ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

An Effective Prevention Approach Against ARP Cache Poisoning Attacks in MikroTik-based Networks

Ekarin Suethanuwong[1]

ABSTRACT

Nowadays, leading manufacturers of enterprise-grade networking devices offer the dynamic ARP inspection (DAI) feature in their Ethernet Switches to detect and prevent ARP cache poisoning attacks from malicious hosts. However, MikroTik Ethernet switches do not yet support this feature. Within MikroTik-based networks, three potential approaches exist to prevent ARP cache poisoning attacks, each with drawbacks. This paper proposes an innovative approach called Gateway-controlled ARP (GCA) to prevent ARP cache poisoning attacks on a router-on-a-stick (RoaS) network using MikroTik networking devices, where a single router performs inter-VLAN routing through one physical interface. With this approach, all Ethernet switches are configured to forward ARP messages from hosts directly to the router for inspection and handling. A RouterOS script based on the GCA approach was implemented and executed on the router to handle all incoming ARP requests from any host in all VLANs, ensuring all hosts receive legitimate ARP responses from the router. This approach can effectively prevent spoofed ARP packets sent by malicious attackers. This approach was tested and evaluated on an actual RoaS network, focusing on processing time, CPU Load, and response time. The evaluation results show that the approach effectively prevents ARP cache poisoning attacks.

Article information:

Keywords: Address Resolution Protocol, ARP Spoofing, ARP Cache Poisoning, MikroTik-based Network

Article history:

Received: April 13, 2024
Revised: July 25, 2024
Accepted: October 10, 2024
Published: November 16, 2024 (Online)

DOI: 10.37936/ecti-cit.2025191.256401

1. INTRODUCTION

Currently, the Internet has become an integral part of human life. It is evident that most people typically carry mobile devices (e.g., smartphones, tablets, and laptops) to connect to the Internet for various activities, including work in organizations. According to the survey [1], the number of internet users worldwide has been steadily growing and reached approximately 5.35 billion in January 2024, accounting for 66.2 percent of the global population. In the last decade, many companies have allowed users to bring their devices to connect to their local area networks (LANs) for business activities in intranets and the Internet [2]. However, this practice puts corporate data at risk of cyberattacks, defined as malicious attempts to steal, alter, destroy information, or even breach information systems [3].

Recently, cyberattacks have been increasingly critical to information systems. Typically, attackers exploit weaknesses in information systems to carry out cyberattacks. The man-in-the-middle (MITM) attack is one of the most dangerous cyberattacks, wherein an attacker secretly intercepts communication between two hosts, allowing them to capture and manipulate sensitive personal information such as credit card numbers and login credentials [4]. In 2017, Equifax, one of the largest credit reporting agencies, suffered a data breach that exposed the financial information of nearly 150 million people in the United States through an MITM attack [5].

Regarding the MITM attack, two hosts believe that they are communicating directly with each other, unaware that an attacker intercepts and relays messages between them. The attacker in the MITM attack can compromise three security aspects (data confidentiality, data integrity, and data availability) [6] as follows. Regarding data confidentiality, the attacker eavesdrops on the communication between the two hosts and steals sensitive personal information. With

[1] The author is with the Department of Information and Digital Technology Management, Faculty of Commerce and Management, Prince of Songkla University, Trang Campus, Thailand, Email: ekarin.s@psu.ac.th

data integrity, the attack intercepts the communication and modifies the transferred messages between the two hosts. In data availability, the attacker disrupts the communication between the two hosts by blocking or destroying the transferred messages.

There are several methods that attackers can employ to conduct the MITM attack, including ARP cache poisoning, DNS spoofing, DHCP spoofing, IP spoofing, rogue access point, email hijacking, and HTTPS spoofing [6]. This work primarily focuses on ARP cache poisoning due to its vulnerability, which can be easily exploited for the MITM attack if the attacker gains access to the data link layer. ARP is a network protocol of the data link layer that maps IP (Internet Protocol) addresses to MAC (Medium Access Control) addresses. However, ARP was designed without security considerations. It lacks authentication mechanisms and integrity verification processes [7]. ARP itself is a stateless protocol, allowing an attacker to send an ARP reply packet to poison an ARP cache table of any host without first receiving an ARP request packet. ARP cache poisoning not only enables man-in-the-middle (MITM) attacks but also allows other types of attacks, such as distributed denial-of-service (DDoS) and host impersonation [8].

Currently, many enterprise-grade networking companies (e.g., Cisco, Juniper, and Huawei) offer the dynamic ARP inspection (DAI) feature in their Ethernet switches for effectively detecting and preventing ARP cache poisoning attacks [6] [9-10]. This feature works with the DHCP snooping feature, which adds and updates entries for untrusted hosts into the DHCP snooping binding table when the Ethernet switch receives specific DHCP messages [10]. Each DHCP snooping binding table entry contains the MAC address, leased IP address, lease time, VLAN number, and interface information associated with a host. The Ethernet switch with the DAI feature verifies ARP packets against the information in the DHCP snooping binding table. If there is no match, the switch will reject the ARP packets.

Nowadays, MikroTik networking devices are widely used in computer networking of small and medium enterprises (SMEs) due to their low prices and high stability [11-13]. MikroTik routers have been extensively adopted as internet gateways [12] in router-on-a-stick (RoaS) networks. Currently, the DAI feature is not yet available in MikroTik Ethernet switches. The mechanism of the DHCP snooping feature in MikroTik Ethernet switches does not create a DHCP snooping binding table required by the DAI feature. This paper proposes an approach called Gateway-controlled ARP (GCA) to effectively prevent ARP cache poisoning attacks for all hosts in a router-on-a-stick (RoaS) network that uses MikroTik networking devices, where a single router serves as both the internet gateway and the default gateway for all subnets. The main idea of this approach is to control ARP packets through their default gateways located at the router rather than allowing direct communication between hosts.

This paper is structured as follows. Section 2 provides overviews of ARP cache poisoning attacks. Section 3 presents related works. Section 4 elaborates on the DAI mechanism and its disadvantages. Section 5 discusses the existing approaches for ARP cache poisoning prevention in MikroTik-based networks. Section 6 introduces the proposed GCA approach. Section 7 describes the experimental setup and evaluates the results. Finally, Section 8 covers the conclusions and future work.

2. ARP CACHE POISONING ATTACKS

2.1 Address Resolution Protocol (ARP)

In LANs based on IP version 4 (IPv4), ARP is used to translate the IP address of an intended host into the corresponding physical machine address, known as the MAC address. Fig. 1 describes the ARP mechanism. Host A begins to look up the destination MAC address in its ARP cache table, which contains pairs of IP and MAC addresses for hosts in the same subnet. If the destination MAC address is not in the ARP cache table, Host A starts sending out a broadcast ARP request packet, indicating that the destination MAC address is FF-FF-FF-FF-FF-FF. Host A will receive a unicast ARP reply packet with the destination MAC address if the destination host exists in the same subnet. In this case, the source host takes a pair of the IP and MAC addresses from the unicast ARP reply packet to update into its ARP cache table. The entries reside in the ARP cache for a specified period (typically a few minutes) determined by the type of operating system [14].

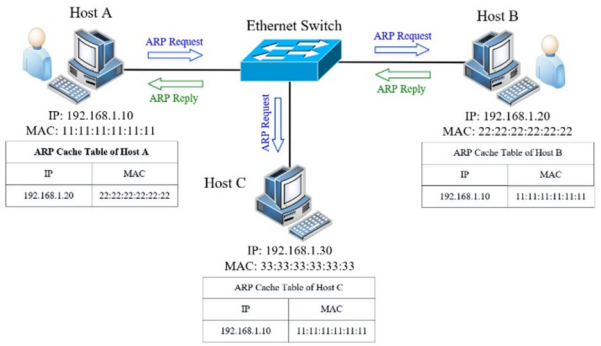

Fig.1: *The Mechanism of ARP Request and Reply Packets.*

ARP is a data link layer protocol or Layer 2 protocol [10]. Fig. 2 depicts the ARP packet format, which consists of an Ethernet header, payload data that contains ARP data and padding, and an Ethernet trailer, including the Frame Check Sequence filed [9] [14]. The fields of the ARP packet format in the ARP data are detailed as follows. The Hardware

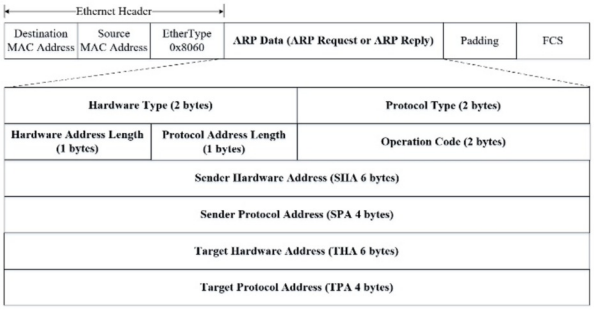

Fig.2: *ARP Packet Format on Ethernet.*

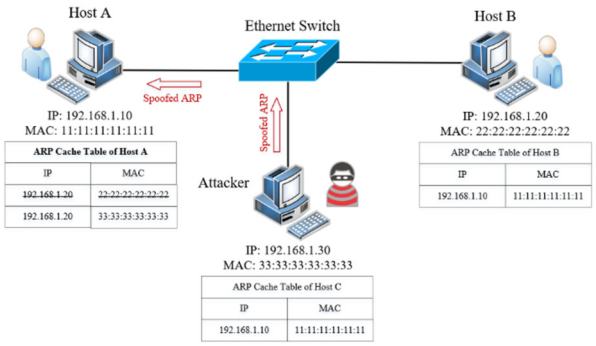

Fig.3: *ARP Cache Poisoning in Host A.*

Type field is specified to be 1 for Ethernet. The Protocol Type field contains 0x0800 for Internet Protocol (IP). The Hardware Address Length and Protocol Address Length fields contain 6 and 4 for the length of MAC and IP addresses in bytes, respectively. The Operation Code field indicates ARP type: (1) ARP request and (2) ARP reply. The Sender Hardware Address (SHA) and Sender Protocol Address (SPA) fields are the MAC address and IP address for the sender Host, respectively. Likewise, the Target Hardware Address (THA) and Target Protocol Address (TPA) fields contain the MAC address and IP address of the destination host, respectively.

2.2 ARP Cache Poisoning Attacks

The vulnerability of ARP lies in its stateless nature, implying that a host can accept any ARP reply without a prior ARP request from itself [9] [15] [16]. Additionally, ARP can update the ARP cache of a receiving host with the IP and MAC addresses of a sender host from either an ARP request or reply packet. ARP cache poisoning, also known as ARP spoofing [17], occurs when an attacker sends either spoofed ARP request packets or spoofed ARP reply packets with incorrect pairs of IP and MAC addresses to a target host, thereby poisoning the ARP cache of that target host. In Fig. 3, the network comprises Host A, Host B, an attacker, and an Ethernet switch. When Host A, possessing the IP address of Host B, intends to communicate with Host B, it checks its ARP cache for the MAC address of Host B. In a scenario of an ARP cache poisoning attack, the attacker periodically sends an ARP request or reply packet to Host A, providing false information, i.e., an IP address (192.168.1.20) and a MAC address (33:33:33:33:33:33). Consequently, Host A updates its ARP cache with the IP address of Host B associated with the MAC address of the attacker. In other words, when Host A attempts to communicate with Host B, it inadvertently transmits messages to the attacker. Host A uses the attacker's MAC address stored in its ARP cache instead of Host B's.

3. RELATED WORKS

ARP is inherently a stateless protocol and lacks an authentication mechanism to validate ARP messages [15]. This vulnerability enables attackers to perform MITM attacks via ARP cache poisoning. Over the years, several software tools have been developed specifically for detecting ARP cache poisoning attacks, such as Wireshark, Ettercap, Snort, Arpwatch, and Antidote [15] [17-19]. Network administrators rely on the detection results from such software tools to identify malicious pairs of IP and MAC addresses. In addition to these software tools, numerous research papers have investigated approaches for detecting ARP cache poisoning [14] [16]. For example, the authors in [16] employed the ICMP protocol to detect ARP cache poisoning attacks. However, relying solely on automatic detection mechanisms is insufficient to protect LANs from ARP cache poisoning attacks, as network administrators may be too late to detect and prevent such attacks after attackers have already obtained sensitive information such as credentials or credit card numbers [15]. Therefore, this paper primarily focuses on prevention mechanisms for ARP cache poisoning attacks. From the past until now, many approaches have been investigated for preventing ARP cache poisoning attacks, as outlined below.

3.1 Cryptography-based Approach

One of several ideas for preventing ARP cache poisoning attacks is to secure ARP using cryptography [14] [20-22]. S-ARP (Secure ARP) [20] is a cryptography-based approach that authenticates ARP replies through public-key cryptography. Each host on the same LAN has a public/private key pair certified by a local trusted party called the Authoritative Key Distributor (AKD). There are three main drawbacks to this approach. Firstly, S-ARP only authenticates ARP replies, leaving the ARP cache vulnerable to spoofed ARP requests. Secondly, S-ARP requires modification of all hosts to add functions for encrypting, decrypting, generating public and private key pairs, and sending extra packets to obtain the public key of an ARP request from the AKD every time an ARP reply packet arrives at a host. Thirdly, S-ARP does not support the dynamic assignment of

IP addresses from a DHCP server. Consequently, a customized version of the DHCP server, named S-DHCP, is necessary to communicate with the S-ARP server.

Other similar cryptography-based approaches, such as LS-ARP in [22], also require modification of the standard ARP protocol on all hosts to authenticate ARP messages using cryptographic algorithms. S. M. Morsy and D. Nashat [14] proposed a detection and prevention approach for ARP cache poisoning attacks called D-ARP. It was developed based on the following two conditions. (1) The original ARP protocol cannot be modified. (2) It does not depend on a centralized server, thus avoiding a single point of failure. A drawback of this approach is that every host within the network needs to install the D-ARP function. Consequently, it is unsuitable for dynamic environments where hosts can be added or removed from the networks dynamically.

3.2 Static IP-MAC Entry in ARP Cache

Although manually adding the IP and MAC address pairs of all hosts in the same LAN to each host's ARP cache is simple and can effectively prevent ARP poisoning attacks, since ARP message exchanges are no longer needed to acquire MAC addresses of other hosts, this approach is not scalable for a large network segment [8] [23-24]. Additionally, it is unsuitable for a dynamic environment where hosts are frequently added or removed from the network. It would require a heavy workload for network administrators to continually add or update the ARP caches of all hosts throughout the network [8] [25]. Abdel Salam *et al.* [25] proposed an approach for automatically adding static entries of IP-MAC address pairs into the ARP caches of all hosts.

M. Data [26] also proposed an approach to manage ARP cache entries automatically. It validates pairs of IP and MAC addresses in ARP cache entries against an allowlist containing trusted pairs of IP and MAC addresses derived from the *arping* tool [27]. The main drawback is that checking all IP addresses in the ARP caches of all nodes generates a lot of broadcast ARP request packets, even without an ongoing ARP cache poisoning attack. Additionally, the approaches in [25-26] have the drawback of being host-based, requiring additional software on all hosts.

3.3 Secondary ARP Cache-based Approach

In the past, several research works introduced an additional ARP cache known as a secondary ARP cache, which contains correct pairs of IP and MAC addresses used for validating data in a primary ARP cache [28-30]. N. Tripathi and B. M. Mehtre [28] proposed an ICMP-based secondary cache approach for detecting and preventing ARP cache poisoning. This approach adds a secondary ARP cache that permanently stores correct and updated IP and MAC address pairs from ARP reply packets in a text file, ensuring the validity of the primary ARP cache on every host and gateway. The main disadvantage of this approach is that ARP request packets spoofed from an attacker, which can poison a primary ARP cache, are not considered. Furthermore, this approach is unsuitable for networks with uncontrollable hosts because hosts with their firewalls enabled might block ICMP packets.

S. Kumar and S. Tapaswi [29] proposed a centralized detection and prevention approach for ARP cache poisoning by utilizing an additional secondary ARP cache for each host and an ARP Central Server (ACS). With this approach, any change in the primary ARP cache of a host is constantly monitored and detected against its secondary ARP cache, validated by sending a request packet based on TCP (Transmission Control Protocol) to the ACS. The main drawback of the proposed approach is the need to add a dedicated host to run only the ACS in a network segment. Another downside, like [28], is that installing additional software to run this approach on every host and gateway with various platforms in a large-scale network segment may not be practical.

3.4 Voting-based Approach

S. Y. Nam *et al.* [31] proposed an enhanced version of ARP, employing a voting-based mechanism to prevent ARP poisoning-based MITM attacks named MR-ARP. MR-ARP aims to maintain backward compatibility with the existing ARP standard while avoiding using cryptography techniques and central servers. Y. Zhao et al. [32] proposed an enhanced approach named EMR-ARP to improve the voting process by integrating computational puzzles based on public key cryptography to overcome the limitations of MR-ARP in both wired and wireless LANs. The significant drawback of both the MR-ARP and EMR-ARP approaches is that installing the software to run the MR-ARP process on every host and gateway is impractical for large-scale networks and unsuitable for wireless LANs, especially with the prevalence of personal mobile devices where installing the software may not be feasible.

P. Arote and K. V. Arya [33] introduced a voting-based approach to prevent ARP cache poisoning. Like [16], they utilized the ICMP protocol at a central server (CS) to detect ARP cache poisoning on any host, explicitly focusing on spoofed ARP request packets. This approach lacks a mechanism for electing the CS, implying that a network administrator must manually define it. Like [31-32], the main drawback of this approach is that it is a host-based solution. Another limitation is that using the ICMP protocol, as seen in [15] and [16], requires ensuring that the firewalls of all hosts are allowed incoming ICMP echo request packets.

3.5 Controller-based Approach in SDN

This section focuses on techniques for preventing and mitigating ARP poisoning in software-defined networking (SDN). SDN represents a novel approach that decouples the control and data planes [34-36]. The central component of SDN architecture is the controller, which serves as a centralized device governing all underlying networking devices (such as switches and routers) via the OpenFlow protocol. ARP cache tables in SDN, like those in traditional networks, are susceptible to poisoning. H. Y. Ibrahim *et al.* [35] proposed an approach to detect and prevent ARP spoofing attacks while eliminating broadcast ARP request traffic to all other hosts. Unlike the approach in [34], this approach involves maintaining a main table containing the IP-MAC pair information of all hosts. This approach shares similarities with the work in [36], where the controller serves as an ARP proxy, responding to ARP request packets with corresponding ARP reply packets instead of relying on the destination host. However, it surpasses [36] in its automation, as the IP-MAC pair information in the main table can be automatically obtained from DHCP offer packets, whereas in [36], the main table requires manual configuration. This approach extends the POX controller with a mechanism that modifies the L2 learning component and extracts IP and MAC addresses from DHCP offer packets, storing them in the main table. Initially, the controller installs flow rules on the underlying switches to direct ARP request packets to the controller.

3.6 Switch-based Approach

M. Ren *et al.* [37] proposed a switch-based approach for detecting MITM attacks conducted through ARP cache poisoning. With this method, they introduced an algorithm implemented on an Ethernet switch to monitor incoming and outgoing ARP reply packets at the interfaces connected to end devices, known as access ports. The drawback of this approach is its vulnerability to ARP cache poisoning of legitimate hosts within a short period during the ARP poisoning detection. Consequently, it cannot wholly prevent ARP cache poisoning attacks targeting hosts. In addition, the approach overlooks spoofed ARP request packets. Despite these drawbacks, a switch-based solution offers advantages over a host-based one. Ethernet switches can autonomously block traffic at receiving interfaces upon detecting incoming spoofed ARP traffic.

4. DYNAMIC ARP INSPECTION

Currently, Ethernet switches from well-known enterprise-grade networking companies such as Cisco, Juniper, and Huawei include the Dynamic ARP Inspection (DAI) feature to prevent ARP cache poisoning [6] [9] [10]. This feature works in conjunction with the DHCP snooping enabled. The DAI feature intercepts and validates all incoming ARP request and reply packets from interfaces configured as untrusted. Interfaces connected to end devices (e.g., laptops or desktops) are typically untrusted, while those linking to other network devices are trusted. DAI determines the validity of incoming ARP packets based on valid bindings of IP and MAC addresses in the DHCP snooping binding table of the Ethernet switch. In essence, an incoming ARP packet is considered invalid and discarded if the sender protocol and hardware addresses do not match any entry in the same VLAN or if the sender hardware address specified in the ARP packet's body does not match the source MAC address in the Ethernet header. Otherwise, the Ethernet switch forwards the incoming ARP packet to its destination host. The drawback of the DAI feature is that each Ethernet switch must maintain its DHCP snooping binding tables in line with the DHCP lease table of a DHCP server. Inconsistent data in these tables requires a network administrator to release and renew the IP addresses of hosts in the network, particularly when an Ethernet switch is changed.

5. EXISTING APPROACHES OF MIKROTIK

MikroTik Ethernet switches lack the DAI feature, even if they support DHCP snooping [38]. Nevertheless, three possible approaches exist to prevent ARP cache poisoning attacks in MikroTik-based networks, each with its drawbacks as outlined below.

5.1 Static ARP Tables

Setting a static ARP table is a highly effective approach for preventing ARP cache poisoning attacks. It involves creating a static ARP table on each host and the gateway. However, this approach comes with significant administrative overhead. Any network changes require manual updates to the ARP tables on all hosts and the gateway. Consequently, this approach proves impractical for organizations managing large networks with numerous hosts.

5.2 Local Proxy ARP

Unlike static ARP tables, this approach does not rely on host-based solutions, meaning that adding configuration on individual hosts is not required. Instead, all Ethernet switches within each subnet must use private VLANs, also known as port isolation. It ensures that Ethernet switch ports are isolated from each other and only communicate with a designated uplink port. Additionally, the gateway interface connecting within the same subnet must be set as a local proxy ARP to handle the forwarding of all traffic transmitted between hosts within the subnet. However, this approach has the drawback of network performance degradation, as communication between

hosts in the same subnet must pass through the gateway.

5.3 Bridge Filter Rules

Similar to the local proxy ARP approach, this approach is also not a host-based solution, but the difference lies in the fact that it does not require setting the gateway to act as a local ARP proxy. Instead, Ethernet switch ports directly connecting to hosts must be configured with bridge filter rules for incoming ARP traffic. These rules specify that the Ethernet switch port should accept all incoming ARP request packets and ARP reply packets where the source IP and MAC addresses in the ARP body and Ethernet overhead match the IP and MAC addresses of the host connected to that Ethernet switch port. Otherwise, the Ethernet switch drops the other incoming ARP packets from the host. The primary issue with this approach is network throughput degradation, as all Ethernet switch ports directly connected to hosts forward all incoming traffic through the Ethernet switch's CPU instead of its built-in switch chip. It occurs because the bridge filter rules can only be processed on the Ethernet switch's CPU, necessitating the disabling of hardware offloads for all Ethernet switch ports directly connecting to hosts.

6. PROPOSED APPROACH

This section presents a new approach called Gateway-controlled ARP (GCA) to overcome the previously mentioned restrictions for preventing the ARP cache poisoning attack on a RoaS network based on MikroTik networking devices. Note that a RoaS network means that one router performs inter-VLAN routing for VLANs via a single router interface. The GCA approach was designed under three conditions as follows: (1) It is not a host-based solution, meaning hosts do not need additional software or configuration; (2) All traffic except ARP traffic can be directly communicated between hosts in the same subnet without the need for a local ARP proxy on the gateway's interface; (3) All ports of a MikroTik Ethernet switch that directly connect to hosts must forward incoming traffic through its built-in switch chip, not the CPU (Central Processing Unit) of the Ethernet Switch. The GCA approach defines the configurations of the MikroTik router and Ethernet switches in a RoaS network described below.

6.1 Configuration in MikroTik Ethernet Switch

In the RoaS network using MikroTik Ethernet switch, all switches must be configured to forward all incoming ARP traffic from connected hosts to the router acting as a gateway. Consequently, only the router handles each ARP request packet from hosts by returning an ARP reply packet with the correct binding of IP and MAC addresses. This approach resembles the one proposed in [35] for Software-Defined Networking (SDN), where the switches are only required to forward ARP packets to the SDN controller to inspect poisoned ARP packets and respond to ARP packets. All Ethernet switch ports must be configured with switch rules that process on the built-in switch chip to forward incoming ARP traffic to the router. Furthermore, the hardware offloads of all Ethernet switch ports must remain enabled, ensuring the built-in switch chip forwards all incoming traffic rather than relying on the CPU.

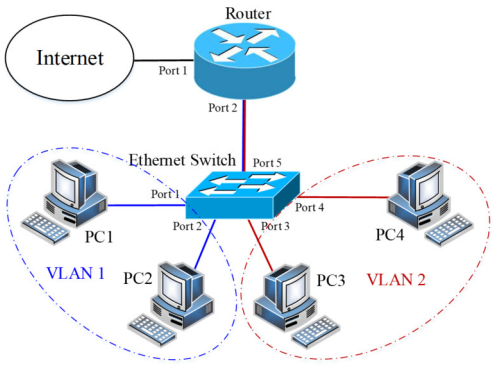

Fig.4: *An Example of a RoaS network with 2 VLANs.*

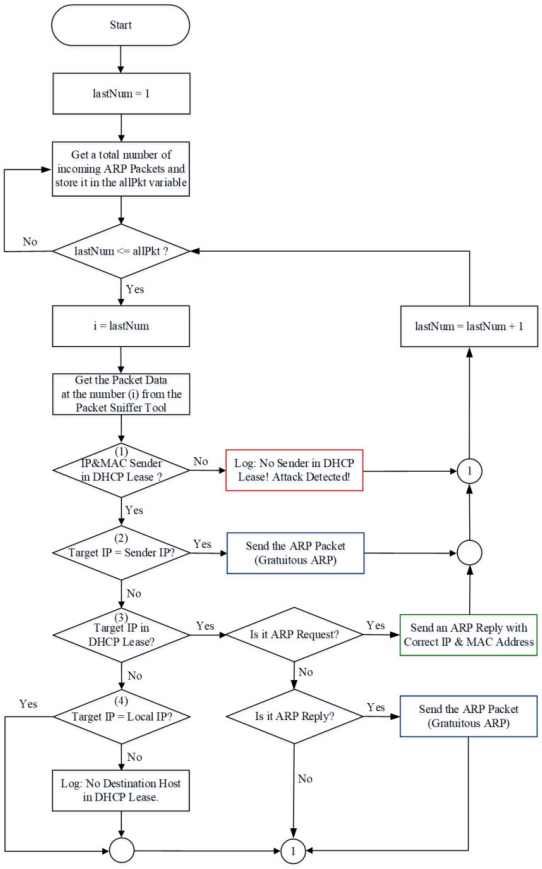

Fig.5: *The flowchart of the RouterOS script.*

6.2 Configuration in MikroTik Router

In a RoaS network, a single MikroTik router acts as the gateway for all subnets or VLANs, as illustrated in Fig. 4. For the proposed approach, a DHCP server that automatically provides IP addresses and other network parameters to hosts within each VLAN is set up on the MikroTik router. The DHCP server stores all IP and MAC address bindings in the DHCP lease table. The ARP property on the gateway interface is set to reply-only to protect against ARP cache-poisoning attacks at the gateway. In other words, the gateway only responds with its IP and MAC addresses to an ARP request packet from a sender having an IP-MAC address pair found in the gateway's ARP table. However, if the sender's IP-MAC address pair is unavailable in the gateway's ARP table, the gateway will not return an ARP reply packet. In this configuration, no ARP requests are initiated from the gateway interface. Entries in the ARP table can be statically configured by a network administrator or dynamically added by the DHCP server when leasing an IP address. The former case is used when an IP address is manually set on a host, whereas the latter is done by enabling the add-arp parameter of the DHCP server.

6.3 RouterOS Script in MikroTik Router

The packet sniffer tool on the router is configured to capture only incoming ARP packets at the gateway interface. At startup, the operating system of the MikroTik router, known as RouterOS, executes a RouterOS script. The RouterOS script, called GCA, consists of instructions on responding to incoming ARP request packets that inquire about the MAC address of a host. It sends an ARP reply packet with the correct IP and MAC addresses binding. Fig. 5 illustrates the flow of processes in the script. The script continuously reads and analyzes each incoming ARP packet from the packet sniffer tool, performing the following processes in order.

(1) Verifying the sender's IP and MAC addresses against the DHCP lease table: The process checks the sender's IP and MAC addresses (in the SPA and SHA fields) against the DHCP lease table. If a match is found, it examines the target addresses (TPA and THA fields). Otherwise, the sender is flagged as a potential attacker. In the latter case, a log message can alert a network administrator.

(2) Identifying the incoming ARP packet as gratuitous: This process compares the TPA field's target IP address with the SPA field's sender IP address. The incoming ARP packet is gratuitous if the target IP address matches the sender IP address. The script then proceeds to send the ARP packet to the gateway interface, facilitated by employing the Traffic-Generator tool in RouterOS. Conversely, the subsequent process will proceed if the target IP address is not identical to the sender IP address.

(3) Checking the target IP is available in the DHCP lease table: If the target IP address appears in the DHCP lease table, the operation field of the incoming ARP packet will determine its ARP type. Otherwise, the process (4) will proceed. If the ARP type indicates an ARP request, the script retrieves the corresponding MAC address from the DHCP lease table to construct and send an ARP reply packet back to the sender host using the Traffic-Generator tool. However, if the ARP type is an ARP reply, the script interprets the incoming ARP packet as a gratuitous ARP packet and then proceeds to send it through the gateway interface, like the previous process.

(4) Checking that the target IP is identical to the local IP: If the target IP address matches the local IP address on the gateway interface, the script takes no action because the gateway's ARP itself is responsible for sending an ARP reply packet containing its gateway interface's local IP and MAC addresses. However, if the target IP address differs from the local IP address, a log message indicating that the destination host is not in the DHCP lease table can inform a network administrator, suggesting that the destination host does not exist in the network. The script then resumes reading the next incoming ARP packet from the packet sniffer tool and performs all processes again.

7. EXPERIMENT AND EVALUATION

Fig. 6 shows a network with a star topology to evaluate the proposed approach. It consists of five devices: one MikroTik router serving as the network's gateway, one MikroTik Ethernet switch, and three hosts, including two legitimate hosts (PC1 and PC2) and one malicious host acting as an attacker. All hosts, namely PC1, PC2, and the attacker, reside within the same VLAN, indicating they are part of the same subnet. The network address of the VLAN, in CIDR notation, is 192.168.10.0/24. CIDR stands for Classless Inter-Domain Routing. The hardware models of the Ethernet switch and router are CRS328-24P-4S+RM and hAP ac^2, respectively, with detailed specifications provided in Table 1 [39-40]. In Table 1, the more powerful router (CCR1009-7G-1C-1S+) will be used instead of the hAP ac^2 router for performance comparison. PC1 and PC2 operate on the Ubuntu operating system (version 22.04.4), while the attacker utilizes the Kali Linux operating system (version 2024.1). All hosts, including PC1, PC2, and the attacker, run as virtual machines using the VirtualBox hypervisor.

The router and Ethernet switch are configured according to the settings in Table 2. With the proposed approach, the Ethernet switch and router must be configured as follows. The Ethernet switch is configured to redirect all incoming ARP traffic from all connected hosts (port 1, port 2, and port 3) to the

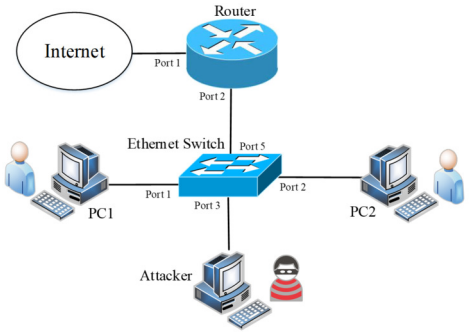

Fig.6: *The Network Structure for Experiments.*

uplink port (port 5) using the RouterOS command in Table 3. Additionally, the router is configured to dynamically add a pair of IP and MAC addresses into the router's ARP table immediately after the DHCP server has assigned the IP address. The ARP property on the VLAN interface in the router is set to reply-only, ensuring that it replies to an ARP request packet only if the IP and MAC addresses of the sender in the ARP request packet match an ARP entry in the ARP table. As shown in Table 4, the DHCP server is configured to assign the following IP addresses: 192.168.10.1 for the gateway, 192.168.10.10 for PC1, 192.168.10.20 for PC2, and 192.168.20.30 for the attacker. These IP addresses are reserved and consistently assigned for experimental and evaluation purposes. It is important to note that the RouterOS version of the MikroTik router and Ethernet switch is 7.14.1.

Table 1: *Hardware Specification of MikroTik Devices.*

Devices	Hardware Specification
hAP ac^2 (Router)	Architecture: ARM 32bit, CPU core count: 4 CPU nominal frequency: 716 MHz Size of RAM: 128 MB, Storage size: 16 MB Maximum Ethernet Speed: 1 Gbps
CCR1009-7G-1C-1S+ (Router)	Architecture: TILE, CPU core count: 9 CPU nominal frequency: 1.2 GHz Size of RAM: 2 GB, Storage size: 128 MB Maximum Ethernet Speed: 1 Gbps
CRS328-24P-4S+RM (Ethernet Switch)	Architecture: ARM 32bit, CPU core count: 1 CPU nominal frequency: 800 MHz Size of RAM: 512 MB, Storage size: 16 MB Maximum Ethernet Speed: 1 Gbps

Table 2: *Configuration for Router and Ethernet Switch.*

Devices	Configuration
Router	Port1: DHCP Client Port1: Source NAT (Network Address Translation) VLAN ID: 10 VLAN Interface: Port2 & DHCP Server VLAN Network: 192.168.10.0/24 VLAN Interface's IP Address: 192.168.10.1
Switch	Bridge Interface: Port1, Port2, Port3, Port5 Bridge Interface: VLAN Filtering (Enabled) VLAN ID: 10 Trunk Port: Port5 Access Port: Port1, Port2, Port3

Table 3: *RouterOS Commands for the GCA Approach.*

Devices	RouterOS Commands
Router	/interface vlan add $arp = reply - only$ interface=bridge1 name=vlan10 vlan-id=10 /ip dhcp-server add $add - arp = yes$ address-pool=dhcp-pool interface=vlan10 lease-time=10m name=dhcp1
Switch	/interface ethernet switch rule add mac-protocol=arp $new - dst - ports = ether5$ ports=ether1, ether2, ether3 switch=switch1

Table 4: *IP and MAC Addresses of All Devices.*

Device	IP Address	MAC Address
Gateway	192.168.10.1	48:A9:8A:9C:BC:2A
PC1	192.168.10.10	08:00:27:76:01:75
PC2	192.168.10.20	08:00:27:AB:F9:14
Attacker	192.168.10.30	08:00:27:1E:36:4A

With the preceding configuration, all traffic except ARP traffic from all hosts can directly communicate with each other via the Ethernet switch. For ARP traffic, all hosts can communicate only with the router. As mentioned in subsection 2.2, ARP cache poisoning attacks can occur via spoofed ARP request and reply packets. In other words, an attacker will spoof the SPA and SHA fields in either ARP request packets or reply packets to create an incorrect mapping of IP and MAC addresses in an ARP cache. To evaluate whether the proposed approach can prevent ARP cache poisoning attacks on all legitimate hosts and the router in the network, the attacker consecutively generates spoofed ARP request and reply packets using the packEth tool [15] [41] in both the network implementing the proposed approach and the traditional network (i.e., the network without the proposed approach) to poison the targets, including the ARP caches of PC1, PC2, and the gateway. To poison the ARP caches, the attacker sets up the SHA field of ARP packets with the attacker's MAC address but the IP address of either PC1, PC2 or the gateway in the SPA field. The Wireshark program is used at PC1, PC2, and attacker to validate the incoming ARP packets, while the packet sniffer tool at the router's gateway captures all incoming ARP packets. In Table 5, "Poisoned" indicates that the ARP cache has been poisoned, whereas "Unaffected" means the ARP cache remains unaffected. These results demonstrate that the proposed approach completely prevents ARP cache poisoning attacks from both spoofed ARP request packets and reply packets. This effectiveness is achieved by redirecting all ARP traffic at the Ethernet switch from the hosts to the router, which handles legitimate ARP traffic based on its DHCP lease table. Consequently, the attacker cannot communicate directly with the hosts for ARP traffic.

In addition to preventing ARP cache poisoning attacks, the GCA script also measures and averages the processing times for both spoofed and legitimate ARP packets. These measurements are based on 100

Table 5: *ARP Cache After ARP Cache Poisoning Attack.*

Device's ARP Cache	Traditional Network		Proposed Approach	
	Spoofed ARP Request	Spoofed ARP Reply	Spoofed ARP Request	Spoofed ARP Reply
PC1	Poisoned	Poisoned	Unaffected	Unaffected
PC2	Poisoned	Poisoned	Unaffected	Unaffected
Gateway	Poisoned	Poisoned	Unaffected	Unaffected

Table 6: *Average Processing Times of the GCA Script.*

ARP Packets	Results	Processing Times (milliseconds)	
		hAP ac^2	CCR1009-7G-1C-1S+
Legitimate ARP Request	Response with ARP Reply	239.237401	199.608573
Spoofed ARP Request	Attack Detected	190.533751	147.448218
Spoofed ARP Reply	Attack Detected	183.449765	145.329931

consecutive packets generated using the packEth tool for each ARP request and reply. Three measurement scenarios are conducted with different types of ARP traffic: (1) legitimate ARP request, (2) spoofed ARP request, and (3) spoofed ARP reply. It is important to note that legitimate ARP request packets originate from PC1 to inquire about the MAC address of PC2. In contrast, spoofed ARP request and reply packets are transmitted from the attacker to poison the ARP cache of PC1, using PC2's IP address for the SPA field and the attacker's MAC address for the SHA field in the ARP packets.

The processing time for a spoofed ARP packet, also known as ARP spoofing detection time, is the duration between when the script detects an incoming ARP packet and when it identifies the packet as spoofed. Conversely, for legitimate ARP packets, the processing time is between when the script reads an incoming ARP request and when it is complete, sending the corresponding ARP reply. The script utilizes the timestamp command to acquire these time instants. In addition, the processing times are also measured using the higher computing router (i.e., CCR1009-7G-1C-1S+) with its detailed specification, as shown in Table 1. In Table 6, the average processing times in the router with less computing power (hAP ac^2) for legitimate and spoofed ARP request/reply packets are higher compared to the more powerful router (CCR1009-7G-1C-1S+), approximately 40 ms to 60 ms.

Furthermore, the performance of increased CPU load is considered when executing the GCA script. The previous experimental setup for average processing times is used for this evaluation. The increased CPU loads under various numbers of legitimate ARP request packets per second (PPS), i.e., 0, 500, 1,000, 1,500, 2,000, 2,500, 3,000, 3,500, and 4,000, are determined by subtracting the router's average CPU loads

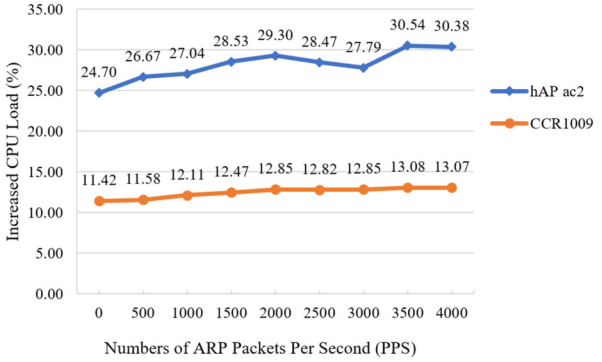

Fig. 7: *The increased CPU Loads for the GCA Approach.*

Table 7: *Response Times of Legitimate ARP Request.*

Routers	Target Hosts	Response Times (milliseconds)	
		Traditional	GCA Approach
hAP ac^2	PC1	3.345	248.415
	PC2	2.018	244.490
	Gateway	1.987	1.896
CCR1009-7G-1C-1S+	PC1	3.387	217.363
	PC2	2.060	210.301
	Gateway	1.782	1.850

in the network with the GCA approach from those in the traditional network. The CPU load of each router in the network with the GCA approach and the traditional network is continuously recorded every second within 100 seconds for legitimate ARP request packets from PC1 to inquire about PC2's MAC address. The packEth tool is employed to generate various numbers of legitimate ARP request packets per second. In Fig. 7, the CPU loads in the networks using either of the two routers (hAP ac^2 and CCR1009-7G-1C-1S+) with the proposed approach are greater than those of the traditional network at legitimate ARP requests of 0 to 4000 PPS by 24.70% to 30.54% and 11.42% to 13.08%, respectively.

Also, the response times of legitimate ARP packets in the network using the proposed approach are evaluated compared to those of the traditional network. The response times of ARP request packets in each network are averaged from the measured round-trip times (RTT) of 100 ARP packets. To set up this measurement, the attacker utilizes the arping tool [27] to measure the round-trip times by transmitting legitimate ARP request packets to the following targets: PC1, PC2, the router's gateway, and then waits for the corresponding ARP reply packets to be returned. In most Linux distributions as well as Windows operating systems, the default timeout for ARP requests is typically set to 1 second. According to Table 7, the response times of ARP requests in the network with the GCA approach are within 210 - 250 ms. It means that the GCA approach, in the worst-case scenario, can handle a maximum of 4 ARP request packets arriving at the gateway simultaneously, as the waiting

time for the last ARP reply packet must be within the default timeout for ARP requests. However, the response times for the gateway, acting as the target host, from the network with the GCA approach and the traditional network are not significantly different and fall within two milliseconds. This similarity arises because the GCA approach does not undertake the responsibility of responding to ARP request packets that inquire about the gateway's MAC address.

8. CONCLUSION AND FUTURE WORK

This paper presents the proposed GCA approach to effectively prevent ARP cache poisoning attacks in RoaS networks using MikroTik networking devices. The main advantage of the proposed approach is that it is not a host-based solution. In other words, it does not require any modification or additional mechanisms on end devices or hosts. Another significant contribution of this paper is that the proposed approach can be applied not only to MikroTik networking devices but also to other commercial networking devices lacking the DAI feature; that is, Ethernet switches must be configured to redirect ARP packets from any host to the router gateway. In addition, a router needs to include a command script based on the proposed approach to monitor, analyze, and transmit ARP packets. According to the experimental results, the proposed approach can prevent all ARP cache poisoning attacks from spoofed ARP request/reply packets. The performance of the proposed approach has been evaluated in terms of processing time, increased CPU load, and response time compared with the traditional network. The increased CPU load is not extremely high, especially with the CCR1009-7G-1C-1S+ router (13.08%). However, the processing time of the GCA script is somewhat considerable (199 - 240 ms.). It leads to relatively high response times (210 - 250 ms) compared to traditional networks (2 - 3.5 ms). The downside of this approach is that it introduces a single point of failure, as it relies on the router in a RoaS network. If the router fails, ARP on all hosts will be unable to communicate with it. In future work, the proposed approach will be implemented in the network protocol stack of operating systems and hardware, such as an ASIC chip, to improve performance, particularly regarding response time. In addition, the proposed approach will be developed further for a RoaS network configured with VRRP (Virtual Router Redundancy Protocol) to provide redundancy in case the primary router fails.

AUTHOR CONTRIBUTIONS

Ekarin Suethanuwong is the sole author who contributed to all aspects of the work, including conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing—original draft preparation, writing—review and editing, visualization, and supervision. The author has read and agreed to the published version of the manuscript.

References

[1] A. Petrosyan, "Worldwide digital population 2024," Statista, Technical Report, Jan. 31, 2024, Accessed: Mar. 18, 2024, [Online.] Available: `https://www.statista.com/statistics/617136/digital-population-worldwide`

[2] B. Alotaibi and H. Almagwashi, "A Review of BYOD Security Challenges, Solutions and Policy Best Practices," *Proceedings of the 2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*, pp. 1-6, Saudi Arabia, 2018.

[3] G. A. Safdar and A. Mansour, "Security and Trust Issues in BYOD Networks," *IT Professional*, vol. 25, no. 4, pp. 45-51, July-Aug. 2023.

[4] M. Sharma and S. Ravichandra, "Design and implementation of a mechanism to identify and defend against ARP spoofing," *Proceedings of the 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pp. 1-6, Delhi, India, 2023.

[5] C. G. Weissman, "Here's why Equifax yanked its apps from Apple and Google last week," Sep. 2017.

[6] 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.

[7] M. Nobakht, H. Mahmoudi and O. Rahimzadeh, "A Distributed Security Approach against ARP Cache Poisoning Attack," *Proceedings of the 1st Workshop on Cybersecurity and Social Sciences (CySSS)*, pp. 27-32, Japan, May 30, 2022.

[8] S. Hijazi and M. S. Obaidat, "A New Detection and Prevention System for ARP Attacks Using Static Entry," *IEEE Systems Journal*, vol. 13, no. 3, pp. 2732-2738, Sep. 2019.

[9] J. S. Meghana, T. Subashri and K. R. Vimal, "A survey on ARP cache poisoning and techniques for detection and mitigation," *Proceedings of the 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN)*, pp. 1-6, Chennai, India, 2017.

[10] H. A. S. Adjei, M. T. Shunhua, G. K. Agordzo, Y. Li, G. Peprah and E. S. A. Gyarteng, "SSL Stripping Technique (DHCP Snooping and ARP Spoofing Inspection)," *Proceedings of the 2021 23rd International Conference on Advanced Communication Technology (ICACT)*, pp. 187-193, PyeongChang, Korea (South), 2021.

[11] W. J. Tay, S. L. Lew and S. Y. Ooi, "Remote Access VPN using MikroTik Router," *Proceedings of the 2022 International Conference on Com-*

puter and Drone Applications (IConDA), pp. 119-124, Kuching, Malaysia, 2022.

[12] J. M. Ceron, C. Scholten, A. Pras and J. Santanna, "MikroTik Devices Landscape, Realistic Honeypots, and Automated Attack Classification," *Proceedings of the NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1-9, Budapest, Hungary, 2020.

[13] D. C. Puchianu, N. Angelescu, G. Predusca, D. Circiumarescu and E. Diaconu, "Comparative study of power consumption on MikroTik and FortiGate routers," *Proceedings of the 2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1-4, Bucharest, Romania, 2020.

[14] S. M. Morsy and D. Nashat, "D-ARP: An Efficient Scheme to Detect and Prevent ARP Spoofing," *IEEE Access*, vol. 10, pp. 49142-49153, 2022.

[15] S. Singh and D. Singh, "ARP Poisoning Detection and Prevention Mechanism using Voting and ICMP Packets," *Indian Journal of Science and Technology*, vol. 11, no. 22, pp. 1-9, June 2018.

[16] G. Jinhua and X. Kejian, "ARP spoofing detection algorithm using ICMP protocol," *Proceedings of the 2013 International Conference on Computer Communication and Informatics*, pp. 1-6, Coimbatore, India, 2013.

[17] G. Kaur and J. Malhotra, "Comparative Investigation of ARP Poisoning Mitigation Techniques Using Standard Testbed for Wireless Networks," *Journal of Cyber Security and Mobility*, vol. 4, no. 2-3, pp. 53-64, Nov. 2015.

[18] J. Singh, S. Dhariwal and R. Kumar, "A Detailed Survey of ARP Poisoning Detection and Mitigation Techniques," *International Journal of Control Theory and Applications (IJCTA)*, vol. 9, no. 41, pp. 131-137, 2016.

[19] R. K. Bijral and A. Gupta, "Study of Vulnerabilities of ARP Spoofing and its detection using SNORT," *International Journal of Advanced Research in Computer Science*, vol.8, no.5, May – June 2017.

[20] D. Bruschi, A. Ornaghi and E. Rosti, "S-ARP: a secure address resolution protocol," *Proceedings of the 19th Annual Computer Security Applications Conference*, pp. 66-74, Las Vegas, NV, USA, 2003.

[21] O. S. Younes, "Securing ARP and DHCP for mitigating link layer attacks," *Sadhana*, vol. 42, no. 12, pp. 2041–2053, Dec. 2017.

[22] V. Prevelakis and W. Adi, "LS-ARP: A lightweight and secure ARP," *Proceedings of the 2017 Seventh International Conference on Emerging Security Technologies (EST)*, pp. 204-208, Canterbury, UK, 2017.

[23] R. Kaur, E. G. Singh and S. Khurana, "A security approach to prevent ARP poisoning and defensive tools," *International Journal of Computer and Communication System Engineering (IJCCSE)*, vol. 2, no. 3, pp. 431-437, Jun. 2015.

[24] D. Majumdar and R. R. Chintala, "Shadow APR-ing for APR Poisoning Detection," *Proceedings of the 9th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 128-131, Coimbatore, India, 2023.

[25] A. M. Abdel Salam, W. S. Elkilani and K. M. Amin, "An automated approach for preventing ARP spoofing attack using static ARP entries," *International Journal of Advance Computer Science and Applications (IJACSA)*, vol. 5, no. 1, 2014.

[26] M. Data, "The Defense Against ARP Spoofing Attack Using Semi-Static ARP Cache Table," *Proceedings of the 2018 International Conference on Sustainable Information Engineering and Technology (SIET)*, pp. 206-210, Malang, Indonesia, 2018.

[27] L. Allen, T. Heriyanto and S. Ali, *Kali Linux – Assuring Security by Penetration Testing*, Packt Publishing, 2014.

[28] N. Tripathi and B. M. Mehtre, "An ICMP-based secondary cache approach for the detection and prevention of ARP poisoning," *Proceedings of the 2013 IEEE International Conference on Computational Intelligence and Computing Research*, pp. 1-6, Enathi, India, 2013.

[29] S. Kumar and S. Tapaswi, "A centralized detection and prevention technique against ARP poisoning," *Proceedings of the 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec)*, pp. 259-264, Malaysia, 2012.

[30] D. R. Rupal, D. Satasiya, H. Kumar and A. Agrawal, "Detection and prevention of ARP poisoning in dynamic IP configuration," *Proceedings of the 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pp. 1240-1244, Bangalore, India, 2016.

[31] S. Y. Nam, D. Kim and J. Kim, "Enhanced ARP: preventing ARP poisoning-based man-in-the-middle attacks," *IEEE Communications Letters*, vol. 14, no. 2, pp. 187-189, February 2010.

[32] Y. Zhao, R. Guo and P. Lv, "ARP Spoofing Analysis and Prevention," *Proceedings of the 2020 5th International Conference on Smart Grid and Electrical Automation (ICSGEA)*, pp. 572-575, China, 2020.

[33] P. Arote and K. V. Arya, "Detection and Prevention against ARP Poisoning Attack Using Modified ICMP and Voting," *Proceedings of the 2015*

International Conference on Computational Intelligence and Networks, pp. 136-141, Odisha, India, 2015.

[34] T. Alharbi, D. Durando, F. Pakzad and M. Portmann, "Securing ARP in Software Defined Networks," *Proceedings of the 2016 IEEE 41st Conference on Local Computer Networks (LCN)*, pp. 523-526, Dubai, United Arab Emirates, 2016.

[35] H. Y. Ibrahim, P. M. Ismael, A. A. Albabawat and A. B. Al-Khalil, "A Secure Mechanism to Prevent ARP Spoofing and ARP Broadcasting in SDN," *2020 Proceedings of the International Conference on Computer Science and Software Engineering (CSASE)*, pp. 13-19, Duhok, Iraq, 2020.

[36] H. Cho, S. Kang and Y. Lee, "Centralized ARP proxy server over SDN controller to cut down ARP broadcast in large-scale data center networks," *Proceedings of the 2015 International Conference on Information Networking (ICOIN)*, pp. 301-306, Cambodia, 2015.

[37] M. Ren, Y. Tian, S. Kong, D. Zhou and D. Li, "A detection algorithm for ARP man-in-the-middle attack based on data packet forwarding behavior characteristics," *Proceedings of the 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 1599-1604, Chongqing, China, 2020.

[38] "Bridging and switching," RouterOS – MikroTik Documentation, Accessed: Mar. 18, 2024. [Online]. Available: `https://help.mikrotik.com/docs/display/ROS/BridgingandSwitching`.

[39] "MikroTik hAP ac^2 - Dual-concurrent Access Point," Accessed: Mar. 18, 2024. [Online]. Available: `https://mikrotik.com/product/hap\_ac2`

[40] "MikroTik CCR1009-7G-1C-1S+ Cloud Core Router," Accessed: Mar. 18, 2024. [Online]. Available: `https://mikrotik.com/product/CCR1009-7G-1C-1Splus`

[41] S. Srivastava, S. Anmulwar, A. M. Sapkal, T. Batra, A. K. Gupta and V. Kumar, "Comparative study of various traffic generator tools," *Proceedings of the 2014 Recent Advances in Engineering and Computational Sciences*, pp. 1-6, India, 2014.

Ekarin Suethanuwong is an assistant professor of information technology at Prince of Songkla University, Trang campus, Thailand. He received his B.Eng. in Electrical Engineering (with a major in Communication) with honors from Prince of Songkla University in 1999, his M.Eng. in Computer Engineering from Prince of Songkla University in 2005, and his Ph.D. in Computer Science from Vienna University of Technology, Austria, in 2012, with Prof. Hermann Kopetz as his research advisor. In 2018, he became a MikroTik certified trainer and has since provided several MikroTik training courses to the public. To date, he has obtained several well-known industry certifications in computer networking, security, and cloud computing, including MikroTik (MTCNA, MTCRE, MTCTCE, MTCSE, MTCSWE, MTCUME), Cisco (CCNA), Huawei (HCIA-Datacom and HCIA-Security), AWS solution architect. His research interests are real-time data communication, time-triggered Ethernet, advanced Ethernet technology, wired and wireless networking, Huawei and MikroTik networking, cyber security, ethical hacking, and cloud computing.