. In the table 5 SWF is a target variable based on that ML algorithm that takes the decision. Out of total datasets, we split the datasets into two parts 0.20 for training and 0.80 for testing purpose. We normalized the data before applying the ML algorithm and did the hyper parameter tuning before applying the algorithm.

Table 5. Feature sets convention table

Feature Sets	**Convention**
Error in packet during receiving time	RX_Error
Error in packet during transmission time	TX_Error
LLDP packet processing time in switch	PD
Bandwidth of connected link	BW
Switch id flag	SWF
Port status of the Switch	PS

4.3. *Knowledge Discovery*

All the feature sets which are cited in the table 5 are important to identify the malicious switches in the network. It is necessary to realize the feature importance of the feature set which are being used in the training and testing of the ML model. If we discuss scenario 1 perspective LLDP packet will get delay due to time-based delay attack, then feature vector set $< SWF, PS, PD >$ will take the decision based on PD during the model decision. While other traffic feature sets will remain as per normal traffic of the network. While scenario 2 perspective model shows that the LLDP packets that get delayed due to traffic congestion not due to poisonous switches. In this type of situation, correct network switch combination of feature vector set $< SWF, BW, PD >$ will take the decision. Scenario 3 represents the model that distinguishes that type of nonpoisonous switches in the network in which packets get delayed due to link failure between specific switches in the network. In this situation feature vector set $< SWF, BW, PS, PD >$ have more importance to classify the switch. For scenario 4, it projects that LLDP packets get delayed at the switch side due to packet-related errors either during receiving or during transmission in normal conditions in the network. Therefore, considering this situation in the network feature vector set $< RX\_Error, TX\_Error, SWF, PD, PS >$ will decide the attack. Therefore, to achieve high accuracy in the model scenario 4 plays a vital role in the model.

4.4. *Algorithmic Introduction*

4.4.1. *Normalization:*

In machine learning, normalisation is a data preparation procedure that is often utilised. Normalization is the process of converting the values of numeric columns in a dataset to a similar scale without distorting disparities in value ranges. When the datasets do not follow the Gaussian distribution, normalization is suitable for data pre-processing. Therefore, it is important in algorithms like K- Nearest Neighbour, which does not speculate any data distribution.

$$\hat{D}[i,:] = \frac{d[i,:]}{||d[i,:]||} \tag{5}$$

In the equation 5, D is dataset in which has N number of columns(features sets), M number of rows(entries) and, *D[i,:]* represent entry i. A vector is expanded or reduced to a unit sphere when scaled to a unit range. The altered data may be viewed as a group of vectors with varying orientations on the D-dimensional unit sphere when applied to the complete dataset.

4.4.2. *K-Nearest Neighbors:*

The KNN algorithm is a supervised machine learning technique. The K-nearest neighbors of the unseen data coordinates are found for an appropriate K value, and the associated class (malicious or normal) is assigned to the unseen data coordinates by making the class with the most data coordinates out of all K neighbor classes. The distance between data coordinates is highly important in KNN. As a result, before using the KNN method, the input data vectors must be scaled to achieve good result score. The complexity of model is define by the value of K parameter. The complexity of model escalates when the value of parameter K rises. As a result this model introduces more over-fitting.

$$Euclidean - distance(d, d') = \sqrt{\sum_{(j=1)}^{n} (d_j - d'_j)^2} \tag{6}$$

$$P(y = i \mid X = d) = \frac{1}{k} \sum I(y^{(j)} = i) \tag{7}$$

The Euclidean distance formula is represented by equation 6, where d and d' are two data coordinates in m-dimensional space. d_j and d'_j are Euclidean vectors originating at the origin of the space. The input d belongs to the class with a higher probability computed as in 7 where k is number of neighbors.

4.5. *DTA-DS model evaluation Metrics and Results*

The performance measures produced from the confusion matrix are used to quantify the accuracy (AC), precision (P), recall (R), F1-Score, and RoC AUC Score of the intrusion detection rate. Table 6 describes the confusion matrix that depicts the algorithm's performance.

Table 6. Confusion Matrix Table

	Predicted as a Normal	**Predicted as an attack**
Normal type of switch class	True Positive(TP)	False Positive(FP)
Poisonous type of switch class	False Negetive(FN)	True Negetive(TN)

An adequate detection strategy has a high rate of detection and accuracy, as well as a low number of false alarms. The number of false alarms is related to the rate of miss classifications. Below mentioned metrics are used to evaluate the model, and the formulas for calculating them are presented:

- Accuracy (AC): It describes the ratio of correctly identified poisonous observation out of total observation done by the model.
$$Accuracy = \frac{TN + TP}{FP + TP + FN + TN}$$
- Precision (P): It demonstrates the percentage of detection correctly predicted malicious switches to total predicted switches in the network. If the value of this metric is high which indicates the lower false alarm rate.
$$Precision = \frac{TP}{FP + TP}$$
- Recall (R): It illustrate the ratio of correctly predicted malicious switches to all actually malicious switches. High recall values shows model have correctly identified malicious switches from malicious switches class.
$$Recall = \frac{TP}{TP + FN}$$
- F1-Score (F): It gives a more accurate measure of accuracy by balancing accuracy and recall.
$$F1 - Score = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

Above table 7 shows the metric comparison of ML classifiers for mininet topology (figure:2). For dataset collection, all scenarios are considered as mentioned in section 4.2. For training and testing purposes to ML classifier models total 17460 number of tuples are collected as per feature set mentioned in table 5. To evaluate the results of different models of the ML, we have split 20% and 80% of the datasets for training

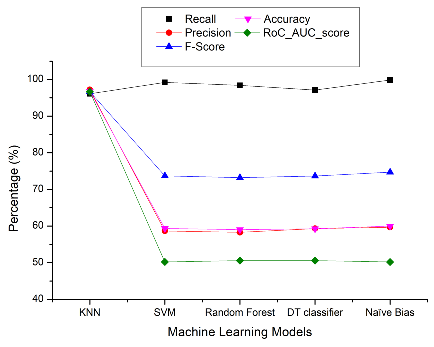

Fig. 12. Machine learning models metrics comparison for Mininet topology

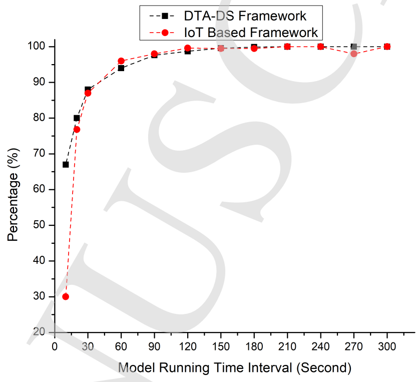

Fig. 13. F1-Score comparison between DTA-DS framework and IoT base framework

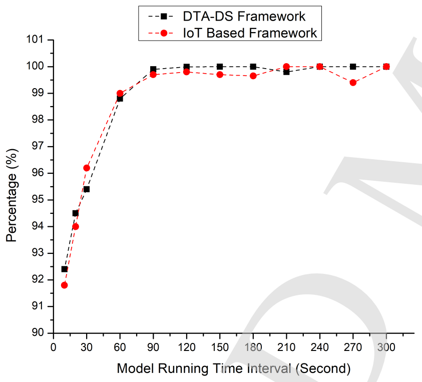

Fig. 14. Accuracy comparison between DTA-DS framework and IoT base framework

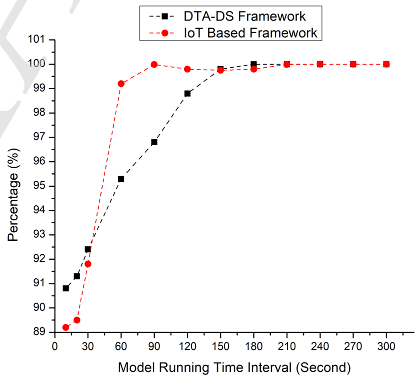

Fig. 15. Precision comparison between DTA-DS framework and IoT base framework

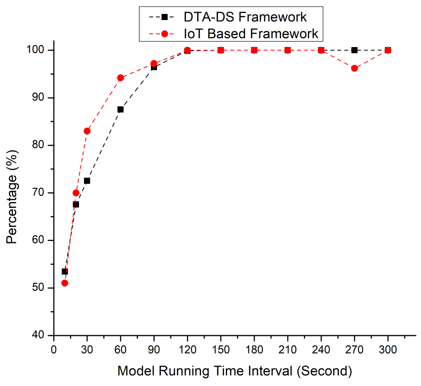

Fig. 16. Recall comparison between DTA-DS framework and IoT base framework

Table 7. Different ML classifiers performance metrics comparison for Mininet topolgy

Type of Classifiers	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**RoC_AUC_Score**
K Nearest Neighbors	96.82	97.22	96.11	96.66	96.67
SVM	59.36	58.66	99.22	73.72	50.19
Random Forest	59.02	58.28	98.43	73.21	50.55
Decision Tree Classifier	59.27	59.32	97.11	73.65	50.57
Naïve Bias	60.01	59.68	99.88	74.71	50.19

and testing purposes respectively. We used 10 fold cross-validation approach in our experiment.

In the above figure 12 KNN model shows the highest Accuracy, Precision, F1-score, and ROC_AUC score compared to other models. On the other hand, SVM model shows a higher recall value (99.22%) compared to other models. If we compare the recall value of the KNN model with the SVM model, the KNN model's recall value is 96.11 which is slightly lower than the SVM model. If we consider and interpret all the values of metrics of the models, overall KNN model performs well compared to other models.

In addition to this, we have compared our framework with the fog-assisted SDN Controlled Framework for enduring anomaly detection in an IoT network [29]. It is a fog-assisted software-defined networking (SDN) driven intrusion detection system for IoT networks. They have used UNSW-NB15 datasets for training and testing of the ml classifiers to identify the intrusion in real time. To evaluate the results in [29], authors have compared Recurrent Neural Multi- Layer Perceptron (MLP), Network(RNN), and Alternate Decision Tree (ADT) classifiers. Among these three, ADT classifier performs better compared to others. Therefore, we have compared Accuracy, F1-Score, Precision and Recall of this model with DTA-DS KNN classifier. In [29], authors have conducted the experiment in ryu controller with tree type topology with depth value four (4) and fanout value eight (8) in the Mininet. In order to compare the results, we have emulated the experiment in the same type of topology. Emulation was conducted for 300 seconds for both approaches and results of both approaches are compared.

As shown in figure 13 , DTA-DS gave higher F1-Score than IoT based framework after 20 seconds where the emulation was run for 300 seconds. In the interval of 40 to 150 seconds DTA-DS gave a slightly lower F1-Score compared to the IoT base framework. Afterwards, DTA-DS gave a continuous higher F1-Score than IoT based framework. Comparative high values of the F1-score show classifier is more robust and precise to identify the poisonous switches in the network.

As shown in figure 14, the accuracy of the DAT-DS is higher than the IoT base framework up to 20 seconds. Between 25 to 60 seconds, IoT based framework has slightly higher accuracy. After 90 seconds it is observed that DTA-DS has a higher accuracy compared to the IoT base framework. This shows that the model classification rate for poisonous is high out of the total given sample inputs compare to the other model.

In the figure 15, initially DAT-DS have a higher precision compared to IoT base framework. Between 30 to 150 seconds IoT base framework observed higher precision. After 180 seconds it is observed that DTA-DS and IoT Base frameworks are running at the same precision score. A higher precision score indicates that the classifier identified the poisonous switches and make correct annotations for identification from poisonous switch class(class type - 1. poisonous switch and 2. normal switch) compared to another model.

In the figure 16, it is observed that initially at about 10 seconds DTA-DS has a higher recall value. Between the interval of 15 to 120 seconds, IoT base framework has a higher recall value compare to DTA-DS. It has been observed that after 120 seconds DTA-DS approx 100 percent recall value during the emulation time. This means the classifier correctly identified poisonous switches out of total incorrectly identified normal switch as poisonous.

5. Conclusion

This research is a part of the future work of SDN-RM approach in which we planned to test the resiliency of the SDN-RM in the presence of attack. Therefore, in this work, we first implement a the time based delay attack on the SDN-RM. In the next phase, we have presented a delay-based time attack detection system(DTA-DS) in which we have detected a time-based delay attack using machine learning classifiers. In the time based delay attack, attacker delay LLDP packets at the particular switch which cause the reduction in the resiliency of the SDN-RM. We observe a reduction in success rate and throughput in the network under attack especially when the number of attacker switches increases in the network. Also, it is necessary to detect this type of attack which has a major impact on the resiliency of the SDN-RM. Therefore, we have developed a delay-based time attack detection system(DTA-DS) to detect this type of attack. DTA-DS is based on KNN classifier to detect the poisonous switches in the network. We have compared KNN model with the other models in terms of different evaluation metrics in physical and mininet virtual topology. It is evident from the evaluation metrics comparison results that the KNN model gives better results than other machine learning models and achieves high accuracy to detect the poisonous switches in the network. In addition to this, we have compared our approach with other existing similar type of Fog-assisted SDN controlled framework having similar type of setup. We have observed that within selected window period of the emulation our approach performs better compare to IoT based framework. In the future, we are also planning one step further to develop the mechanism which helps SDN-RM to prevent this type of attack in real-time to increase the resiliency of the SDN-RM.

References

1. Jhaveri, Rutvij H., Rui Tan, Arvind Easwaran, and Sagar V. Ramani. ,Managing industrial communications delays with software-defined networking. *In 2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*(IEEE, 2019), pp. 1-11.
2. Ndonda, Gorby Kabasele, and Ramin Sadre,A low-delay SDN-based countermeasure to eavesdropping attacks in industrial control systems*Network Function Virtualization and Software Defined Networks (NFV-SDN)* (IEEE Conference,2017), pp.1-7.
3. S. Viswanathan, R. Tan, and D. Yau,Exploiting power grid for accurate and secure clock synchronization *ACM Transactions on Sensor Networks (TOSN), vol 4. no2*,(ACM, 2018).
4. Jhaveri, Rutvij H., Sagar V. Ramani, Gautam Srivastava, Thippa Reddy Gadekallu, and Vaneet Aggarwal, Fault-resilience for bandwidth management in industrial software-defined networks. *IEEE Transactions on Network Science and Engineering 8, no. 4* **27** (2021),pp. 3129-3139.
5. Ndonda, Gorby Kabasele, and Ramin Sadre, A low-delay SDN-based countermeasure to eavesdropping attacks in industrial control systems *Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, (2017), pp. 1-7.
6. Y. Zhang and V. Paxson, Stuxnet worm impact on industrial cyberphysical system security, *in 37th Annual Conference of the IEEE Industrial Electronics Society*, (IEEE, 2011), pp 4490-4494.
7. Y. Li, Z. P. Cai, and H. Xu, LLMP: Exploiting LLDP for Latency Measurement in Software-Defined Data Center Networks, *Journal of Computer Science and Technology*, 33, no. 2 (2018), pp. 277-285.
8. X. Lou, C. Tran, R. Tan, D. Yau, and Z. Kalbarczyk, Assessing and mitigating impact of time delay attack: A case study for power grid frequency control, *in ACM/IEEE International Conference on Cyber- Physical Systems (ICCPS)*, (ACM-2019), pp. 207-216.
9. Jhaveri, Rutvij H., Rui Tan, and Sagar V. Ramani, Real-time QoS-aware Routing Scheme in SDN-based Robotic Cyber-Physical Systems *In 2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)* (IEEE,2019), pp. 18-23.
10. E. Korkmaz, M. Davis, A. Dolgikh, and V. Skormin, Detection and mitigation of time delay injection attacks on industrial control systems with PLCs, *in International Conference on Applied Cryptography and Network Security (ACNS)*, (Springer, Cham, 2017), pp. 62-74.
11. C. Feng, T. Li, and D. Chana, Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks, *in IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, (IEEE, 2017), pp-261-272.
12. Ndonda, Gorby Kabasele, and Ramin Sadre, A low-delay SDN-based countermeasure to eavesdropping attacks in industrial control systems*In 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, (IEEE, 2017), pp. 1-7.
13. Jero, Samuel, Xiangyu Bu, Cristina Nita-Rotaru, Hamed Okhravi, Richard Skowyra, and Sonia Fahmy, BEADS: automated attack discovery in OpenFlow-based SDN systems *In International Symposium on Research in Attacks, Intrusions, and Defenses*, (Springer, Cham, 2017), pp. 311-333.
14. Wang, BWang, Bo, Biwu Fang, Yajun Wang, Hesen Liu, and Yilu Liu, Power system

transient stability assessment based on big data and the core vector machine *IEEE Transactions on Smart Grid*, no.5 (IEEE, 2016), pp. 2561-2570.

15. He, Miao, Junshan Zhang, and Vijay Vittal, Robust online dynamic security assessment using adaptive ensemble decision-tree learning*IEEE Transactions on Power systems 28*, no. 4, (IEEE, 2013), pp. 2013.
16. Sidhu, Tarlochan S., and Lan Cui, Contingency screening for steady-state security analysis by using FFT and artificial neural networks *IEEE Transactions on Power Systems 15*, no. 1, (IEEE, 2000), pp. 421-426.
17. Santos, Dorabella, Amaro de Sousa, and Carmen Mas Machuca, Robust SDN controller placement to malicious node attacks*In 2018 21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, (IEEE, 2018), pp. 1-8.
18. Sargolzaei, Arman, Kang K. Yen, Mohamed N. Abdelghani, Saman Sargolzaei, and Bogdan Carbunar,Resilient design of networked control systems under time delay switch attacks, application in smart grid*IEEE Access 5* (IEEE,2017), pp. 15901-15912.
19.
20. Liu, Xuan, Zhiyi Li, and Zuyi Li. , Optimal protection strategy against false data injection attacks in power systems, *IEEE Transactions on Smart Grid 8*, no. 4, (IEEE, 2016), pp. 1802-1810.
21. L. Tan, K. Yu, F. Ming, X. Cheng, G. Srivastava, Secure and Resilient Artificial Intelligence of Things: a HoneyNet Approach for Threat Detection and Situational Awareness,*IEEE Transactions on Network Science and Engineering*, vol. 9, (IEEE, 2022), pp. 271-281.
22. L. Zhen, Y. Zhang, K. Yu, N. Kumar, A. Barnawi and Y. Xie, Early Collision Detection for Massive Random Access in Satellite-Based Internet of Things,*IEEE Transactions on Vehicular Technology*, vol. 70, no. 5,(IEEE,2021), pp. 5184-5189.
23. D. Wang, Y. He, K. Yu, G. Srivastava, L. Nie, R. Zhang, Delay Sensitive Secure NOMA Transmission for Hierarchical HAP-LAP Medical-care IoT Networks, *IEEE Transactions on Industrial Informatics*, (IEEE, 2021), 10.1109/TII.2021.3117263.
24. K. Yu et al., Securing Critical Infrastructures: Deep-Learning-Based Threat Detection in IIoT*IEEE Communications Magazine*, vol. 59, no. 10,(IEEE, 2021), pp. 76-82.
25. K. Yu, Z. Guo, Y. Shen, W. Wang, J. C. Lin, T. Sato, Secure Artificial Intelligence of Things for Implicit Group Recommendations,*IEEE Internet of Things Journal*, (IEEE, 2021).
26. Cao, Jie, Da Wang, Zhaoyang Qu, Mingshi Cui, Pengcheng Xu, Kai Xue, and Kewei Hu, A novel false data injection attack detection model of the cyber-physical power system, *IEEE Access,* (IEEE, 2020), pp. 95109-95125.
27. Jemmali, Mahdi, M. Denden, Wadii Boulila, Rutvij H. Jhaveri, Gautam Srivastava, and Thippa Reddy Gadekallu, A Novel Model Based on Window-Pass Preferences for Data-Emergency-Aware Scheduling in Computer Networks, *IEEE Transactions on Industrial Informatics*, (IEEE,2022).
28. Wang, Han, Xingwang Li, Rutvij H. Jhaveri, Thippa Reddy Gadekallu, Mingfu Zhu, Tariq Ahamed Ahanger, and Sunder Ali Khowaja, Sparse Bayesian learning based channel estimation in FBMC/OQAM industrial IoT networks *Computer Communications*, vol. 176, (Elsevier, 2021), pp. 40-45.
29. Shafi, Qaisar, Abdul Basit, Saad Qaisar, Abigail Koay, and Ian Welch, Fog-assisted SDN controlled framework for enduring anomaly detection in an IoT network *IEEE Access*, no. 6, (IEEE, 2018), pp. 73713-73723.