Naif Arab University for Security Sciences
Journal of Information Security and Cybercrimes Research
مجلة بحوث أمن المعلومات والجرائم السيبرانية
https://journals.nauss.edu.sa/index.php/JISCR

JISCR

Spoofing Attack Mitigation in Address Resolution Protocol (ARP) and DDoS in Software-Defined Networking

Harun Jamil[1], Abid Ali[2], and Faisal Jamil[3]*

[1] Department of Electronics Engineering, Jeju National University, Jejusi 63243, Korea.
[2] Department of Computer Science, University of Engineering and Technology Taxila, Pakistan.
[3] Department of ICT and Natural Sciences, Faculty of Information Technology and Electrical Engineering, Norwegian University of Science and Technology (NTNU), Larsgårdsvegen 2, Å lesund, 6009, Norway.

Received 01 Apr. 2022; Accepted 06 Jun. 2022; Available Online 16 June. 2022

Abstract

Software Defined Networking (SDN) shows network operations to be performed for efficient network operations. Due to the increase in network devices, the percentage of attacks is also increased, and it is challenging to provide defense against such attacks. In SDN, the control plan is separated from the data plane. The control plan is implemented using some central devices called SDN controllers. In SDN Address Resolution Protocol (ARP), spoofing and Distributed Denial of Services (DDoS) attacks are carried out on an enormous scale. These are commonly launched attacks in SDN. Due to these attacks, the network performance is down, and network services are dead. This paper proposed a new auto detection methodology to detect ARP and DDoS attacks and mitigate SDN networks from these attacks.

Additionally, we implemented two algorithms: one for flow rules and the second for attack detection. An individual server was installed to check the malicious traffic installation. We present the new forward flooding rules to detect and mitigate attacks. The experiments are performed using LINUX-based network implementation. Our proposal successfully improves network security and enhances network efficiency.

I. Introduction

In the present era, incredible innovations and development have been made in the field of telecommunications. Computer networking has a key value in the huge and widely spread telecommunications network. A computer or data network is a telecommunications network that allows several nodes to share resources [1]. Whenever we talk about resource sharing, the risk of attacks always exists, leading to misuse and spoiling the network resources or even service denial. In traditional computer networks, we are familiar with attacks like DDoS, packet spoofing, and link flooding attacks, etc. ARP broadcast storm is one of the major causes of traffic congestion in the traditional network [2].

Software-Defined Networking (SDN) is a new network paradigm that brings innovations to the world's networks. SDN separates the control plane

Keywords: Cybersecurity, Software Defined Networks, ARP Inspection, Address Resolutions, Intrusion Prevention System Design.

Production and hosting by NAUSS

* Corresponding Author: Faisal Jamil
Email: faisal@jejunu.ac.kr
doi: 10.26735/VBVS3993

1658-7782© 2022. JISCR. This is an open access article, distributed under the terms of the Creative Commons, Attribution-NonCommercial License.

from the data plane. SDN is a programming-based network handling policy. SDN resolves the different aims of traditional networks to separate the control and data planes. ARP spoofing broadcast storm is also managed through SDN's data plane [3]. In SDN, a centralized unit is a control plane built using the controller. Load balancing, routing, network intelligence, and network management policies are implemented using an SDN switch [4]. The traditional network framework uses simple network protocols to implement very limited security and cannot handle the ARP spoofing and DDoS network attacks. The traditional network consists of routers, switches, modems, hubs, and other network adapters connected, whether wireless or wired [5]. The communication protocol is implemented to interact among these network connections. The network administrator is bound to implement these limited protocols and implementations. Protocols are implemented to communicate these devices in the network. In a traditional network, the network admin controls the network operations by allowing configuration information through the vendor provided. In the network, from device to device communication, network operations are implemented through the division of network responsibilities on network switches, routers, and controllers [6].

Attacks on SDN are extensive in nature. ARP was introduced to shift the IP addresses to local network addressesin order to to maintain the ARP of the host device. ARP lacks for the authentication that which system sends the packet which leads towards the ARP packets security issues in SDN. ARP spoofing poisons the ARP maintained cache by sending wrong ARP packets. In the ARP, the host maintains its cache by updating the incoming and outgoing ARP packets. ARP spoofing is of two types: one is ARP request, and the second is ARP reply attacks [7].

SDN is a new paradigm in network concepts with the ability to overcome the limitations of traditional networks. SDN also overcomes the security issues a traditional network faces to implement the corresponding policies. SDN decouples the traditional network and plans information to more intelligent-based network operations. There are two main planes: the control plane and the data plane. In SDN, the decoupling is possible by separating the hardware from the software layer. Fig. 1 shows the decoupling of the SDN network into two sub-layers.

The control plan plane is transferred to the software layer, and the data plane is shifted into the hardware layer. A centralized application is developed to implement and control the networking policies in the network. SDN also encourages opening the ways for network virtualization. Data flow optimization, accuracy, flexibility, and consistency in the configuration are very efficient in SDN [5].

Although SDN is the security adopted in network operations, it protects the user from network attacks. In SDN, attacks like DDoS and Link Flooding Attack (LFA) are launched through Address Resolution Protocol (ARP) or IP. DDoS and LFA are the most deployed network attacks in SDN. IP or ARP packets are designed to know about ARP spoofing attacks in SDN using MAC address [7] [8] [9]. The IP address of each system is designed to participate effectively in the network performance. A third-party can alter these packets easily. The MAC and IP addresses can be changed into particular host networks. The situation develops new challenges for SDN, and we aim to create a new SDN policy to mitigate ARP spoofing attacks. ARP or IP packets are usually used to know the MAC address or the system's IP address in the network.

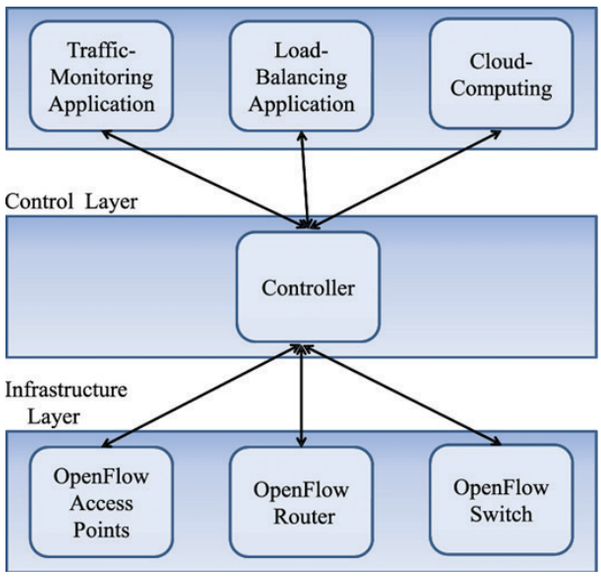

Fig. 1 Basic Architecture of SDN Network [6].

These packets can be modified easily by an adversary party, and the MAC address or the IP address can be changed to a particular host from the adversary party. This creates new challenges in SDN, and this research was undertaken to resolve those issues.

The controller is a key component of SDN. The attacks want to hold the controller due to its high involvement in SDN traffic flow and control. The controller is effective by multiple attacks, and among these attacks, the ARP spoofing is most commonly used by the intruders to attack ARP. Malicious nodes select the ARP and poison the network, resulting in failure of network operations, topology failure, and hijacking of the network operations. ARP spoofing and DDoS attacks are considered in this research paper. This paper implements the solution for ARP packets that are considered secure through the server. The controller directly deals with packets from switches. The proposed algorithm is implemented on the proposed server in the network. Packets from every host are analyzed for possible ARP spoofing attacks. The packets are dropped with reasonable host nodes to get actual programming exercises. The malicious traffic blocking algorithm will block the specific port after detecting threats. In this way, SDN controller will be saved from unnecessary processing, resulting in efficient and secure usage of the network resource.

A. Research Contributions

In this research, we address ARP spoofing and DDoS attacks that are mitigated through the proposed research in an SDN environment. The main research contributions are mentioned in the below points.

- Proposed ARP spoofing module in SDN. It prevents ARP spoofing attacks and provides efficient DDoS attack detection and mitigation.
- Installation of flow rules on every switch makes the network monitor the traffic and entries on the controller to enhance the packet forwarding mechanism.
- Proposed ARP attack detection module installs ARP protection flow rules and prevention from ARP spoofing attacks.
- The proposed algorithm prevents ARP spoofing DDoS attacks and controls cache security through the proposed architecture.
- The proposed system performance is compared with recent ARP spoofing attacks over parameters like connection development, packets flow, overhead time, CPU utilization before, under, and after attacks, and flow rules installation using a controller.

In section II of the paper, we present the relevant up-to-date literature review, which enlightened the contributions of the proposed work. Section III explains a proposed solution using two main algorithms and proposed methodology discussions. Results and discussions are defined in section IV of the proposed work, in which we compare the results with existing approaches. Finally, Section V presents the conclusion and future directions related to the proposed work.

II. Literature Review

In Software Defined Network, data and control planes are separated, which helps in faster configuration and provisioning of network connections. Using the SDN approach, both the network and behavior of traffic can be administrated in a centralized model. Consequently, no independent access and configuration for each network device are required. The decision-making system which decides the destination for traffic (control plane) and the underlying systems that forward the traffic towards the destination (data plane) has been decoupled in this networking paradigm [10]. In addition to these advantages, unlike a traditional network, the SDN network can be made more efficient and flexible once programming controls are enabled over the traffic and devices [11]. The logical view of the commonly accepted Software Defined Network reference architecture can be viewed in Fig. 2. This architecture has a centralized intelligence in software-based controllers. These controllers can control the network devices without any vendor interference and have a global view of the network. Consequently, there is no need to understand and implement several protocols on these network devices; instead, SDN controllers can provide enough instructions to these devices

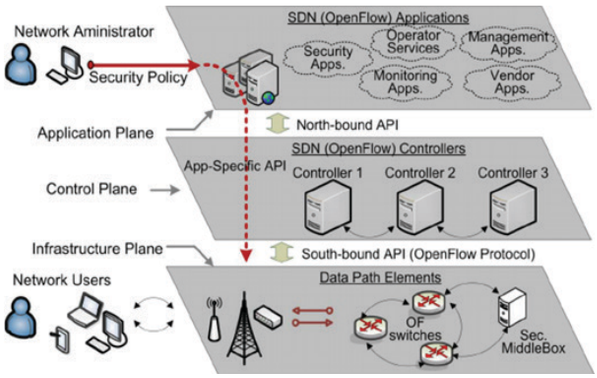

Fig. 2 Software Defined Network Reference Architecture.

to provide such functionalities. All these aspects save resources and time because they make the network easily controlled or programmed by centralized controllers rather than configuring several devices manually [12].

In [9], Rangisetti et al. proposed the SDN features using ARP packets. The authors handled switches using the controller in multiple subnets. The edge system is implemented with subnets. The edge network controls all the ARP requests. The controller controls the network operations and handles required flow rules on each switch separately to access these situations. Flow rules are installed and handled using flow rule implementations through the controller flow table. In this approach, the controller handles all network operations and keeps track of all SDN devices. The Controller ARP Table (CAT) uses network devices to track the packet's destination. The table is updated from time to time for new packets to arrive and be released from devices.

In [13] Fabian Schneider et. al. explored the ARP traffic in SDN. Currently, ARP traffic handling is one of the main challenges in SDN to control and manage. The control is generated if network devices are not configured properly. The bulk in the network traffic creates unnecessary traffic generation and overhead in packet delivery in the network. The Controller ARP Table creates and installs the flow entries on the SDN switch after installing configuration information. The authors explain that ARP spoofing attacks mostly affect the controller. If the main component of the SDN is infected, then the hacker infects the whole network.

The authors in [12] proposed a new technique for scalability, performance, interoperability, and security policies in SDN. The authors propose the performance and programmability tradeoff for the development of the SDN network [14]. The authors conclude that SDN is suitable for traditional network operations. The communication is overhead between controller and switches for the network operations. The multi-controller environment is established to maintain the data at the network's backend. The hybrid approach is achieved through the proposed scenario. The overhead time is reduced for communication through the controller [3].

This research implements the security issues using a centralized SDN controller. DDoS attacks are implemented using security models defined in the network operations. The security is implemented using switches and controllers to implement the available security [15] in SDN and the interoperability in the network to study the main features. The traditional SDN controller is not effective through the provided network. The controller manages the protocols and standards implemented in SDN. The internal network devices are configured to have an effective link between legacy network devices and SDN. The DDoS attack can degrade the network performance and become a state of hijacking. SDN centralized controller is a single point of failure where the attacker can easily hack the network, if they access the controller by attacking the controller [16]. Fig. 3 shows the impact of the DDoS attack on SDN.

We reviewed the DDoS and ARP spoofing attack using a literature review, which is not implemented

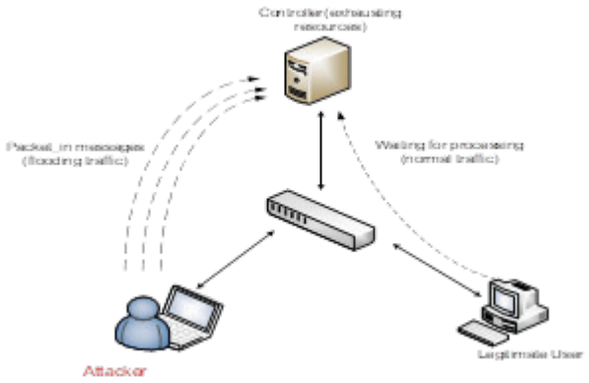

Fig. 3 Impact of DDoS attack on SDN [8].

in the current techniques. It is an issue implemented in the network operations. The problem is critical to managing ARP protocol in SDN to get MAC or IP information. ARP spoofing attacks enable the packets to travel and reach an unauthorized malicious node [17]. From this attack, CAT (Controller ARP Table) can be poisoned. Due to this, the wrong entry in the CAT table network can be hijacked, or network performance can be degraded. This attack can lead to further attacks like a man-in-the-middle attack (MITM). The network security of the proposed technique is at risk. The examples are shown through literature and the existing state-of-the-art studies to review the problem statement. The following challenges are considered in this research for mitigation in the SDN environment:

- We implemented customized information retrieval through the network topology.
- ARP packets are difficult to manage network topology, and the proposed server is customized through proposed algorithms.
- The proposed server implements the possible ARP protection schemes to implement the server-based topology technique.
- Device identification and malicious traffic generation technique.
- Block the malicious device by implementing further processing in controller architecture.

III. Proposed Solution

This paper proposed a Network Device Identification procedure to address the proposed server. The proposed server mitigates these attacks due to the implemented algorithm. For this purpose, the controller should have all the network device information, and this record is also maintained by the CAT (Controller ARP Table). Our proposed server takes the data from the controller and the DHCP server to perform actions to mitigate malicious traffic from the network.

A. Flow Rules Installation

Initially, we get topology information from network devices. After that, the flow rules are installed through Algorithm 1 on all switches to direct the ARP traffic to the proposed ARP handling server.

Algorithm 1 Flow Rules Installation

Initialization

Input: Number of Packets

Output: Forward Packet Route

Procedure:

1. Start
2. $info_{controller}^{(New)}$ → Switches
   ∴ Controller fetch switches Information
3. $install_{flow-rules}^{(packets)}$ → Switches
   ∴Flow Rules Installation
4. if($PKT_{current}$ → APR) II ($PKT_{destination}$ == FF:FF:FF:FF:FF:FF)
   PKT_{sent}→ $Port_{verification}$
5. else (PKT_{sent} → $Controller_{packet}^{(All)}$ && Forward → $Flow_{packet}^{(Selected)}$)
6. end if
7. end

The controller is introduced to install all the flow rules on every switch or network server. The controller is responsible for handling the flow rules and maintaining the paths. After installing flow rules, the switches are programmed to forward the ARP packets/traffic to the SDN switch. After receiving network traffic distribution on the proposed server, complete system analysis is performed. Algorithm 1 explains the flow rules installation and ARP packet determination.

B. Detect ARP Attack in Network

Algorithm 1 implements the topology information and installs switch flow rules. The outcome of Algorithm 1 effectively shows the traffic flow rules and provides effective control parameters to determine the flow information. Graphical information is presented to store network information. The flow information is used to verify the ARP requests. The proposed server checks and detects packets from every host in the network. The initial step is to check the validity of packets that either belong to the proposed network or not. After this verification, the ARP request is implemented to check whether it belongs to the proposed network or not. If the request belongs to our network, the actions are taken to implement the network traffic handling and ARP attacks. Fig. 4 explains the proposed scenario.

Initially, the users generate ARP traffic to forward the network traffic using ARP. If the user is connected with a high level of legacy switch, the packets are forwarded to network switches. On the arrival of packets, the flow entries in the flow table are checked against each packet. After checking, if some or all of the flow entries are not matched, the packet is forwarded to the proposed server for ARP check. The scenario is shown in Fig. 3. The proposed server is connected to every switch in the network. The proposed server uses Algorithm 2 implementation to check the possible ARP attack scenario. The packets are checked based on network topology. If all packets belong to our network, they should be forwarded with the response towards the server implementation. If they do not belong as checked by the proposed server, packets are dropped from the network.

We consider the case study of PCs which generated ARP packets with some IP addresses. Let us consider a PC-1 with the address of 10.0.0.1 to generate ARP packets for IP addresses. The network operations are shown in Fig. 3. The switches receive the packet and are forwarded to the SDN switch for possible implementation. SDN switch checks the packet type, data, and IP address and forwards to the next switch, if it is an ARP data request, to the proposed server in the network. Upon receiving the server, it decides whether the packet belongs to the proposed network or some other unknown network. After examination, the proposed server decides successfully that the packet belongs to the proposed network, then an ARP reply is generated to PC-1 through the SDN switch. The possible communication starts, and network performance is communicated. Additionally, if the packet belongs to the same network, it drops after verification of MAC, if MAC does not exist in flow table rules. Based on our solution, if the packet is not dropped, the attacker can launch an attack on the network.

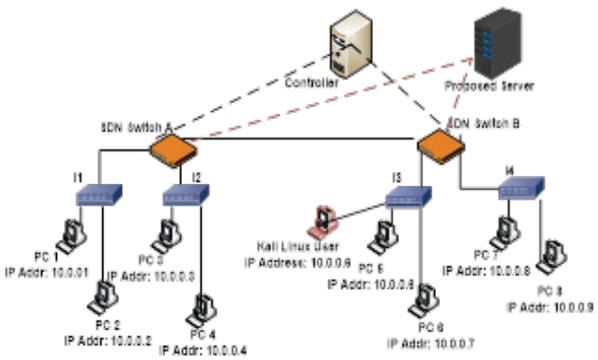

Fig. 4 The architecture of the Proposed Solution.

Suppose a rival user forwards the packet to the proposed network in the second scenario. Based on information in the packet, the user pretends to be the honest user of the same network, but the IP address belongs to another user from the same network. Based on the packet's initial information, the user looks like a genuine user from the network. In this case, the proposed server checks MAC to IP mapping to check the originality of the pretended user from the IP mapped table implemented in the proposed server. If the entry matches the MAC address, the server responds through the proposed MAC address that the packet comes from the original user. Otherwise, the proposed server drops the packet and saves the entry into the flow table. Another check is implemented in proposed Algorithm 2 that if multiple requests are generated from the source device, then the corresponding port on the network is blocked.

Algorithm 2 ARP Attack Detection

Initialization
Input: Broadcast Addresses, N-Nodes, ARP Packets
Output: Flow Rules
Procedure:

1. $Start$
2. $Initialize \rightarrow CAT[\,]Table$
3. $for(all\ nodes)\ do$
4. $\quad Add\ (IP, MAC) \rightarrow CAT[\,]\ Table$
5. $End\ for$
6. $if\ (PKT_{src}\ !=\ CAT[\,]\ \&\&\ PKT_{dest}\ !=\ CAT[\,])$
7. $\quad drop_{PKT}()$
8. $else$
9. $\quad if(PKT \rightarrow ARP)$
10. $\quad\quad check(IP\ \&\ MAC) \rightarrow CAT[\,]$
11. $\quad\quad send(IP/MAC)$
12. $\quad End\ if$
13. $\quad if(PKT_{dest} ==$
    $FF{:}FF{:}FF{:}FF{:}FF{:}FF\ \&\&\ PKT_{src} \rightarrow CAT[\,])$
14. $\quad\quad install_{rules}(broadcast)$
15. $\quad End\ if$
16. $End\ if$

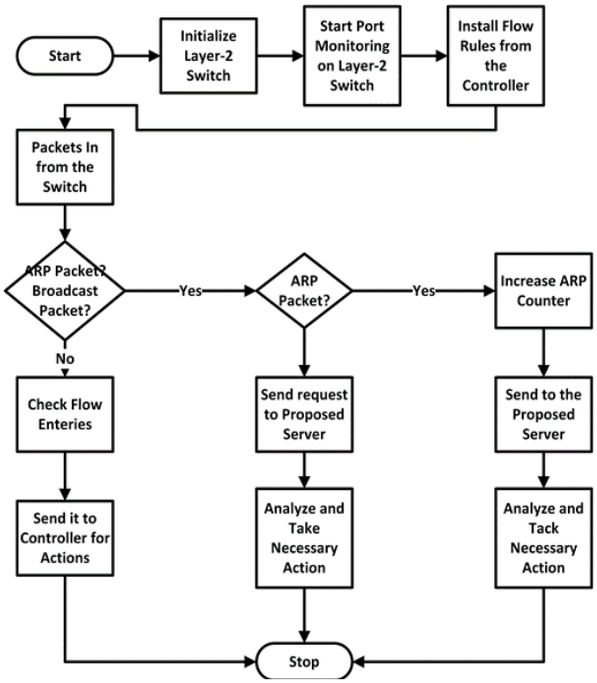

Fig. 5 Process Flow Diagram of Proposed System.

C. System Flow Data Diagram

Fig. 5 shows the process flow diagram of the proposed system. All the procedures are implemented in the flow diagram from start to end. The flow diagram clearly shows that the open flow switch sends this traffic to the proposed server whenever ARP or malicious traffic is generated. Then, our server analyzes the traffic and takes appropriate action. If traffic was found malicious, our server drops the packets of this traffic and blocks the device from generating malicious traffic.

VI. Results and Discussions

We selected Intel core i7 6th generation system with 8 GB of Ram and Ubuntu Linux Operating System. The system contains a hypervisor server with 16GB of RAM and 32 cores for development. We use a Mininet simulation environment to test the experiments over virtual machines in this setup. We define multiple SDN switches, hosts, legacy switches, and controllers according to the networking environment. The proposed problem statement and the proposed solution are implemented to extend the provided link between these devices. Virtual links are created between the hosts, networks, and other defined parameters.

Fig. 6 shows the proposed system topology. We designed three legacy switches, SDN switches, and nine host machines in the experiment. To attack the provided scenario, we use one machine for scenario purposes with 10.0.0.11 as an IP address. The attack scenario is implemented using Kali Linux implementation. The Kali Linux-based system implements the attack scenario on the network with the implementation of the system. The machines created the ARP and other spoofing attacks to implement in the proposed scenario to poison the network topology. The proposed system is implemented with the controller to exchange the attacked data. The rules are implemented on SDN switches for efficient network traffic flow information. Attack detection, load on CPU, throughput, attack mitigation, through proposed Algorithm 2. The algorithm implements the different attack scenarios to implement the proposed methodology. The ARP request and reply attacks are initiated to test the system's security. In this research, we considered ARP spoofed requests, attacks, reply to attacks, and DDoS attacks for consideration to protect the network.

A. Spoofed ERP Attacks

In the Spoofed ERP attack, the victim's computer's cache table is filled with fake entries from the hosts. The attack is initiated to intercept the victim's network traffic. The attack injects thousands of ERP requests into the network or on the victim's PC. The cache is updated through incorrect entries inside the network. In our proposed solution, we mitigate this type of network attack. Spoofed ERP requests are of two types; one is that an ERP request attack looks similar to an ARP request attack through IP addresses. The network is considered effective through legitimate users and provides efficient cache information. The information is maintained through network policies.

When the network is in attack mode, communication is literally not possible, or only very little communication is possible, due to overhead in the network. The attacker monitors the user's traffic and packets/traffic flow. The SDN controller uses the ARP requests with packet information to control the flow entries on switches using the proposed server.

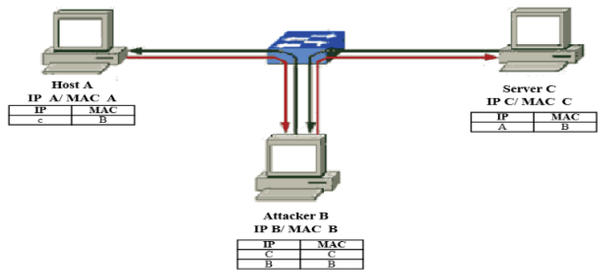

Fig. 6 Demonstration of ARP Spoof Attack.

Other users update the cache information through wrong entries in the table flow. The communication is halted and provides very messy information. The traffic control and network latency information is initiated through the provided network traffic handling parameters. The hosts are responsible for communication among parties. Fig. 6 shows the example of a Spoof ARP attack.

B. Distributed Denial of Service (DDoS) Attack

DDoS is one of the most dangerous network attacks. It uses the ARP spoofing method to attack the network. DDoS eventually degrades the network performance and cuts down the network operations. Due to DDoS, one particular device in the network is targeted and fails for whole network operations. It degrades the network performance and provides the network delay in the network operations. The DDoS attacks are launched through multiple computers on one single device or computer. In response to these, the computer fails to access the network operations. The single targeted computer is spoofed through attack scenarios.

To address these problems, we evaluate our proposed system over network parameters like attack detection time, CPU load, throughput, and attack mitigation time. Attack detection time is the sum of time from the attack launched to the detection of the attack by the controller on the network. Mitigation time is the total time to mitigate the attack, after attack detection in the network.

C. Attack Detection for the Proposed Methodology and Mitigation Time

Attack Detection time is when the possible ARP spoofing attacks are detected in the network, and Mitigation time is the total time to mitigate against these attacks. Algorithm 2 implements the procedures for attack detection and mitigations in Fig. 6. The proposed technique is compared with the existing technologies. In Fig. 4, we evaluate that our proposed system performs better than previous techniques to detect and mitigate malicious attacks. The traditional network is secured using SDN technology. The investments in the SDN switches are limited and provide limited abilities to communicate and provide effective management.

In this experiment, we adopted twenty computers through a simulation environment. The real-time environment is designed to check the proposed simulation environment. The existing solution is implemented through performance parameters. The performance is compared with existing solutions. From Fig. 6, it is clear that the proposed system performs better in response, detection, and mitigation times in the proposed technique compared to existing approaches. According to the policy of port blocking, the fake user is detected with intruder design. A policy of port blocking while detecting any intruder or fake user using ARP spoofing techniques minimizes the network's traffic load. This led to an increase in 100% bandwidth of the links between the hosts. From the equation:

$$\text{Bandwidth per user} = \frac{\text{Total bandwidth}}{\textit{Number of users}} \quad (1)$$

In the attack scenario, which is described in Fig 3.2 Blocking the intruder or hacker in our proposed algorithm policy, then bandwidth is equal to

$$\text{Bandwidth per user} = \frac{\text{Total bandwidth}}{\textit{Number of users -Intruder or hacker}} \quad (2)$$

From the above equation, it is clear that bandwidth per user is increased due to a small value or less value in the denominator.

D. CPU Utilization

In our methodology, CPU use is very important to test the performance of the proposed algorithm. The CPU performance is checked before, during, and after the attack. The performance is also checked during the mitigation process. From Fig. 8, it is clear that in our proposed methodology, the CPU utilization is higher than the existing

approaches. The normal use of CPU operations is provided to enhance the working performance. The proposed system's CPU utilization does not affect the system performance and enhances the networking solutions. The utilization of the proposed server is not effectively managed and enhances the system performance ratio. Through the CPU utilization, the server performance is also higher.

The CPU utilization graph in Fig. 7 of our proposed methodology shows better performance than existing techniques. The CPU utilization of the proposed system is more efficient than the traditional controller in the system. The controller in the most recent literature is compared with the existing system implementation. Our proposed system controller information is lower than the existing system with system implementation details. The controller forwards the malicious traffic to the proposed server.

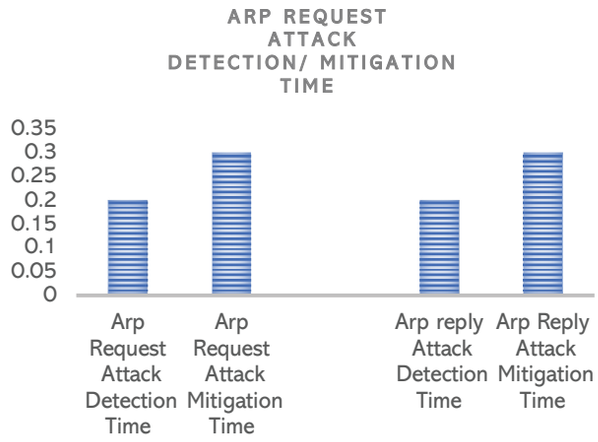

Fig. 7 Time Evaluation of ARP attacks with Proposed & Existing Sols.

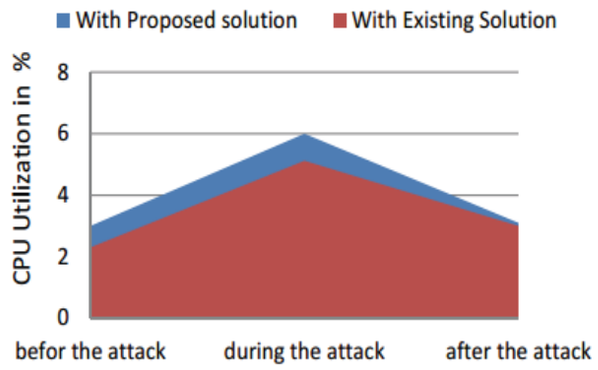

Fig. 8 Comparisons of CPU Utilizations with Proposed and Existing Sols.

E. Packet Delivery Ratio

The packet delivery ratio is another parameter considered to measure the performance of the proposed system. With time intervals, the packet delivery ratio is measured. The successful evaluation of the proposed technique evaluated that the proposed approach effectively evaluated the packet delivery ratio. Fig. 9 shows the packet delivery ratio. The proposed solution effectively delivers more packets than the existing solution. The proposed server detects the attack and minimizes its effects when a new attack is launched.

In the proposed system, the performance is better in terms of the packet delivery ratio. When the attack is launched in the proposed system, the system detects the attack and locates the targeted host. Based on the host location, the proposed system successfully blocks the port. Due to the proposed architecture, the traffic is minimized for attacker packets; as a result, the packets delivery ratio will be very high.

F. Network Throughput

Throughput is another factor used to measure the network performance. In the proposed methodology, the throughput is the minimum utilization of system resources. In the proposed approach, we link the host and controller before and after the attack. The throughput is compared with the existing system implementation. Fig. 10 effectively shows the throughput of our proposed system with the existing state-of-the-art techniques. The results show that the proposed system effectively enhances the working of network performance.

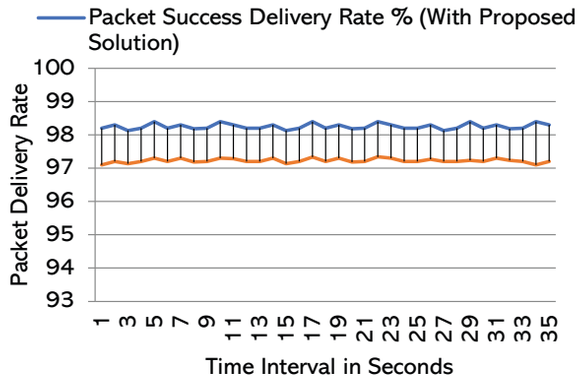

Fig. 9 Packet Success Delivery Rate.

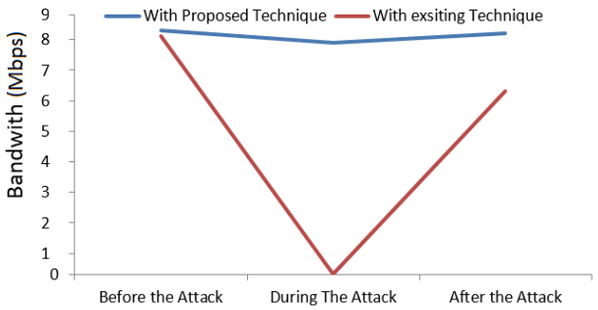

Fig. 10 Throughput between the Links.

The throughput in the proposed network is measured through equation 3. The throughput is measured by data packets like TCP windows size to Round Trip Time (RTTT).

$$\textbf{Throughput} = \frac{\text{TCP window size (Data Packets)}}{\text{(Round Trip Time (RTT)}} \quad (3)$$

According to the proposed methodology from Fig. 3, the throughput before attack in the existing technique is computed as in the below section.

Throughput from the literature before the attack

Default TCP windows size = 65,535 bytes / 524280 bits Round Trip Time = .060 seconds

To calculate the average throughput, equation 4 is.

$$\textbf{Throughput} = \frac{524280}{.060} = 8738000 \times \frac{10^6}{10^6} = 8.738 \text{ Mbp} \quad (4)$$

Throughput from proposed technique before the attack

Default TCP window size = 65,535 bytes / 524280 bits Round Trip Time = .060 seconds.

The average packet delivery is computed using equation 5.

$$\textbf{Throughput} = \frac{524280}{.060} = 8738000 \times \frac{10^6}{10^6} = 8.738 \text{ Mbp} \quad (5)$$

Throughput during the attack

Existing State-of-the-Art Techniques

Window Size (TCP) = 0 bits (6)

It cannot transform any data due to DDoS attack.

Round Trip Time (RTT): Total time from submission to completion. The answer is in real number.

$$\textbf{Throughput} = \frac{0 \text{ bits}}{\text{a large number (seconds)}} \quad (7)$$

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

$$= 0 \text{ Mbps}$$

During attack throughput for proposed technique Consider

Window Size (TCP) = 65,535 bytes/524280 bits (8)

Round Trip Time (RTT) = 0.070 seconds. (9)

So

$$\textbf{Throughput} = \frac{524280 \text{ bits}}{.070 \text{ seconds}} \quad (10)$$

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

$$= 7.638 \ (7)\ 7.638 \ \text{Mbps}$$

Throughput after the attack

For the existing techniques (After the attack)

Window Size = 65,535bytes/ 524280 bits (11)

Round Trip Time (RTT) = 0.080 seconds.

$$\textbf{Throughput} = \frac{524280 \text{ bits}}{.080 \text{ (seconds)}} \quad (12)$$

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

$$= 6.5 \text{ Mbps}$$

For the proposed techniques (after the attack)

Default Window Size (TCP) = 65,535 bytes/524280 bits (13)

Round Trip Time (RTT) = 0.060 seconds. (14)

$$\textbf{Throughput} = \frac{524280 \text{ bits}}{.060 \text{ seconds}} \quad (15)$$

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

$$= 8.738 \text{ Mbps}$$

From the above-compiled results, we made the graphs shown in Fig. 9. From the graphs, we can be sure that our proposed solution performed better in the worst conditions due to special features of the block, the port, or the intruder.

V. Conclusion and Future Directions

SDN Layer-2 attacks are considered in this research. A novel ARP spoofing and DDoS attack detection and mitigation technique are proposed in this research. DDoS and ARP spoofing are the most vulnerable attacks in SDN that downgrade the network performance. Most of the attacks are

launched in this network through network spoofing and performance measurements. The attacks poisoned the network topology and enhanced network management. The proposed system measures the network topology and provides an individual server-based implementation mechanism. The server-based customized mechanism is communicated through the proposed architecture. The proposed solution consists of a server with a customized network topology connected with the network. Our proposed Algorithm 1 is initiated to install the SDN switches' flow rules to establish the SDN traffic management. The packets are forwarded to the server for the request and response parameters. The attack detection and mitigation algorithm detect ARP requests from sources. The attacker location is also measured to monitor the network performance of the proposed system. We implement the proposed architecture using a simulation-based technique. The results show that the proposed technique performs better in terms of throughput, CPU utilization, network latency, and work in an attack scenario.

In future work, this technique can be used in data centers. In data centers, the majority of the traffic consists of ARP packets to locate a device. Data centers consist of thousands of machines. If one machine moves, then the ARP request packet generates in the network, and every machine that receives this packet and responds pond will be unicast by the intended machine. Large network bandwidth is utilized from this activity, and system processing power is also utilized. To overcome this issue, we can use our proposed techniques. After that, the controller is responsible for that kind of traffic.

Funding

This article did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of Interest

Authors declare that they have no conflict of interest.

References

[1] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing IoT Services Through Software Defined Networking and Edge Computing: A Comprehensive Survey," *IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 1761-1804, thirdquarter 2020, doi: 10.1109/COMST.2020.2997475.

[2] X. Zhang, L. Cao, Z. Meng and X. Yao, "A solution for ARP attacks in software defined network," in *AIIPCC 2021; 2nd Int. Conf. Artificial Intell. Inf. Process. Cloud Comput.*, China, 2021, pp. 1-9.

[3] T. Girdler and V. G. Vassilakis, "Implementing an intrusion detection and prevention system using Software-Defined Networking: Defending against ARP spoofing attacks and Blacklisted MAC Addresses," *Comput. Electr. Eng.*, vol. 90, p. 106990, Mar. 2021, doi: 10.1016/j.compeleceng.2021.106990.

[4] E. Unal, S. Sen-Baidya, and R. Hewett, "Towards Prediction of Security Attacks on Software Defined Networks: A Big Data Analytic Approach," in *2018 IEEE Int. Conf. Big Data (Big Data)*, USA, 2018, pp. 4582-4588, doi: 10.1109/BigData.2018.8622524.

[5] A. F. Ali and W. S. Bhaya, "Software Defined Network (SDN) Security Against Address Resolution Protocol Poisoning Attack," *J. Comput. Theor. Nanosci.*, vol. 16, no. 3, pp. 956-963, 2019, doi: 10.1166/jctn.2019.7982.

[6] H. Y. Khalid, P. M. Ismael, and A. B. Al-khalil, "Efficient Mechanism for Securing Software Defined Network against Arp Spoofing Attack," *J. Duhok Univ.*, vol. 22, no. 1, pp. 124-131, 2019, doi: 10.26682/sjuod.2019.22.1.14.

[7] N. Ravi, S. M. Shalinie, and D. Danyson Jose Theres, "BALANCE: Link Flooding Attack Detection and Mitigation via Hybrid-SDN," *IEEE Trans. Netw. Serv. Manag.*, vol. 17, no. 3, pp. 1715-1729, Sept. 2020, doi: 10.1109/TNSM.2020.2997734.

[8] M. Pal Singh and A. Bhandari, "New-flow based DDoS attacks in SDN: Taxonomy, rationales, and research challenges," *Comput. Commun.*, vol. 154, pp. 509-527, Mar. 15, 2020, doi: 10.1016/j.comcom.2020.02.085.

[9] A. K. Rangisetti, R. Dwivedi, and P. Singh, "Denial of ARP spoofing in SDN and NFV enabled cloud-fog-edge platforms," *Clust. Comput.*, vol. 24, pp. 3147-3172, June 2021, doi: 10.1007/s10586-021-03328-x.

[10] P. -W. Tsai, C. -W. Tsai, C. -W. Hsu, and C. -S. Yang, "Network Monitoring in Software-Defined Networking: A Review," *IEEE Syst. J.*, vol. 12, no. 4, pp. 3958-3969, Dec. 2018, doi: 10.1109/JSYST.2018.2798060.

[11] J. Xie et al., "A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 1, pp. 393-430, Firstquarter 2019, doi: 10.1109/COMST.2018.2866942.

[12] T. -Y. Lin et al., "Mitigating SYN flooding Attack and ARP Spoofing in SDN Data Plane," in *2020 21st Asia-Pac. Netw. Oper. Manag. Symp. (APNOMS)*, Korea, 2020, pp. 114-119, doi: 10.23919/APNOMS50412.2020.9236951.

[13] M. N. Munther, F. Hashim, N. A. Abdul Latiff, K. A. Alezabi, and J. T. Liew, "Scalable and secure SDN based ethernet architecture by suppressing broadcast traffic," *Egypt. Inform. J.*, vol. 23, no. 1, pp. 113-126, Mar. 2022, doi: 10.1016/j.eij.2021.08.001.

[14] J. Xia, Z. Cai, G. Hu, and M. Xu, "An Active Defense Solution for ARP Spoofing in OpenFlow Network," *Chin. J. Electron.*, vol. 28, no. 1, pp. 172-178, Jan. 2019, doi: 10.1049/cje.2017.12.002.

[15] H. Aldabbas and R. Amin, "A novel mechanism to handle address spoofing attacks in SDN based IoT," *Clust. Comput.*, vol. 24, pp. 3011-3026, 2021, doi: 10.1007/s10586-021-03309-0.

[16] V. K. Tchendji, F. Mvah, C. T. Djamegni, and Y. F. Yankam, "E2BaSeP: Efficient Bayes Based Security Protocol Against ARP Spoofing Attacks in SDN Architectures," *J. Hardw. Syst. Secur.*, vol. 5, pp. 58-74, 2021, doi: 10.1007/s41635-020-00105-x.

[17] M. Reazul Haque et al., "Analysis of DDoS Attack-Aware Software-Defined Networking Controller Placement in Malaysia," in Recent Trends in Computer Applications, J. M. Alja'am, A. El Saddik, and A. Sadka, Eds., Cham, Switzerland: Springer, 2018, pp. 175-188.