Cluster Computing
https://doi.org/10.1007/s10586-021-03309-0

A novel mechanism to handle address spoofing attacks in SDN based IoT

Hamza Aldabbas[1] · Rashid Amin[2]

Received: 26 November 2020 / Revised: 16 February 2021 / Accepted: 24 May 2021

Abstract
The Internet of Things (IoT) is a network of devices (servers, sensors, nodes, and so on) used to conduct tasks like health monitoring, production monitoring, business transactions, etc. In IoT, the traditional networking paradigm in which the control and data planes are vertically integrated is utilized to link various types of networks. Software Defined Networking (SDN) is a relatively new concept that separates the control plane from the data plane, making network management and maintenance easier. In SDN, network operators prioritise the security of the overall system. The most severe attacks on systems target the Address Resolution Protocol (ARP), which then frequently act as a springboard for more complicated attacks. This paper proposes a secure SDN-based IoT architecture to manage and reduce ARP spoofing attacks by deploying a new machine near the SDN controller to handle address resolution questions. To examine address spoofing threats, we move ARP traffic to this new machine. This module works in tandem with the controller, gathering topology data and ARP requests in order to detect potential attack conditions. The ARP data is analyzed using custom methods. According to simulation results, the proposed technique increases network throughput, improves attack detection and mitigation time by 35% over existing techniques.

Keywords ARP spoofing · Internet of things (IoT) · Network security · Port blocking · SDN based IoT

1 Introduction

The Internet is a huge data network that connects billions of devices, people, and things across wired and wireless networks and mobile [1]. A range of threats is aimed against these persons and network resources. Communication security, data protection, user privacy, and other related issues have been important problems since the birth of data sharing and networking [2]. The internet of things (IoT) is a framework that allows several network devices to detect, communicate, gather, assess, and make decisions according to critical analyses and forecasts, as shown in Fig. 1. In IoT, several end-user devices are connected through traditional networking devices, i.e., switches, hubs, routers, that may be severely affected by different types of attacks, e.g., Man in the Middle attack, link flooding, DoS, packet spoofing [3]. Due to several devices in IoT, a massive amount of traffic is generated that causes network congestion. Sometimes, this congestion is due to the broadcast storm of ARP packets. Some other attacks, i.e., DoS, Brute force, Zero-day attack, Browser, Backdoor, and Botnet, are the highly-rated attacks lowering the trustworthiness and proficiency of the entire network [4, 5].

Software defined networking (SDN) [34] is the new paradigm that handles most traditional networking limitations. It evolved as a game-changer that reformed the basic networking phenomena, and it runs as the backbone of almost all networking applications. SDN separates the network control and management planes from the data plane by providing low-level operations to the network operators [5]. Through these low-level functionalities, network devices and services are managed easily [6]. In this way, it extends the support for scalable and dynamic

✉ Hamza Aldabbas
Aldabbas@bau.edu.jo

Rashid Amin
rashid.sdn1@gmail.com

[1] Software Engineering Department, Prince Abdullah Bin Ghazi Faculty of Information and Communication Technology, Al-Balqa Applied University, Al-Salt, Jordan

[2] University of Engineering and Technology, Taxila, Pakistan

Published online: 01 June 2021

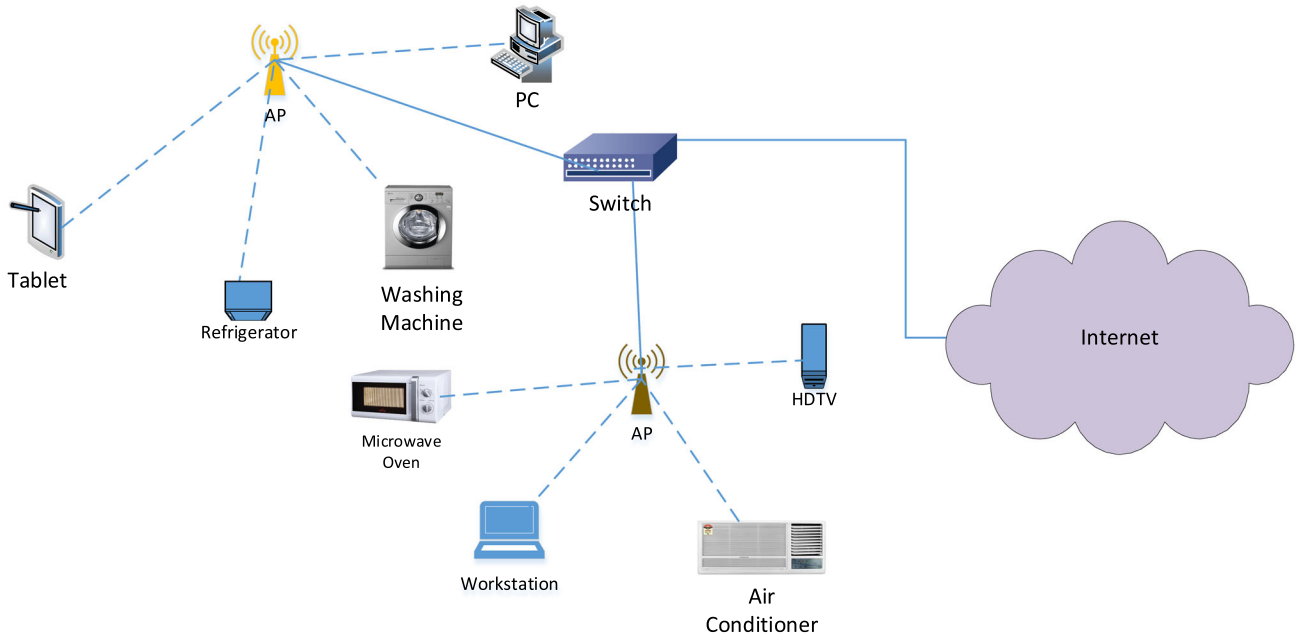


Fig. 1 IoT network containing different types of devices

computing needs for modern complex networks and minimizes the network by allowing adaptive management and control. In the plane's separation of SDN, the only data plane is left with the networking devices, while control and management planes are moved to the controller [7], as shown in Fig. 2. Different types of applications, e.g., routing logics, network policies, management tools, security measures, are deployed at a centralized controller.

Regarding network security issues [8], SDN resolves several traditional networking issues by providing centralized control. SDN offers more flexibility, data flow optimizations, consistency, etc. Compared to manual configurations of traditional networking [7]. SDN-based IoT mainly refers to the use of SDN to facilitate high data transmission in the IoT environment. It can also help to efficiently allocate and manage network resources and fulfil the growth of data demands, users, and devices. This merger is expected to overcome different IoT issues that still exist in a traditional network.

SDN-based IoT [9] are secured by deploying security applications at the SDN controller that secure the users from different types of network attacks. Most famous network attacks, e.g., distributed denial of service (DDoS) and Link Flooding Attack (LFA) [10], are launched by using internet protocol (IP) spoofing or address resolution protocol (ARP) spoofing [11] method. IP or MAC address translation is done by ARP packets in the network used to identify users or systems. An attacker can easily modify these ARP packets to change the IP or MAC address of any system. SDN Controller is the main component in the SDN-based IoT network, a single point of failure, and prone to many attacks. The most common attack is the ARP spoofing attack, in which ARP spoofing packets are initiated by any malicious node (attacker). These packets cause congestion and poison the network topology [12]. Sometimes, these attacks take the lead to the denial of services, service interruption, serious hijacking. Moreover, besides ARP spoofing, link flooding attack, ARP cache poisoning, and Man in the Middle [13], attacks have also been believed with a brief narrative of further indexes.

Several mechanisms are adopted to secure the IoT from different types of attacks [14–18]. Similarly, a different solution has been proposed for SDN-based networks [19]. In this research work, we are dealing with the IoT infrastructure based on the SDN paradigm. IoT based on the traditional network is more susceptible to several types of attacks, while SDN provides more control over the entire network. An SDN based address spoofing identification and prevention mechanism is proposed. When the network is under attack, the controller cannot execute further at the controller port due to continuous packets. Mininet used to insert numerous SDN switches, hosts, and controller under our scenario [20]. Mininet Simulation tool is utilized to make some virtual surroundings where various tests are conducted by utilizing the Pox controller [21].

In [22], the authors presented a machine learning approach to detect ARP spoofing attacks. They tried to minimize the false-positive ratio of DoS attack detection of SDN. In this new system, flow-based and packet-based approaches are combined to enhance system performance.

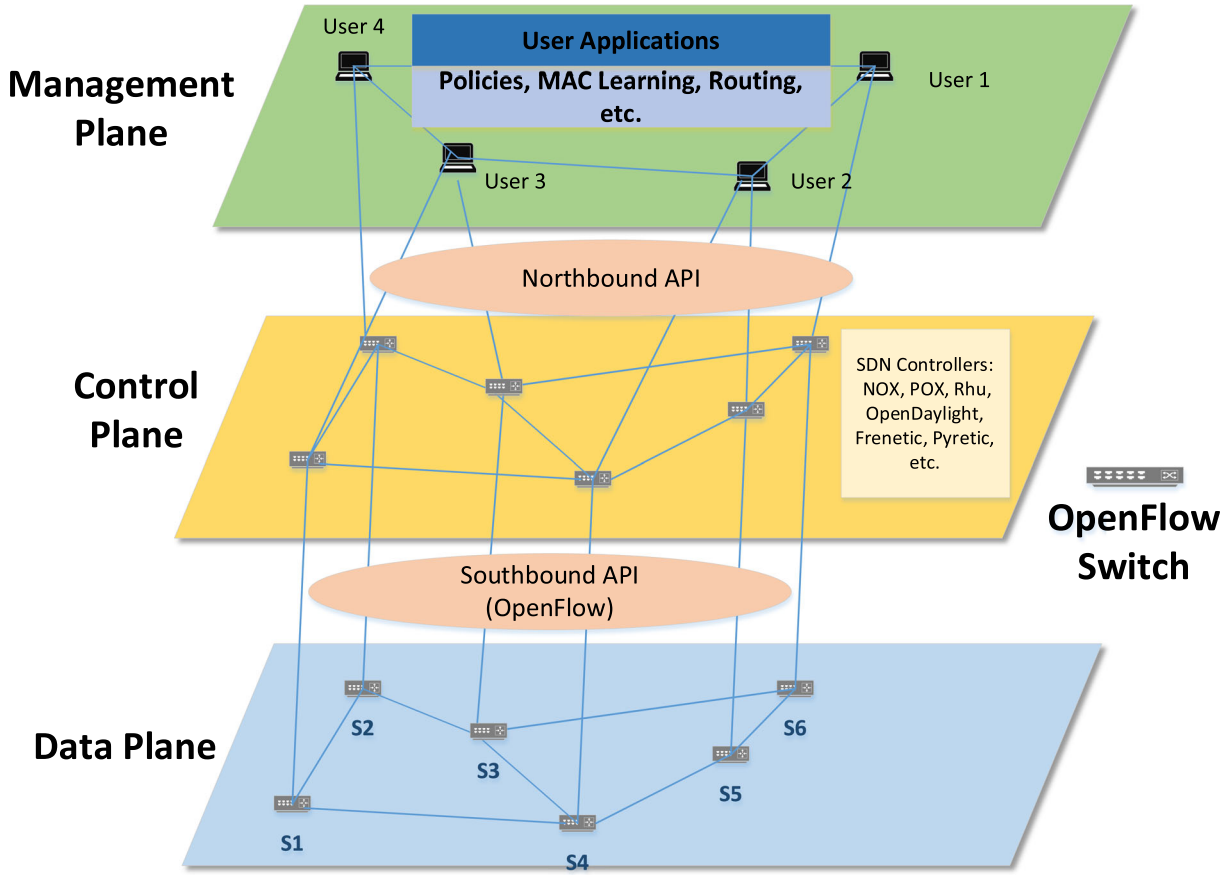


Fig. 2 Layered architecture of SDN

K-nearest neighbor (kNN) and neural networks (NNs) algorithms are adopted to train and test the data set. The dataset NSL-KDD is used for experimentations. This approach relies on the theoretical concepts rather than the proper implementation of the approach. SeArch [19] is an approach used to detect network intrusion in SDN based cloud IoT. It uses a layering architecture of intrusion detection systems (IDS) to detect and mitigate anomalies through SDN-based gateway devices. A deep learning-based approach is used for the detection of anomalies. It works well in a cloud-based environment, but it may face problems for a simple IoT system.

In order to cope with these limitations, a novel mechanism that automatically detects and prevents ARP spoofing attacks is proposed. The ARP packets are forwarded to an individual server instead of the SDN controller to deal with malicious packets. At this server, packets are analyzed for possible attack conditions using custom applications. The server is programmed with the proposed algorithm to check the packets against the host. If the ARP packets belong to the same network, then suitable action is taken; otherwise, the packets are dropped. Moreover, after the detection of malicious traffic, the specific ports that are involved in being blocked. In this way, the SDN controller is spared from unnecessary processing that results in resourceful and protected usage of the network resource. Our main contributions in this paper are as follows:

- A novel system is proposed to automatically install rules on the network devices to transfer the ARP traffic to the proposed server.
- At the proposed server, packets are analyzed to identify the potential threat condition.
- After identification of threats, proper steps are taken to block the respective ports.
- Simulation results indicate that the proposed approach performs better than existing methods.

The remaining contents are organized as follows: "Related work" discusses the related work, and "Problem statement" describes the IoT infrastructure, while SDN-based IoT infrastructure is presented in "Proposed solution". Problem statements and proposed solutions are elaborated in "Problem statement" and "Proposed solution", respectively. "Performance evaluation of the proposed scheme" offers the performance evaluation, and the conclusion is included in "Conclusion".

2 Related work

SDN is a new paradigm that offers flexibility and easiness for network control and management. Sood et al. [23] and Caraguay et al. [24] deliberated IoT challenges and opportunities based on the SDN paradigm. It is observed that SDN-based skills and techniques have a significant impact on making IoT more successful and reliable. SDN provides several features like network management, Network Function Virtualization (NFV), accessing information from multiple resources, Efficient resource utilization, energy management, security and privacy, and many more.

Xie et al. [25] surveyed machine learning (ML) algorithms for SDN security and then discussed how these ML algorithms are used in SDN in security, traffic classification, resource optimization, and QoS prediction and also describe an application of ML algorithms. According to this survey ML techniques bring intelligence to the SDN environment and make the SDN controller more powerful to make optimal decisions. This survey figures out how ML algorithms are implemented to solve SDN problems. The method proposed by Xu et al. in [26] is a K-FKNN-based DDoS detection system for the SDN. The control plane's flow controller collects the network flow data from switches and sends information to the feature extractor. Here it extracts features to create a feature vector of the network. This feature data includes data in the form of bytes sent from source to target and source. Later send this vector to the application for detection. In the database, training data is stored. This training data consists of normal and DDoS attack data. The K-means + + algorithm is used to preprocess the training data. This data is then utilized for DDoS detection applications. Application Plane consists of K-FKNN initializer, K-FKNN based detector, and DDoS Mitigation. K-FKNN Initializer is used to set initial values of parameters that are used in the K-FKNN algorithm. K-FKNN detector receives vector data from the control plane. It normalizes this vector's feature data using the K-FKNN algorithm, preprocessing the data to identify DDoS attacks. DDoS Mitigation is used to detect network flow. If the flow is detected as a DDoS attack, it asks the controller to drop the flow table's action entry. This flow entry is sent to the switch by the controller, and switches drop the network's malicious flow to protect the network.

Ubaid et al. [27] present a security model for incremental SDN. In this approach, a mixed network is considered in which SDN and non SDN devices are deployed in the network. When the network is established, then network topology is obtained by using the LDAP protocol. As there is a mix of devices, SDN devices can be operated directly by the controller, while non SDN devices are operated through SDN devices. So special instructions are used to configure these devices to install flow rules. When rules have been installed, then ARP traffic is made possible to arrive on the additional entity to analyze whether it contains some harmful information or not. If there is some attack on the network, the respective port is blocked to save the entire network.

Shafi et al. [11] describe the effect of DDoS and botnet attacks on the network. These attacks, a botnet prevention system, are IoT designed using SDN and distributed blockchain (DBC) to secure the network form. In this proposed work SDN controller is connected in a distributed manner similar to the blockchain network. It is possible to share authentication information among all of these controllers. These blockchain-enabled controllers can update the flow rules by authenticating a verified version of flow rules. The updated flow rules are installed on the switches to block the unauthenticated exchange of data. Whenever an unauthenticated data is shared among the controllers, there is a possibility of a DDoS attack on the network. By using authenticated flow rules, botnet and DDoS attacks can be prevented.

Wang et al. [28] discuss a new type of DDoS attack called Link Flooding Attack and provides an enhanced mechanism to mitigate this type of attack. When a DDoS attack is launched, then authorized users do not have access to the network resources like bandwidth, file, server, etc. For this purpose, the attacker launches multiple attacks on the network server that make the server very busy. Subsequently, it cannot answer other user requests. Similarly, LFA is also another type of DDoS attack that affects many users by propagating different types of malicious requests. In this way, the server cannot differentiate the traffic from the legitimate user and malicious user. A new scheme called woodpecker is introduced to handle these attacks that try LFA not to take effect. A proactive probe approach is applied to indicate the congested links and measure LFA's probability instantly. Woodpecker uses a centralized traffic engineering algorithm that does not allow the bottleneck for routing nodes. The implementation of SDN Data Centers is shown in Fig. 3.

In [29], the author proposed a machine learning-based system for DoS attack detection in IoT networks. They used averaged one-dependence and two-dependence methods to detect the attack by integrating MultiScheme and voting schemes. The data is captured from the network traffic; data is preprocessed and labeled according to different criteria. Features are selected using famous algorithms, i.e., chai-square, info-gain, then packets are classified using AIDE, A2DE, Bayesian network, etc. algorithms. Malicious packets are captured based on these features.

From the above approaches, it is clear that the existing methods do not satisfactorily address ARP spoofing

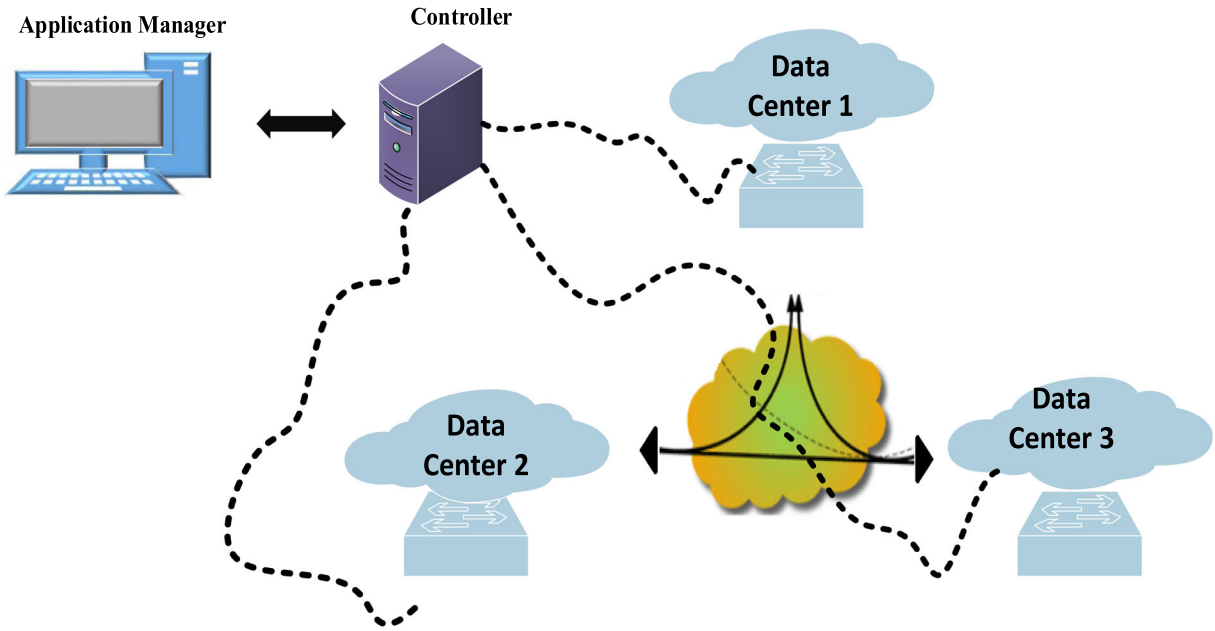


Fig. 3 ARP control mechanism in data centers

attacks. Mostly, when an attack is launched, the SDN controller becomes so much busy to answer any query from other users. So it is necessary to spare the SDN controller from the malicious traffic so that it can perform its duties proficiently and efficiently.

3 Problem statement

IoT is a combination of different types of networks, and it performs several types of tasks like monitoring, measuring, analyzing, classifying, etc. In order to achieve these operations, network devices connect via wired or wireless links. Whenever a node wishes to connect to the network, it is assigned an IP address, and when it wants to share some information with another node, it gets its IP/MAC address. ARP spoofing attack is launched in the network to control or make the controller busy, not to answer other user's requests. When these attacks are launched, an authorized user can access the data packets, and the Controller ARP Table (CAT) may be septic with wrong entries in the CAT table. This situation may lead to a situation when a network is hijacked. Moreover, this also results in stumpy network performance and leads to more attacks like a man in the middle attack (MITM). Thus, the entire network may face severe threats, as shown through examples in related work.

The controller is the main component in SDN-based IoT, and it is more susceptible to different types of security attacks. If an address spoofing attack is successful, then the whole network topology information is poisoned. With such a poisoned network topology, the SDN controller cannot properly install flow rules on network devices. IoT basic building block is also affected by this poisoned topology information that has influenced different applications and services running on the SDN controller. Some of the applications may be misconfigured, and subsequent users suffer from this situation. In this situation, the attacker can control the network resource, which leads to hijacking, DOS attack, and complete network failure. Different SDN controllers like Floodlight, POX, Beacon, etc., are affected by these types of attacks, as it is narrated in several research papers. Different research studies conclude that various SDN controllers (e.g., Floodlight, Beacon, NOX, and POX) are afflicted by these types of attacks [30].

An example scenario of an enterprise IoT network determines the problem in detail shown in Fig. 4. Suppose four Openflow (OF) switches, i.e., l1, l2, l3, and l4, and two wireless access points (AP1 and AP2) that are also OF enabled. These switches are connected, as shown in the Figure. The switch l1 is connected to AP1, and switch l3 is connected to AP2. Different home appliances (i.e., washing machines, HDTV) and smart devices are connected to the APs, forming an IoT network. Initially, the IoT network operates using legacy devices, but these devices are replaced with SDN devices, and an SDN controller is also placed in the network. Any path P from the source device to the destination device's path through the Openflow switches. Two possible conditions of this IoT network are discussed. The first one is an ideal condition; all the

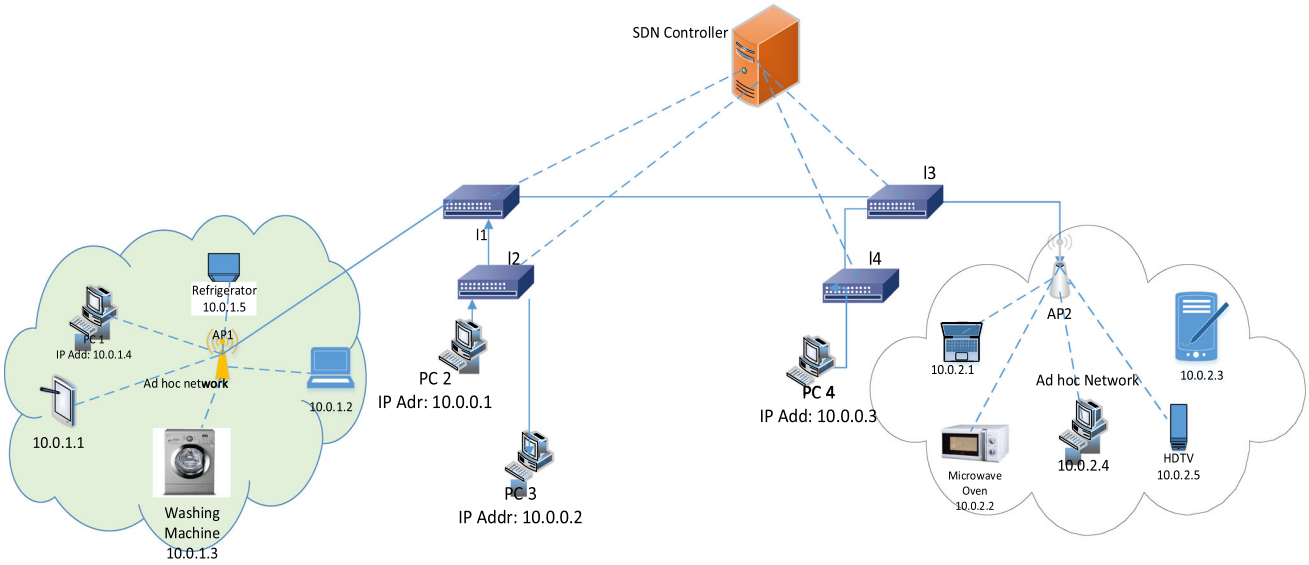


Fig. 4 IoT network scenario

operations are usually performed in this IoT network. The second, when an attacker launches an attack on the IoT.

3.1 Normal condition

A normal condition of a smooth IoT network processing is shown in Fig. 4. For example, a user on PC2 having IP Address 10.0.0.1 wishes to connect to another user having IP Address 10.0.2.4, and the MAC address is not known. To know the MAC address of 10.0.2.4, PC2 sends the ARP request packet to SDN switch l2. SDN switch checks its flow entries to find the respective MAC address; if a respective flow entry is not found, packets are forwarded to the SDN controller. SDN controller finds the respective MAC address and returns it to the switch where the PC2 is connected. For communication purposes, the controller also installs the rules for this route on the respective switches. By using this methodology, the normal operation of this IoT infrastructure is working fine.

3.2 When IoT network is under an attack

In the above section, the ideal operation of an IoT is elaborated. When an attacker wishes to attack the IoT network using ARP packets, the attacker uses an IP address of some other known node. The attacker mostly uses a Kali Linux operating system to attack the network by sending a broadcast message of Gratuitous ARP packets having the IP address of PC2, i.e., 10.0.2.4. These Gratuitous ARP broadcast messages are used to announce any modification in the IPv4 address of PC2. When these packets are reached on the switches, then they update their entries for this update. In this way, the attacker can capture all the traffic of PC2 with hijacking tools installed on the Kali Linux system. In our case, Kali Linux OS is used to launch an ARP spoofing attack, leading to a Man in the Middle attack. From Fig. 5, IoT has been hijacked by a single conventional attack if the controller's proper configuration is not done carefully. When this attack happens, the network topology information stored at the SDN controller got infected. Using this technique, an attacker can capture the network as much as it wants to control the entire network. Kali Linux user having IP address 10.0.2.4 of a user can easily launch DDoS attack to busy the server. In this situation, the server remains busy with the attacker, and other users cannot access the resources. The controller is completely under the control of an attacker, and other users are waiting for their turn that will never happen. Consequently, the performance of the controller is degraded, and users have no access to legitimate resources. This condition may result in potential business loss, interference in the business operations, and damage to the client's confidence.

4 Proposed solution

A comprehensive mechanism is needed to tackle all the ARP spoofing issues in the SDN-based IoT system to address these problems. To handle the issue of address spoofing, we proposed a mechanism that automatically detects ARP attacks in an IoT and mitigates the attacks using an individual machine running the custom code. The SDN controller maintains the topology information of the network and Controller ARP Table (CAT). The network topology information is fetched from the controller and from the dynamic host configuration protocol (DHCP)

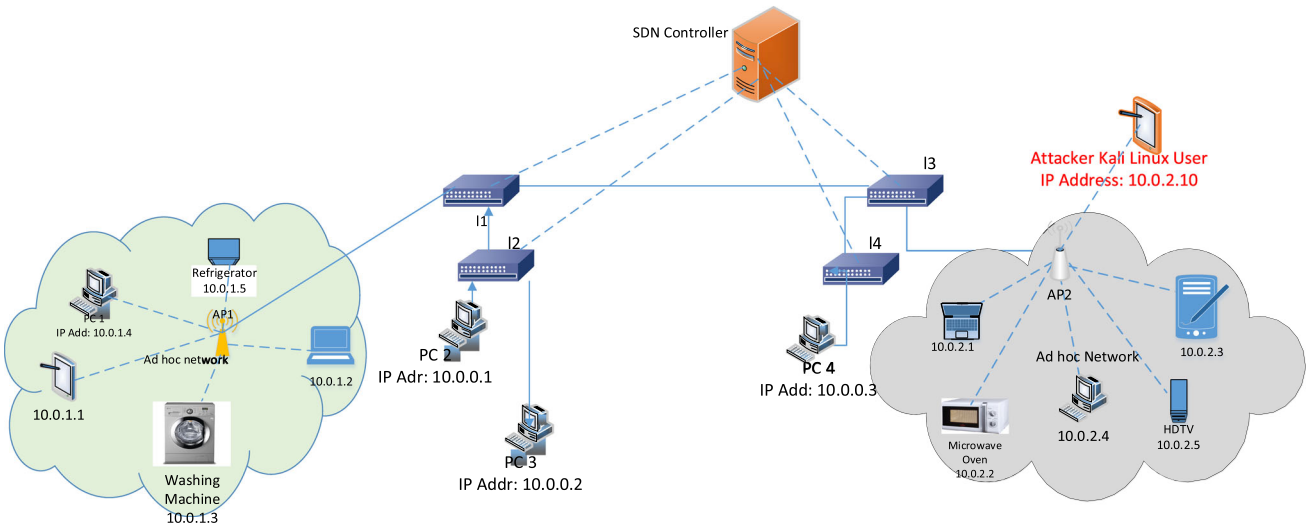


Fig. 5 When IoT is under attack

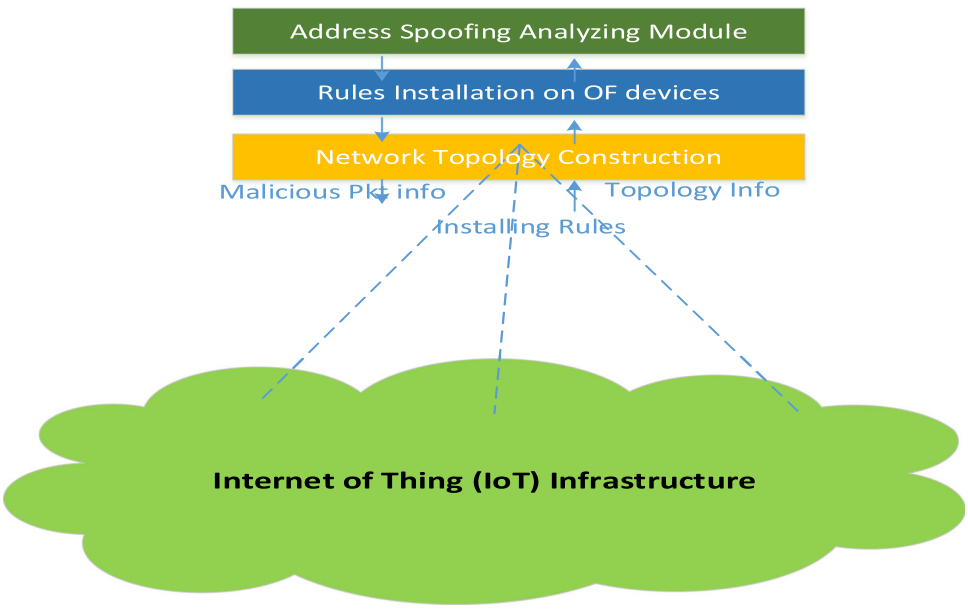
server at the proposed machine to analyze the malicious traffic.

4.1 System design

Figure 6 shows the overall system design for an automatic address spoofing detection and mitigation for SDN-based IoT. This system consists of a separate machine used to deal with the ARP requests because ARP traffic arrives on the controller; then, ports will be blocked, and further processing will not happen on the controller. There are three main components of this new mechanism. Network topology information is fetched from the entire network by the first components of the proposed solution. The second part deals with the installation of flow rules on the network devices so that all the ARP traffic must reach a separate machine. The third component contains traffic analyzing and mitigating ARP spoofing modules. The entire system design is shown in Fig. 8, where the underlying network is an SDN based IoT network.

Network topology information is captured at the proposed machine from the SDN controller and the network devices. SDN devices exchange their connection information with the machine after a fixed interval of time. When topology information is fetched, flow rules are installed on the switches according to flows' requirements. The

Fig. 6 Overall system design



proactive rules are installed to forward APR-related traffic to the machine. After setting these flow rules, network traffic is fetched on the machine, where further processing and analysis is performed for possible address spoofing attacks.

From the network topology of the entire IoT network, graphs describe the network's overall layout. Packets from the host are checked to detect an ARP spoofing attack. As a first step, it is checked that ARP packets belong to this network or not. For the next step, it is checked whether this ARP request goes around this network or not. If the ARP request is from the current network, then required information is shared among two parties, and flow rules are installed. Furthermore, It is explained in the following scenarios.

4.2 Flow rules installation on openflow switches

On the SDN devices, flow rules are installed after getting topology information from all SDN switches. These rules are installed to direct ARP traffic to the proposed machine, where further analysis can be performed to avoid address spoofing. The controller operates special commands to instruct the switches about this task to install these rules. Following algorithm 1 explains the procedure for flow installation.

Algorithm 1: Flow rules installation

Input: N number of Nodes and L number of switches, R rules, P data packets

Output: Path Q to the proposed machine

Get L switches connectivity

R rules (flows) on k < = L switches

if (P ε ARP class) || (P.dest = = Broadcast)

P.dest = = PC_{new}

else

P.dest = = SDN conroller || (moved on Q)

end if

4.2.1 Analysing ARP traffic for possible attack condition

When flows are installed on the SDN switches to direct the ARP traffic to the proposed machine, the graph is generated based on the network topology information. This graph can be used to locate the position of the attacker in the network. The packets from a particular host are selected for analysis to check an attack condition. After analysis, it is confirmed that ARP packets belong to this network or from some other network. If the ARP request belongs to the corresponding network, then appropriate action is taken. It is explained in the following cases:

In the first case, when any user in the network launches an ARP packet, then the corresponding switch receives this ARP request and checks its flow table for the required information. If required ARP information is not found in the switch's flow table, it is forwarded to the proposed machine. In addition to this, it is confirmed that this packet belongs to the current network or not. If it does not belong to this network, then it will be dropped. Otherwise, further processing is performed. The custom algorithms installed on the machine checks the packets for possible attack condition. The following example explains the procedure in detail. Suppose PC2 having an IP address of 10.0.0.1 creates an ARP packet to communicate with user IP address 10.0.2.4 as shown in Fig. 7. ARP packets are reached at the SDN switch that is forwarded to the proposed machine. This machine confirms that the packet belongs to this network by checking the topology table. ARP replay message is sent to the ARP generator with the MAC address of user IP address 10.0.2.4. If this packet does not belong to this network, it is immediately dropped because these packets can launch an attack on the network.

For the second case, an attacker may generate ARP packets by pretending like a legitimate user having the IP address of any other user like IP address, i.e., 10.0.2.4. These ARP packets are the same as these network packets; after reaching the proposed machine, they are analyzed for possible attack conditions. First, the sender IP address is checked from the previously recorded IP mapped table. If an entry is found for this packet, then this MAC address is matched to IP to MAC table, and for this case, it will be true, so packets are discarded. If several ARP requests are generated from this node, then the corresponding port where this node is attached is permanently blocked. In this network is saved from ARP attacks, and the network operates smoothly.

Algorithm 2: Identification of address spoofing attack

Input: P = {p1,p2,…,pn] packets, P_{arp} ARP Request/Reply/Response Packets, B Broadcast Addres, N number of nodes, R rules, L switches

Output: verified network attack

1: Initialize Tab (ARP table)

2: while(I < = N)

3: Add IP/MAC to Tab

4: end while

5: if (p.src $\notin$ Tab && p.dest $\notin$ Tab)

6: discard p

7: else

8: if (p.dest exist)

9: p.dest $\in$ IP/MAC in Tab

10: send IP/MAC address

11: end if

12: Else if (p.dest = = B && p.src $\in$ Tab)

13: Install R on L for B

14: endif

15:end if

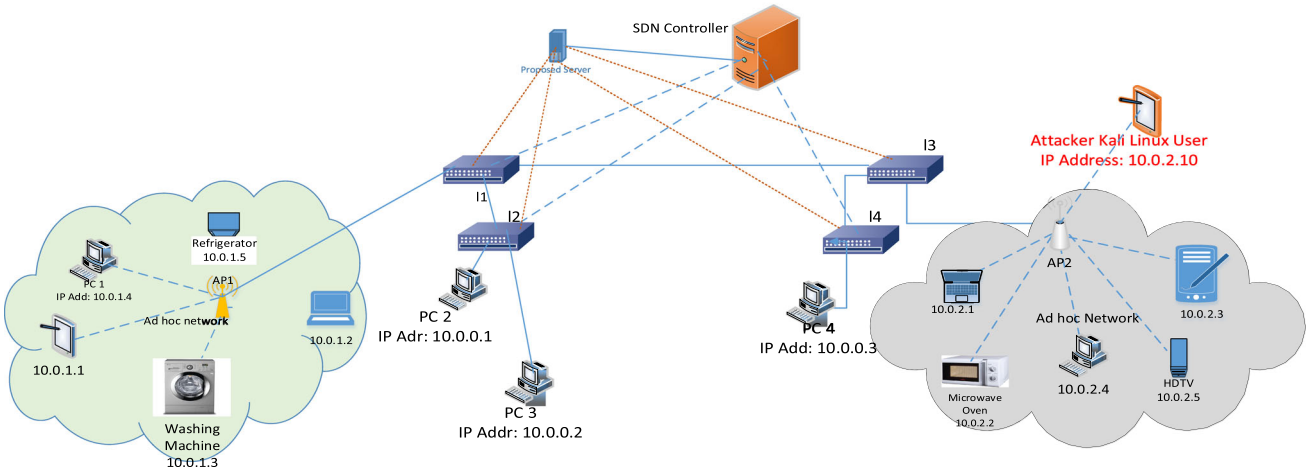


Fig. 7 Proposed solution architecture

A grouping of the host's data is used to match the hosts' address spoofing activity. Although there are certain estimation states where spoofing can be detected, it also performs the cheap computation of the probabilities for an attack condition by enabling a threshold. One can use these probabilities for ranking the hosts how likely ARP spoofing attacks influence these. Using these statistics, a network administrator can easily detect the spoofing attack, and subsequent action can be performed.

4.3 Flow diagram for the proposed approach

Figure 8 is a flow diagram that represents the overall operations for ARP spoofing detection and prevention mechanism. It elaborates on the complete procedure from starting to ending for the proposed solution. From this diagram, it is clear that if some node tries to spoof by using the ARP packet, it is detected in the early stages as this processing is done on a separate machine, and the SDN controller is not affected by this attack.

5 Performance evaluation of the proposed scheme

Various experiments are performed using a specific emulated environment for SDN-based IoT to assess the proposed system's performance. Experimentations are performed on Ubuntu VM with six cores and 8 GB RAM with a hypervisor server containing 16 GB RAM. Mininet is a tool widely used for SDN based simulations. This tool is utilized to make some virtual networking settings in which various assessments are performed on Pox [20] SDN controller. In Mininet, a large number of SDN switches, hosts, and a controller are added according to different scenarios, as discussed in the problem description section and the proposed solution section. We used several setups to evaluate the performance by taking 5, 10, 15, 20, 30 switches. The number of nodes varies as 10, 20, 30, 40, 50 for different experiments.

5.1 Simulation scenarios

The proposed basic network topology is revealed in Fig. 6 and Fig. 7, in which five SDN switches, one SDN controller, two wireless APs are deployed with ten user workstations. SDN switches vary from ten to fifty switches, and the number of users to 100 for the experiment. After establishing the network, an attacker is introduced, having IP address 10.0.0.11. A famous Linux OS called "Kali Linux" is installed for network attacks and network intrusion. This new user with Kali Linux tries to make the Address spoofing and other spoofed packets disturb the entire network topology controller. To evaluate the proposed solution's performance, different parameters like attack discovery time, overhead on the CPU, attack mitigation time, and throughput of the system are computed. Different types of attacks, i.e., ARP request and response attack, spoofed ARP request or response attack, and DDoS attacks. We consider three existing approaches X [15], Y [22], and Z [28] for comparison of the proposed approach.

- Attack discovery time is the entire time in which the challenger party launches an attack on the network, and the controller notices it on the system.
- Attack diminishing time is a time to resolve attacks after the detection of the attack on the network.
- CPU utilization: this factor depicts CPU usage when a network runs in normal condition and when an attack is launched.

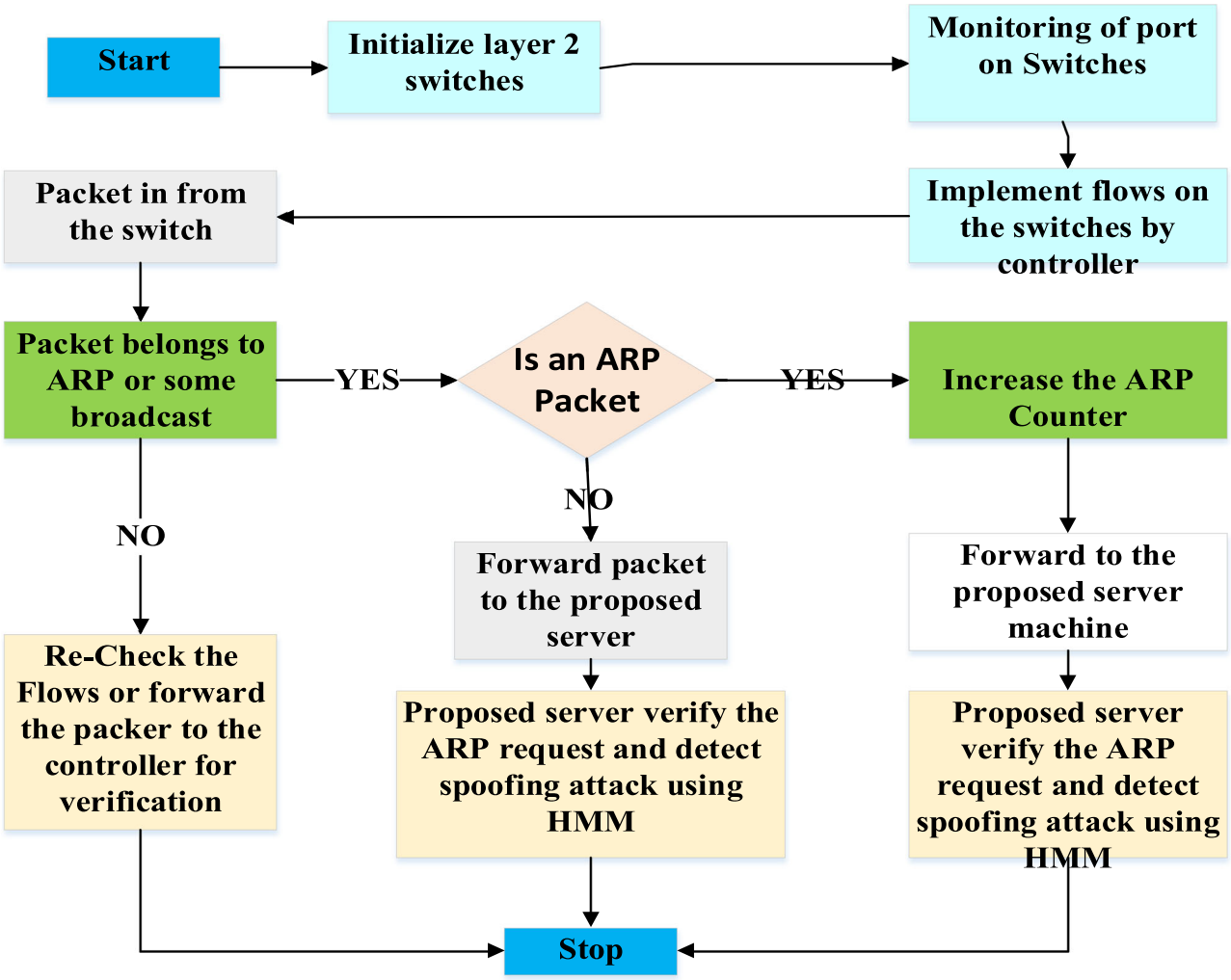


Fig. 8 Flow diagram for attack mitigation

- Packet delivery rate: this tells about the number of packet being transferred when a network attack is launched.
- The throughput of the network represents the number of packets successfully reached at the destination in a given time.

5.2 Discussion on spoofed ARP attack

When the ARP spoofing attack occurred, the victim's cache table was poisoned with an anonymous host's address. This type of attack affects the entire network communication. These attacks can be made by injecting hundreds of false packets of information into the network. Mostly, two types of ARP request influence the network. ARP request packets are initiated using legitimate user's IP addresses in a simple ARP request attack. During this attack, other nodes update their cache with the attacker's IP address. In this way, legitimate users can access the other users, and the adversary party can access the entire network traffic. In order to mitigate these situations, the SDN controller can control these ARP requests by diverting the ARP traffic to the proposed server. On a different server, the processing is performed to filter the attack condition.

5.3 Discussion on DDoS attack

In a DDoS attack, the ARP spoofing technique is used to launch a DDoS attack. This type of attack shuts down or damages the entire network by keeping the server busy for a long time. So, legitimate users unable to get access to network resources. Mostly, this attack is launched by several computers to block the services of the network. In this paper, the proposed approach eliminates the DDoS attack's occurrence by analyzing the suspicious traffic entering the network.

5.4 Attack detection/mitigation time

Attack detection time is the time required to detect the occurrence of an attack, and mitigation time refers to the time in which the system can block the attacker, and normal network operations resumed. Both of these times for the proposed approach and the existing approaches are computed and compared. In Fig. 9, ARP request attack detection times are computed. The existing technique X, Y and Z takes about 25, 30, and 35% more time to detect the attack condition than the proposed approach. One can detect the attack by diverting the traffic towards the proposed server that timely detects the attacks. Similarly, Fig. 10 shows the attack detection time for ARP replay attacks. Although this attack takes more time in all the approaches, i.e., proposed and existing approaches, but the proposed technique has about 30–35% less time than the existing approaches.

In case of attack mitigation time, sometimes the network administrator does not notice the attack condition. It may take a long time for legitimate users to gain access to resources. ARP traffic is examined separately at an individual server in the proposed approach, so ARP attacks are detected automatically, and very soon, attack prevention mechanisms mitigate the situation. Figs. 11 and 12 show the attack mitigation time for both types of attacks, i.e., ARP request attack and ARP replay attack. These figures show that the proposed approach enhances the mitigation time by 30% to 40% than the existing techniques.

5.5 CPU utilization

If a network is under attack, then the SDN controller must process more and more queries as it receives service requests continuously. This situation makes the CPU overburden, and this situation needs to be resolved. the approach detects the attack condition at an early stage. Most ARP traffic is transferred to the proposed server where further analysis is performed to detect malicious. Fig. 13 compares the CPU utilization in both the case, i.e., proposed and existing technique X, Y and Z in terms of ratio in percentage.

The proposed solution is more efficient and effective; that's why CPU utilization of the proposed server is more than the SDN's traditional controller. But let's compared the CPU utilization of the proposed system with the controller of the existing solution. the controller CPU utilization is lower than the controller of the existing solution. Due to all the malicious traffic will be forward to the proposed server.

5.6 Packet delivery rate

Figure 14 presents the significances of successful packet delivery for the time stamp. The results point out that the proposed solution has a higher successful packet delivery ratio as attack conditions do not prevail for a long time. When an attack is launched, the system routinely detects the attack and reduces its consequence on the system. Moreover, one can locate the attack and can block its port permanently. In this way, a large amount of unwanted traffic is minimized. Thus successful packet delivery rate is high.

5.7 Network throughput

The performance of the proposed system is measured by computing the throughput. Throughput represents the

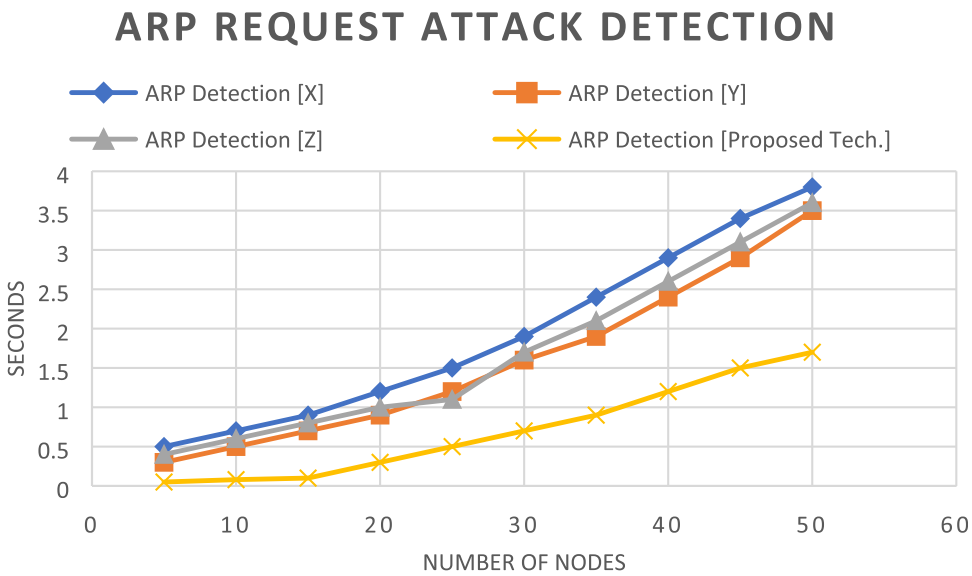


Fig. 9 Time evaluation of ARP request attack detection

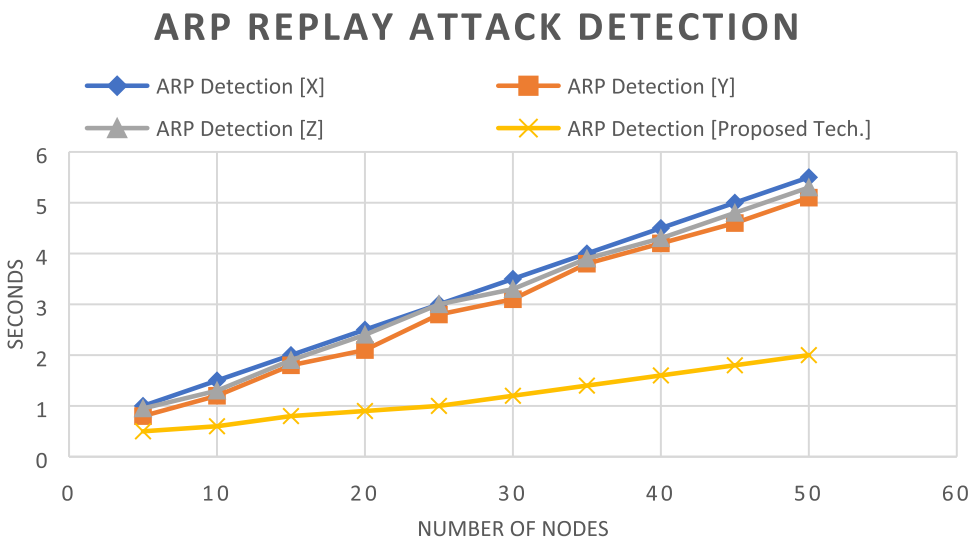


Fig. 10 Time evaluation of ARP replay attack detection

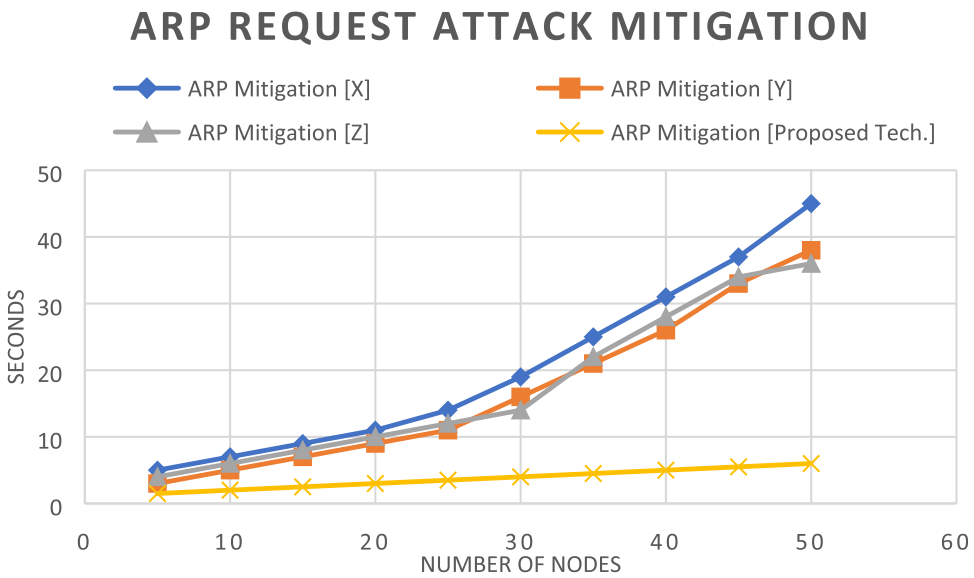


Fig. 11 Time evaluation of ARP request attacks mitigation

overall utilization of the entire IoT network. To compute a link's throughput, the number of packets passing on the link are counted when an attack is launched. Both values of throughputs before and after the attack are compared. The throughput values of the proposed algorithm with the existing approach are compared, as shown in Fig. 15. When an attack is launched in the existing approach, all the resources are stuck because the SDN controller can not service any other request except the attacker. But in the proposed approach, one can identify the intruder in the early stage so network throughput will not be affected.

The maximum network throughput of any network equals the TCP window size divided by the round-trip time of communications data packets.

Throughput = (TCP window size (Data Packets))/ (Round Trip Time (RTT)).

So, according to the proposed scenario, which is described in Fig. 14 before the attacks, throughput can be calculated as:

5.8 Calculating throughput

The calculated throughput For the existing techniques (Before the attack) is determined using.

TCP window size by default: 65,535 bytes or 524,280 bits ∴ defined in IETF RFC 1323.

Round Trip Time: 0.060 s.

To calculate the average of ten packets from host CLI, So according to the equation: -

$$\text{Throughput} = \frac{524280 bits}{0.060 sec}$$

$$= \frac{8738000 x 10^6}{10^6}$$

$$= 8.738 \text{ Mbps}$$

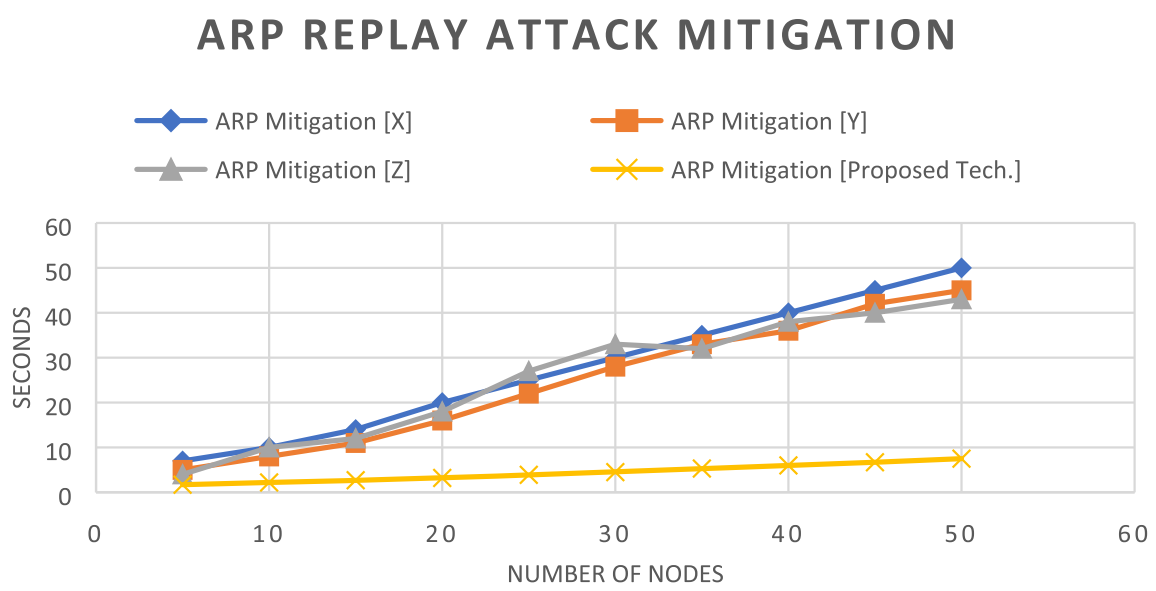


Fig. 12 Time evaluation of ARP replay attack mitigation

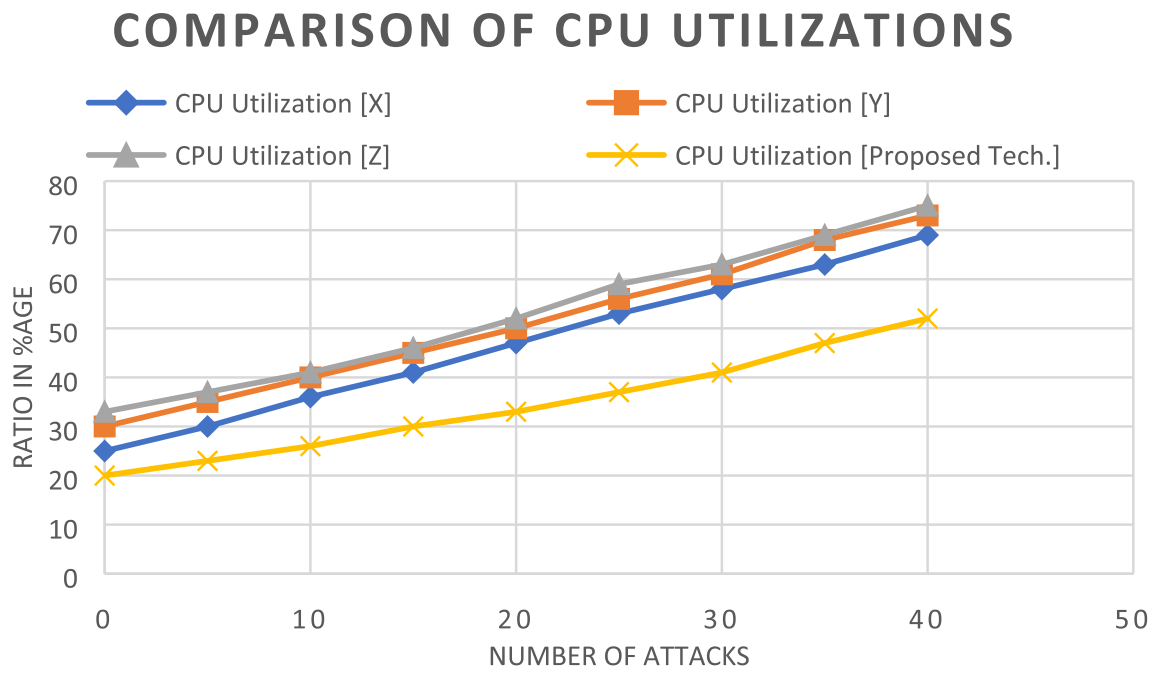


Fig. 13 Comparison of CPU utilization

5.8.1 Throughput for the proposed techniques (before the attack)

TCP window size by default: 65,535 bytes or 524,280 bits
∴ defined in IETF RFC 1323.

Round Trip Time: 0.060 s.

To calculate the average of ten packets from host CLI.

Throughput = $\frac{524280bits}{0.060sec}$

= $\frac{8738000x10^6}{10^6}$

= 8.738 Mbps

5.8.2 Throughput during the attack

To calculate the throughput For the existing techniques (During the attack) is determined using.

TCP window size: 0 bits ∴ cannot transmit any data due to large request from the DDoS attack.

Round Trip Time: a number which is near to infinity to get the answer in real numbers.

Throughput = $\frac{0bits}{alargenumberinsec}$

= $\frac{0x10^6}{10^6}$

= 0 Mbps

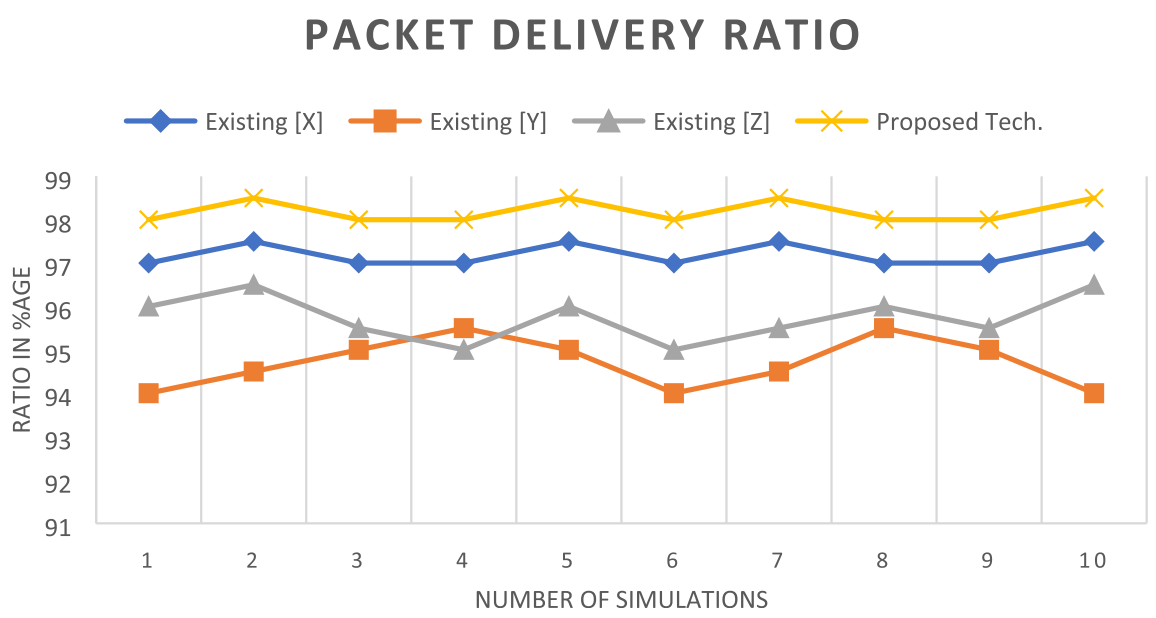
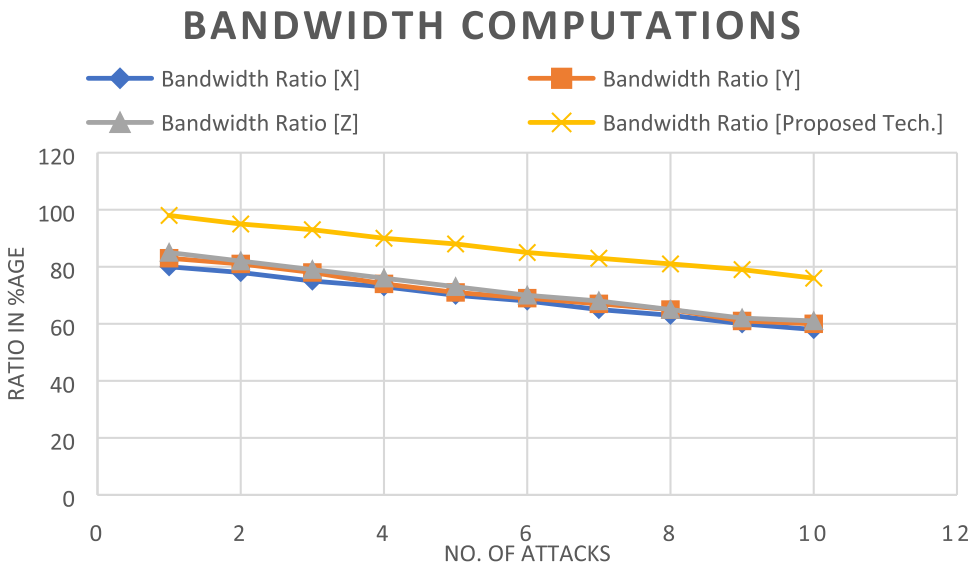


Fig. 14 Packet success delivery rate

Fig. 15 Network bandwidth representation



To calculate the throughput for the Proposed techniques (During the attack) is determined using.

TCP window size: 65,535 bytes or 524,280 bits ∴ defined in IETF RFC 1323.

Round Trip Time: 0.070 s calculate an average of ten packets from host CLI.

Throughput = $\frac{524280bits}{0.070sec}$

= $\frac{7489714x10^6}{10^6}$

= 7.638 Mbps

5.8.3 Throughput after the attack

For the existing techniques (After the attack):-

TCP window size: 65,535 bytes or 524280 bits.

∴ defined in IETF RFC 1323.

Round Trip Time: .080 seconds calculate an average of ten packets from host CLI.

Throughput = $\frac{524280bits}{0.080sec}$

= $\frac{6553500x10^6}{10^6}$

= 6.5 Mbps ∴ throughput decrease

6 Conclusion

With the advancements in network technologies, several types of new and old security measures are needed to be installed on the network. Among other attacks, the ARP spoofing attack is considered the most critical one as an opening attack for other network attacks like DoS, DDoS, Man-in-Middle Attack, etc. This attack is launched in the SDN-based IoT system by spoofing a legitimate node's IP/MAC address. This attack also poisons the network topology, which may lead to unauthorized access to the network resources. the proposed server machine acts as a mediator between the SDN controller and connecting nodes to mitigate the attack condition. Address translation related queries are forwarded to this server, where network traffic is analyzed using a custom method to detect the attack condition. In the case of a network under attack, proper measure is taken, and the attacker's position is located in the network. Several experiments are conducted to measure the proposed system's performance and found that the solution performs better in attack detection and mitigation.

References

1. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., Zhao, W.: A survey on internet of things: architecture, enabling technologies, security and privacy, and applications. IEEE Internet Things J. **4**(5), 1125–1142 (2017)
2. Ding, W., Yan, Z., Deng, R.H.: A survey on future internet security architectures. IEEE Access **4**, 4374–4393 (2016)
3. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., Rajarajan, M.: A survey of intrusion detection techniques in cloud. J. Netw. Comput. Appl. **36**, 42–57 (2013)
4. Jain, J., Pal, P.R.: A recent study over cyber security and its elements. Int. J. **8**, 791–793 (2017)
5. Yang, Y., et al.: A survey on security and privacy issues in internet-of-things. IEEE Internet of Things J. **4**(5), 1250–1258 (2017)
6. Amin, R., Reisslein, M., Shah, N.: Hybrid SDN networks: a survey of existing approaches. IEEE Commun.Surv. Tutor. **20**(4), 3259–3306 (2018)
7. Amin, R., Shah, N., Mehmood, W.: Enforcing optimal ACL policies using K-partite graph in hybrid SDN. Electronics **8**(6), 604 (2019). https://doi.org/10.3390/electronics8060604
8. Amin, R., Shah, N., Shah, B., Alfandi, O.: Auto-configuration of ACL policy in case of topology change in hybrid SDN. IEEE Access **4**, 9437–9450 (2016). https://doi.org/10.1109/ACCESS.2016.2641482
9. Bera, S., Misra, S., Vasilakos, A.V.: Software-defined networking for internet of things: a survey. IEEE Internet Things J. **4**(6), 1994–2008 (2017)
10. S. Shin and G. Gu, "Attacking software-defined networks: A first feasibility study," In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, 2013, pp. 165–166.
11. Shafi, Q. and Basit, A., 2019, January. DDoS Botnet Prevention using Blockchain in Software Defined Internet of Things. In 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST) (pp. 624–628). IEEE.
12. Cox, H., Clark, R. J., Owen, H. L. 2016. Leveraging sdn for arp security. In SoutheastCon, 2016, pp. 1-8.
13. Shimanaka, Toru, S., Ryusuke Masuoka, R., Hay, B. 2019. Cyber deception architecture: covert attack reconnaissance using a safe SDN approach. In Proceedings of the 52nd Hawaii International Conference on System Sciences.
14. Riahi, A., Natalizio, E., Challal, Y., Mitton, N., Iera, A. 2014. A systemic and cognitive approach for IoT security. In 2014 International Conference on Computing, Networking and Communications (ICNC), pp. 183-188
15. Minoli, D., Sohraby, K., Kouns, J. 2017. IoT security (IoTSec) considerations, requirements, and architectures. In 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), pp. 1006–1007
16. Mukherjee, B., Neupane, R. L., Calyam, P. 2017. End-to-End IoT security middleware for cloud-fog communication. In 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), pp. 151–156
17. Gaona-García, P., Montenegro-Marin, C., Prieto, J.D., Nieto, Y.V.: Analysis of security mechanisms based on clusters IoT environments. Int. J. Interact. Multimed. Artif. Intell. **4**, 55 (2017)
18. Hu, K., Houbing, Lu., Wang, Xu., Li, F., Wang, X., Geng, T., Yang, H., Liu, S., Han, L., Jin, Ge.: A front-end electronics prototype based on gigabit ethernet for the ATLAS small-strip thin gap chamber. IEEE Trans. Nucl. Sci. **64**(6), 1232–1237 (2017)
19. Nguyen, T.G., Phan, T.V., Nguyen, B.T., So-In, C., Baig, Z.A., Sanguanpong, S.: Search: a collaborative and intelligent nids architecture for sdn-based cloud IoT networks. IEEE Access **7**, 107678–107694 (2019)
20. Mininet. Retrieved from, http://mininet.org/
21. Noxrepo. Retrieved from, https://github.com/noxrepo/pox
22. Latah, M., Toker, L.: Minimizing false positive rate for DoS attack detection: a hybrid SDN-based approach. ICT Exp. **6**(2), 125–127 (2020)
23. Sood, K., Yu, S., Xiang, Y.: Software-defined wireless networking opportunities and challenges for internet-of-things: a review. IEEE Internet Things J. **3**, 453–463 (2016)
24. Salman, O., Elhajj, I., Chehab, A., Kayssi, A.: IoT survey: an SDN and fog computing perspective. Comput. Netw. **143**, 221–246 (2018)
25. Xie, J., et al.: A survey of machine learning techniques applied to software defined networking (SDN): research issues and challenges. IEEE Commun. Surv. Tutor. **21**(1), 393–430 (2019)
26. Xu, Y., et al.: Efficient DDoS detection based on K-FKNN in software defined networks. IEEE Access **7**, 160536–160545 (2019)
27. Ubaid, F., Amin, R., Ubaid, F.B., Iqbal, M.M.: Mitigating address spoofing attacks in hybrid Sdn. Int. J. Adv. Comput. Sci. Appl. **8**, 562–570 (2017)
28. L Wang, Q Li, Y Jiang, J Wu. 2016. Towards mitigating link flooding attack via incremental SDN deployment. In 2016 IEEE Symposium on Computers and Communication (ISCC), pp. 397–402
29. Baig, Z.A., Sanguanpong, S., Firdous, S.N., Nguyen, T.G., So-In, C.: Averaged dependence estimators for DoS attack detection in IoT networks. Future Gener. Comput. Syst. **102**, 198–209 (2020)
30. Ilyas, Q., Khondoker, R.: Security analysis of floodlight, zeroSDN, beacon and POX SDN controllers. In: Khondoker, R. (ed.) SDN and NFV Security, pp. 85–98. Springer, Cham (2018)

Hamza Aldabbas received his PhD Degree in Computer Science and Software Engineering, De Montfort University, Leicester-United Kingdom (2009–2012). Previously M.Sc, Computer Science (2009) and B.Sc Computer Information Systems (2006) from Al-Balqa Applied University, Al-Salt, Hashemite Kingdom of Jordan. He is currently an Associate Professor at Al-Balqa Applied University /Prince Abdullah bin Ghazi Faculty of Information and Communication Technology-Jordan 2013 until now). Previously a lecturer at De Montfort University/United Kingdom with responsibility for teaching and project supervision at B.Sc & M.Sc levels (2010-to 2012). His research interests include security, IoT, ad hoc networks, machine learning and natural language processing.

Rashid Amin is working as Lecturer at the Department of Computer Science, University of Engineering and Technology, Taxila, Pakistan since august,2014. Before this, he worked as Lecturer at the University of Wah, Wah Cantt, Pakistan for 4 years. He received MS Computer Science and Master of Computer Science (MCS) from International Islamic University, Islamabad. His MS thesis was on Peer -to - Peer Overlay Network over Mobile Ad hoc Network. He is a Ph.D. student at Comsats Institute of Information Technology, Wah Cantt. He has completed his thesis that is under evaluation. His area of research is Hybrid Software Defined Networking. His current research interests include SDN, HSDN, Distributed Systems, P2P and Network Security. He has published several research papers on the topics of hybrid SDN, SDN in well reputed venues (like IEEE Communication Surveys & Tutorial, IEEE Access, Electronics MDPI, IJACSA, etc.). He has been serving as reviewer for international Journals (e.g., NetSoft, LCN, GlobeCom, Fit, IEEE Wireless Communication, IEE IoT, IEEE J—SAC, IEEE Access, IEEE System Journal,, Pervasive and Mobile Computing (PMC), Journal of Network and Computer Applications (JNCA), Peer -to Peer Networking and Applications (PPNA), and the Frontiers of Computer Science (FCS), International Journal of Communication System, etc.