A New Approach for Detecting and Mitigating Address Resolution Protocol (ARP) Poisoning

Ahmed A.Galal[1]*
Department of Network
Misr University for Science and Technology Giza, Egypt

Atef Z.Ghalwash[2]
Department of Computer Science
Faculty of Computers and Artificial Intelligence, Helwan University
Cairo, Egypt

Mona Nasr[3]
Department of Information Systems
Faculty of Computers and Artificial Intelligence, Helwan University
Cairo, Egypt

***Abstract*—Address Resolution Protocol (ARP) Poisoning attack is considered as one of the most devastating attacks in a network context. As a result of its stateless nature and lack of authentication, this protocol suffers from many spoofing attacks in which attackers poison the cache of hosts on the network. By sending spoofed ARP requests and replies. This paper proposes an approach for detecting and mitigating ARP poisoning. This approach includes three modules: Module 1 for giving permission for first time and to store information in the database. There a security measure using MD5 hash is used. Module 2 is for avoiding internal ARP. Module 3 is for detecting whether a MAC has two IPs or an IP has two MACs. The architecture includes a database that gives a great facility and support for storing ARP table information. As ARP table entries generally expire after a short amount of time. To ensure changes in the network are accounted for. Experiments were conducted on real life network environment using Ettercap to check the functionality of the proposed mechanism. The results of experiments show that the proposed approach was able to detect and mitigate ARP poisoning. Especially, whether a MAC has two IPs or an IP has two MACs.**

***Keywords*—Address Resolution Protocol (ARP); ARP detecting; ARP mitigation; ARP spoofing**

I. Introduction

The security of the network is becoming a major concern for network managers and engineers. Traditional networks are vulnerable to a variety of security flaws and assaults. In order to identify the attacks or infections, anti-virus software or an intrusion detection system (IDS) might be used. Inconsistency in communications between hosts, on the other hand, exacerbates security concerns. Security techniques that can be effectively used in network innovation are in short supply.

The fundamental reason for such assaults is a breakdown in communication between security innovation engineers and network designers [1]. A programmer will concentrate on the communication channel, collect data, decode it, and re-insert a copy message. The most important aspects to consider while constructing a secure network are confidentiality and integrity. The variety of attacks, on the other hand, continues to grow with the passage of time [2].

Address Resolution Protocol (ARP) is a mechanism for mapping Internet Protocol (IP) addresses to Media Access Control (MAC) addresses that is widely used. ARP, on the other hand, has a number of flaws. For example, there is no built-in way for a receiving node to verify the sender of a packet [3]. The ARP process excludes any authentication or integrity verification and is unconcerned whether the packet originates from a legitimate source. Any packet that has its fields filled from the allowed set of values is right.

As a result, ARP is a stateless protocol. The nodes can send ARP answers without receiving an ARP request first. Attackers use these weaknesses in ARP to carry out the ARP Cache Poisoning attack. This attack is carried out in traditional networks by poisoning a host's cache through introducing bogus or spoofed IP to MAC address mappings into the victim's ARP cache table.

The detection and mitigation of ARP Poisoning attacks are critical. Because attackers can use this attack to launch other attacks as Denial of Service (DoS), Distributed Denial of Service (DDoS) and Man-In-The-Middle (MITM) attacks. Attackers can use ARPs to communicate with the system they want to attack since they allow links between networks using IP and MAC addresses. ARP Spoofing leads to simply intercepting or dropping and not forwarding the target's packets.

The paper is organized as follows: Section II presents the related work. Section III introduces the proposed ARP approach. Testing and recorded results are shown in Section IV. Section V has the conclusion and future work.

II. Related Work

The research publications relevant to ARP attack detection and prevention will be presented in this section. A framework has been introduced in [4] to detect any rogue user in the network attempting to impersonate another user. The framework included a built-in security measures to prevent attacks on the framework itself and to prevent having a single point of failure. The framework also used robust security mechanisms including RSA. This initializes the random time intervals to prevent other devices from sending client messages. AES encryption has been used for all subsequent messages.

Authors in [5] have examined the conditions for conducting a Man in the middle (MITM) attack in networks using the Address Resolution Protocol (ARP). In addition to, methods for detecting and preventing such attacks. They presented an implementation of an ARP spoofing attack using Python and

*Corresponding Author.

C# (scapy) languages. They have provided Man-in-the-middle attacks examples such as DHCP spoofing and ICMP redirection. While authors in [6] proposed a detection algorithm that looks for differences between the real MAC Address and the response MAC Address of the ARP Packet sniffed. In addition to, a prevention algorithm for ARP Poisoning attacks using static entries in ARP table. The implementation for both algorithms has been done using Python Programming language using scapy library.

A survey study for the theory of ARP spoofing attacks and the various existing techniques proposed to defend ARP against them was presented in [7]. The study has shown that for best security measures, both detection and prevention systems should be implemented in the network with consideration to minimize the cryptography processes. They also recommended the four security requirements to develop a mechanism approach to detect/prevent ARP spoofing attacks shown in Fig. 1.

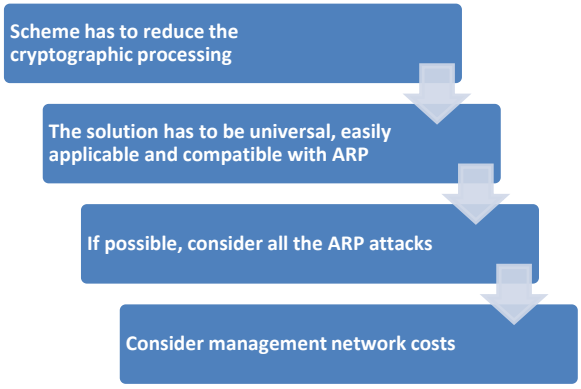

Fig. 1. ARP Detection and Prevention Requirements.

The description of ARP and how it works was discussed in [8]. The topic of ARP Spoofing and its many attacks then has been discussed. Finally, it compared several detection and mitigation approaches, as well as their benefits and drawbacks, in order to combat ARP attacks.

In [9], the authors offered three methods for detecting and preventing ARP spoofing. These methods may have few flaws or disadvantages. Within a LAN, these strategies are simple to implement. Method 1 is the most effective, Method 2 necessitates the creation of a new protocol, and Method 3 is costly and time-consuming owing to software or server deployment. Depending on the needs of the business, any of these strategies can be used.

The authors in [10], looked at various tools and strategies for detecting and preventing Address Resolution Protocol Spoofing attacks. To learn more about how a host maintains the address resolution protocol cache table with a spoofed or false media access control, MAC cache table, the ARP spoof tool was used to send a spoofed Address Resolution Protocol reply packet to a host in a local area network. Lastly, they compared the detection and prevention tools and approaches against the Address Resolution Protocol Spoofing assault in terms of their efficiency in detecting and preventing the attack as well as system performance requirements.

The ARP spoofing attack in traditional networks was investigated in [11]. They first performed ARP spoofing in an SDN network and discovered that the threat of an ARP attack is still present and has a significant impact on the network. They suggested a unique protection method for ARP spoofing based on the OpenFlow platform. The mechanism was given a theoretical analysis, and it was implemented as a module of the POX controller. On the OpenFlow platform and related SDN platforms, this significantly decreased the security danger of ARP spoofing.

A detailed survey on various solutions to mitigate ARP Cache Poisoning attack in SDN was carried out in [12]. Flow graph-based solutions, traffic pattern-based solutions, and IP-MAC Address Bindings-based solutions were all characterized in this survey. All of these solutions were assessed rigorously in terms of their functioning principles, benefits, and drawbacks. Another key element of this study was the ability to compare different systems based on numerous performance measures, such as attack Detection time, ARP response time, delay calculation at the controller, and so on.

Thus, all the pervious related works solutions considered ARP cache table in switch for either detecting or mitigating ARP poisoning. Without resolving the problem in which, the ARP cache table may be destroyed. If the switch is restarted or if an administrator action is performed.

III. Proposed ARP Approach

The proposed ARP approach in this paper consisted of three modules that will effectively detect and mitigate ARP cache poisoning in real life network. The proposed approach overview is shown in Fig. 2. Module 1 is responsible for initialization of eligible new user. Where information will be stored in the database; in addition, checking information of eligible users before going to module 2. Detecting and mitigating internal ARP will be occurred in module 2. While external ARP detection and mitigation will be in module 3. Where intruder is either an IP with two MACs or a MAC has two IPs.

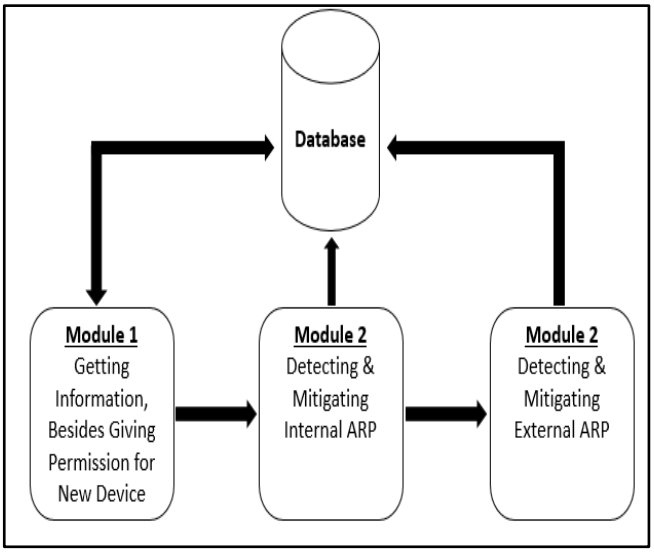

Fig. 2. Proposed Approach Overview.

The database was added to check and compare ARP table content. As the typical timeout for ARP cache is from 10 to 20 minutes [13] [14] [15], thus this made the cache to be cleared automatically. In addition to, the ARP cache table may be deleted due to switch restarting or an admin action. The database was designed to store the following ARP information:

- IP Address.
- MAC Address.
- Hash for IP + MAC.
- Port Number.
- Time.
- Date.
- Switch IP.

The database facilitates the detection of whether a MAC has two IPs or an IP has two MACs.

A. Module 1

Fig. 3 represented the flowchart for module 1. Module 1 was designed for getting and checking device data in database to decide whether forwarding device packets or not. In case which data was found, and then move to module 2. While if not found, an admin has to decide whether to give permission for that device to join network or to block device. In addition module 1 included adding a device that first time asks to join network. After assigning IP to joined device MAC, the proposed approach pop up a message showing this IP with its MAC.

If yes the following information will be stored in the database <IP, MAC, Hash for IP + MAC, Port Number, Time, Date, Switch IP> (first-give-permission) this scenario repeated each new device joined. The hash for IP and MAC has been done using MD5. A 128-bit fingerprint is generated by encoding a string of arbitrary length into an MD5 hash. The MD5 algorithm will always provide the same 128-bit hash value when encoding the same string. When storing sensitive data in databases like MySQL, MD5 hashes are typically utilized with smaller strings [16].

B. Module 2

To avoid internal ARP, module 2 was proposed as in Fig. 4. After checking device data in the database, Module 2 has to check IP and mac stored in switch ARP table. If found, the hash for that IP and MAC would be compared with the stored in the database. If the comparison result was match then, forward packet otherwise go to module 3. While if the hash for IP and MAC was not found in switch ARP table. Check for hash of IP and MAC in the database. If found, switch ARP table will be updated, then return to the start of module 2. If not found, move to module 3.

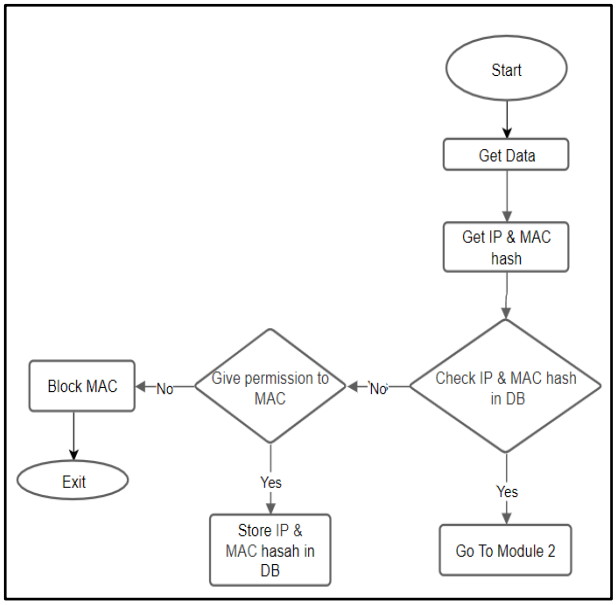

Fig. 3. Module 1 Flowchart.

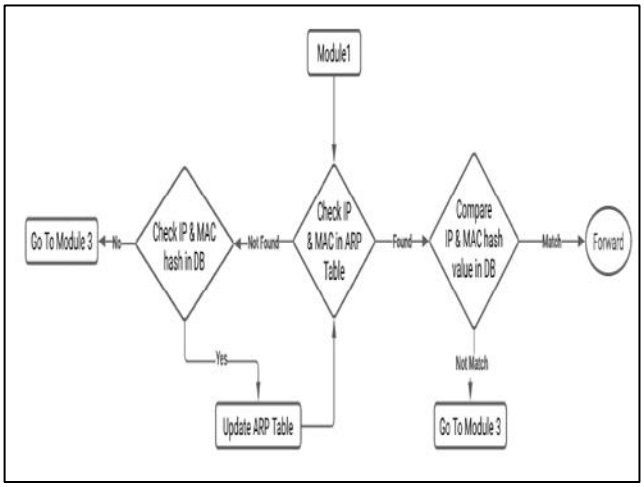

Fig. 4. Module 2 Flowchart.

C. Module 3

Module 3 as shown in Fig. 5, was designed to detect whether a MAC has two IPs or an IP has two MACs. In case of an IP has more than one MAC. The module will retrieve all MAC hashes from the database. Then it will check which MAC belongs to that IP from the database. The packet will be forwarded if MAC belongs to that IP was matched in the database. Otherwise an admin has to decide either to block MAC or to forward packet. The second case is in which a MAC has two IPs. All IPs hashes will be retrieved from the database. Checking which IP belongs to that MAC hash in the database. Besides checking which IP is active. If the result is matched and active then forward the packet. If not matched the admin has to choose either to update the database with new IP, or to block this MAC.

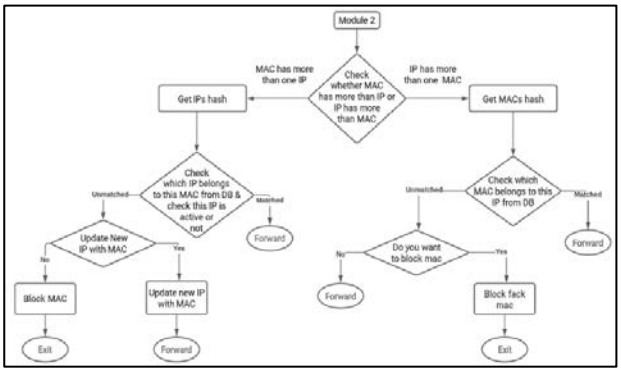

Fig. 5. Module 3 Flowchart.

IV. Testing and Results

This section will illustrate environment setup for the proposed approach. Besides presenting test cases and results, the proposed approach architecture is shown in Fig. 6. In which, two eligible devices were connected. The attacker (intruder) was trying to control the switch that was synchronized with database.

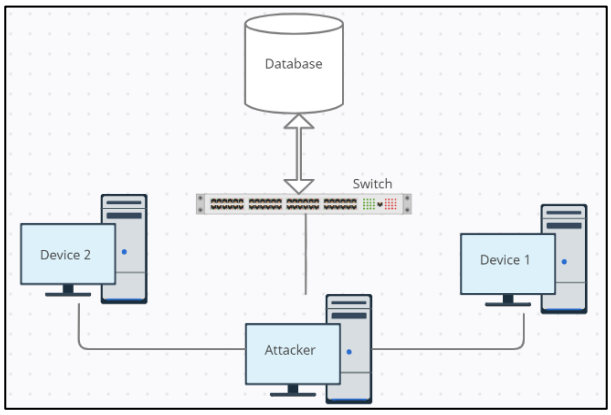

Fig. 6. Proposed Approach Architecture.

A. Environment Setup

The following tools and libraries are required to build up the environment for the suggested approach:

- VMWare Workstation Pro [17]: is a hosted hypervisor that works with x64 versions of Windows and Linux. Users can create Virtual Machines on a single device with VMWare Workstation Pro. The Virtual Machines can then be used with the physical machine at the same time.
- Kali Linux [18]: Kali Linux is a Debian-based Linux distribution operating system designed for digital forensics and penetration testing.
- Scapy [19]: is a computer network packet manipulation tool. Using scapy, packets can be faked, decoded, sent over the wire, captured, and Requests and responses can be matched.
- Netmiko [20]: is a multi-vendor library that makes it easier for Paramiko to connect to network devices via SSH.
- Ettercap [21]: can deconstruct various protocols (even encrypted ones) both actively and passively, and it has a lot of features for network and host investigation.

One kali Linux virtual machine was created to work as attacker using Ettercap.

- MySQL Database [22]: gives consumers the flexibility, scalability, and availability they need to handle the database problems of next-generation web, cloud, and communications services. The database was connected with the proposed approach using mysql.connector.
- Python [23]: Python is a general-purpose programming language with a high level of abstraction. Its design philosophy priorities code readability and makes extensive use of indentation.

B. Phase 1 "Initialization" Testing and Results

In this phase, module 1 checked the database to either allow or deny connecting a new node. Fig. 7 represented that the database was empty. In addition, the proposed approach popped up a message for adding new node. Fig. 8 presented the database after storing a new device information in which, MAC [00:0c:29:dd:29:c8] with IP [172.19.15.18] was successfully added.

C. Phase 2 "Authentication" Testing and Results

Besides matching between the database and the ARP table for authenticating IP and MAC, Phase 2 included checking for IP and MAC in the ARP table or not, which was happened in module 2. Check for IP with its MAC in ARP table. If it is found in the ARP table then the hashed value for IP and MAC is compared with the database.

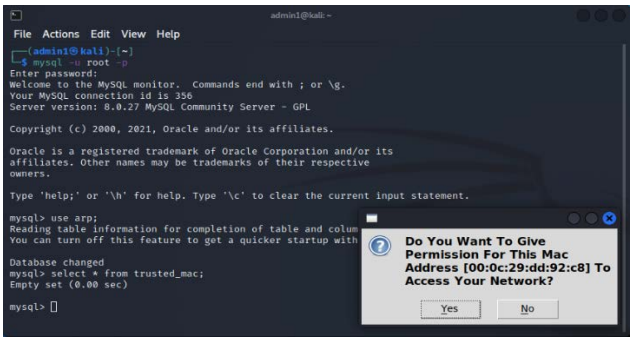

Fig. 7. Empty Database.

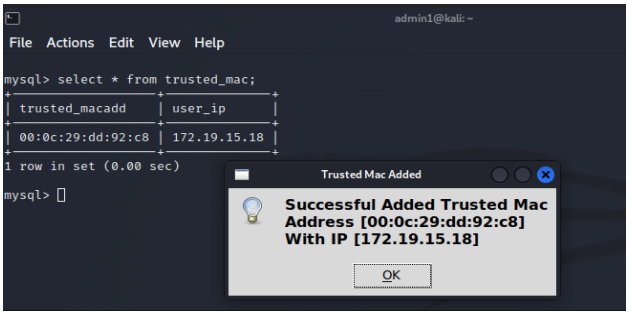

Fig. 8. Adding New Device in the Database.

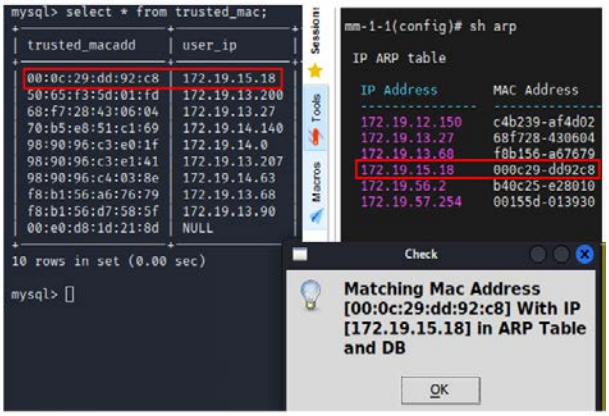

Fig. 9. Check for MAC with IP in Both ARP Table and Database.

The proposed approach popped up a message for MAC [00:0c:29:dd:29:c8] with IP [172.19.15.18] in case of matching and founding in both ARP table and the database as in Fig. 9. If not matched go to module 3.

Fig. 10 shows the case in which MAC [50:65:f3:5d:01:fd] with IP [172.19.13.200] was not found in ARP table. The proposed approach would check for stored MAC and IP hashes in the database. If not found then, it would move to module 3. While in case of found, the proposed approach would update the ARP table automatically in Fig. 11.

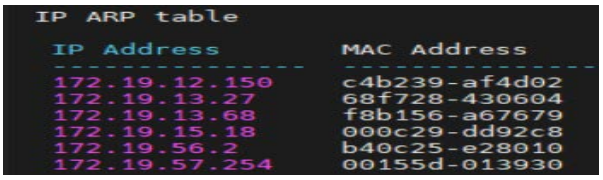

Fig. 10. ARP Table before Update.

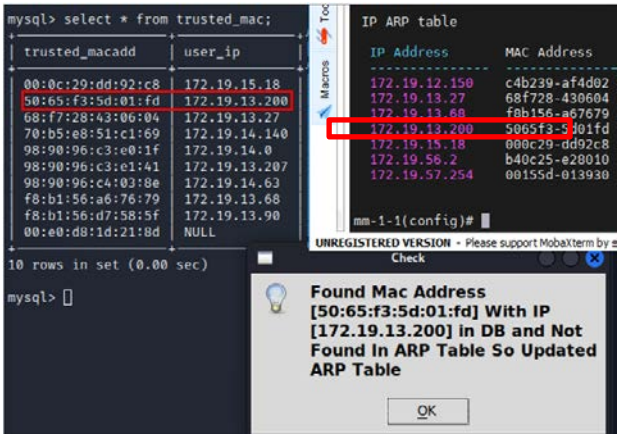

Fig. 11. ARP Table after Updating.

D. Phase 3 "Poisoning Detection & Classi-fication" Testing and Results

Module 3 where classification and detection of whether an IP has two MACs or a MAC has two IPs, was tested in this phase. The first case included testing of a MAC has more than one IP. All IPs hashes was retrieved from the database. Then, the original IP belonged to this MAC was checked in the database as in Fig. 12. Where the ARP table showed MAC [f8:b1:56:a6:76:79] with IP [172.19.12.201] and [172.19.13.68]. While the database showed that IP for MAC [f8:b1:56:a6:76:79] was [172.19.13.68]. The packet of the active and original IP would be forwarded and the other IP would be released. While if not active the admin has to decide either to update the database with new IP or to block the MAC as in Fig. 13.

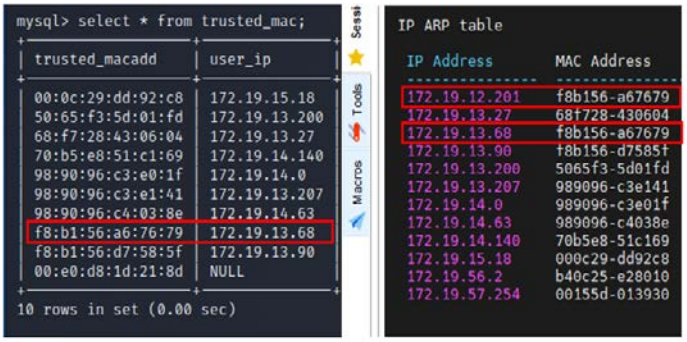

Fig. 12. Check for Original IP.

The second case was for one IP with more than one MAC. The original MAC was checked from the database after getting all MACs hashes. The packet of the matched one was forwarded. But if not matched, the proposed approach popped up a message alerting that ARP poisoning was happened as in Fig. 14. Where IP [172.19.13.207] has original MAC [98:90:96:c3:e1:41] and fake MAC [00:0c:29:dd:92:c8]. Thus, the admin has to decide either to block fake MAC or not.

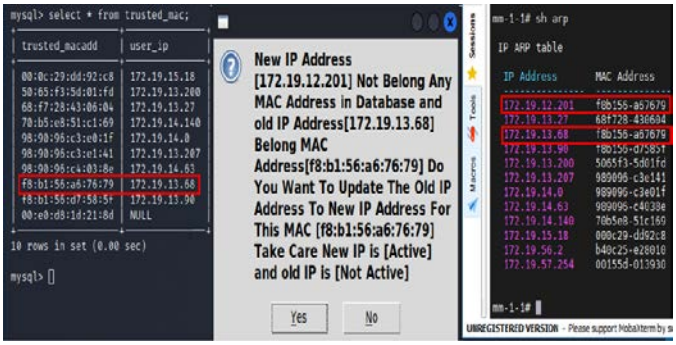

Fig. 13. Update IP or Block MAC.

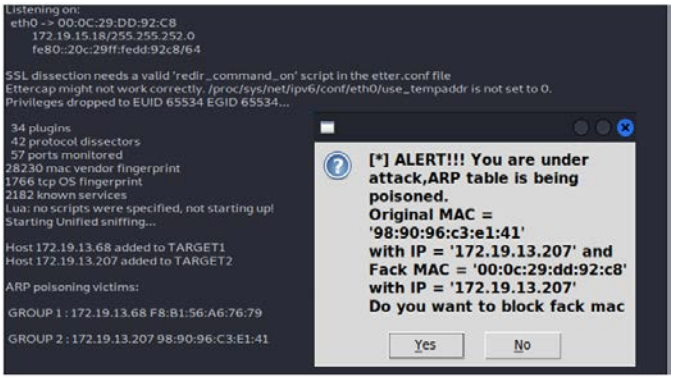

Fig. 14. IP with Two MACs.

V. Conclusion and Future Work

This research proposed a new approach for detecting and mitigating ARP poisoning attack. The database was added to the proposed approach architecture for storing information from switch. Avoiding missing the data as typical timeout for ARP cache is from 10 to 20 minutes. Furthermore, the ARP cache table may be deleted due to switch restarting or an admin action. The proposed approach has included three modules. The first one was for getting data and storing it in the database. Besides giving permission for new joining device, the second one was for checking IP and MAC in both ARP table and comparing them with the stored records in the database.

The comparison has been done for hashing of IPs and MACs that were stored in the database. The used function for hashing IP and MAC was MD5 function. The third module was to detect either a MAC has more IPs or an IP has more MACs. Where a successful detection for both cases has been done and presented in testing and results shown in Fig. 12, 13 and 14. The ARP poisoning attack was generated using Ettercap.

For future work, the proposed approach may be applied for detecting ARP poisoning in Software Defined Network (SDN). The proposed approach may also apply for other attacks as Man-In-The-Middle (MITM) attack.

References

[1] C. D.Francis Xavier and C.Divya, "Address Resolution Protocol Based Attacks: Prevention and Detection Schemes," in *Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018)*, Springer, 2019, p. 247–256.

[2] M. Conti, N. Dragoni and V. Lesyk, "A Survey of Man In The Middle Attacks," IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2027 - 2051, 2016.

[3] Z. Shah and S. Cosgrove, "Mitigating ARP Cache Poisoning Attack in Software-Defined Networking (SDN): A Survey," Electronics , 2019.

[4] K. D. Bhattacharya, N. S. H. Karthick, P. Suresh and N. Bhalaji, "DetecSec: A Framework to Detect and Mitigate ARP Cache Poisoning Attacks," in Evolutionary Computing and Mobile Sustainable Networks, Springer, 2022, p. 997–1007.

[5] P. P. Stepanov, G. V. Nikonova, T. S. Pavlychenko and A. S. Gil, "The problem of security address resolution protocol," Journal of Physics: Conference Series, 2021.

[6] A. Majumdar, S. Raj and T.Subbulakshmi, "ARP Poisoning Detection and Prevention using Scapy," Journal of Physics: Conference Series, 2021.

[7] S. Hijazi and M. S. Obaidat, "Address resolution protocol spoofing attacks and securityapproaches: A survey," Wiley, 2019.

[8] R. Vaishnavi and S. Goyal, "A Detailed Survey for Detection and Mitigation Techniques against ARP Spoofing," in 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2020.

[9] G. Agrawal and V. Chennai, "Detection and Prevention of ARP-Spoofing Attacks," International Journal of Engineering Research & Technology (IJERT), vol. 8, no. 10, 2019.

[10] T. A. ASSEGIE and P. S. NAIR, "COMPARATIVE STUDY ON METHODS USED IN PREVENTION AND DETECTION AGAINST ADRESS RESOLUTION PROTOCOL SPOOFING ATTACK," Journal of Theoretical and Applied Information Technology, vol. 97, no. 16, 2019. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[11] X. Jing, C. Zhiping, H. Gang and X. Ming, "An Active Defense Solution for ARP Spoofing in OpenFlow Network," Chinese Journal of Electronics, vol. 28, no. 1, 2019. K. Elissa, “Title of paper if known,” unpublished.

[12] Z. Shah and S. Cosgrove, "Mitigating ARP Cache Poisoning Attack in Software-Defined Networking (SDN): A Survey," Electronic, 2019.

[13] "The Windows Club," 8 September 2021. [Online]. Available: https://www.thewindowsclub.com/how-to-clear-arp-cache-in-windows. [Accessed 29 March 2022].

[14] "Network Command Reference," [Online]. Available: https://netref.soe.ucsc.edu/node/20#:~:text=The%20typical%20lifetime%20of%20an,20%20minutes)%20have%20been%20observed.. [Accessed 29 March 2022].

[15] "parsons-technology.com," [Online]. Available: https://parsons-technology.com/how-do-i-scan-arp-on-windows/. [Accessed 29 March 2022].

[16] "MD5 Hash Generator," [Online]. Available: https://www.md5hashgenerator.com/. [Accessed 1 Augast 2021].

[17] [Online]. Available: https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html?msclkid=035ab3aeb1981 1ecbf27f7efbb37856b. [Accessed 28 July 2021].

[18] [Online].Available: https://www.kali.org/docs/installation/?msclkid=e7d930bab19a11ecb69ac0998f02a7ce. [Accessed 29 July 2021].

[19] [Online].Available: https://scapy.net/?msclkid=0a6c51adb19c11eca81cb4bd35facb34. [Accessed 29 July 2021].

[20] [Online].Available: https://pypi.org/project/netmiko/?msclkid=dde5ec40b19c11ecbfdcce04f1742acf. [Accessed 29 July 2021].

[21] [Online].Available:https://www.ettercap-project.org/?msclkid=c140605fb19d11ec9eb5e091e216729e. [Accessed 30 July 2021].

[22] [Online].Available: https://www.mysql.com/?msclkid=79374b5ab19e11ec807d6d6e8b4f38b6. [Accessed 30 July 2021].

[23] [Online].Available: https://www.python.org/?msclkid=fb10350cb1a211ec86584fd4b5ede560. [Accessed 30 July 2021].