BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **17**, No 4
Sofia • 2017 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2017-0042

# Security Solution for ARP Cache Poisoning Attacks in Large Data Centre Networks

*B. Prabadevi, N. Jeyanthi*

*School of Information Technology and Engineering VIT University, Vellore, India*
*E-mails: prabadevi.b@vit.ac.in njeyanthi@vit.ac.in*

**Abstract:** *The bridge protocol* (*Address Resolution Protocol*) *ARP, integrating Ethernet* (*Layer 2*) *and IP protocol* (*Layer 3*) *plays a vital role in TCP/IP communication since ARP packet is the first packet generated during any TCP/IP communications and they are the first traffic from the host. In the large data center, as the size of the broadcast domain* (*i.e., number of hosts on the network*) *increases consequently the broadcast traffic from the communication protocols like ARP also increases. This paper addresses the problem faced by Layer 2 protocols like in-secured communication, scalability issues and VM migration issues. The proposed system addresses these issues by introducing two new types of messaging with traditional ARP and also combat the ARP Cache poisoning attacks like host impersonation, MITM, Distributed DoS by making ARP stateful. The components of the proposed methodology first start the process by decoding the packets, updates the invalid entry made by the user with Timestamp feature and messages being introduced. The system has been implemented and compared with various existing solutions.*

**Keywords:** *Large data center networks, Broadcast storms, VM migration, Timestamp, ARP cache poisoning attacks.*

## 1. Introduction

Today’s Enterprises are collecting and processing TeraBytes (TB) of secured data on air and are utilizing technologies like a cloud to host their information. These bulky data storage fascinates most of the attackers which makes data security in this aspect a greater challenge. Big Data analytics processes large scale of unstructured and structured data. Various definitions for Big Data exist in the literature. Of those, the tremendous growth of data has been well described in 3Vs [1] – Volume, Velocity, and Variety. Furthermore Vs viz., Veracity, Variability, and Value have been added to its growing characteristics and challenging nature.

Though there is not a single profound architecture for Big data, researchers suggest various architecture layouts like 5-layered Big data Systems Architecture [2], namely Data source layer, Integration layer, Data storage layer, Analytics –

computing layer and Presentation layer. The components of these layers clearly depict the heterogenic and tremendous nature of Big Data systems.

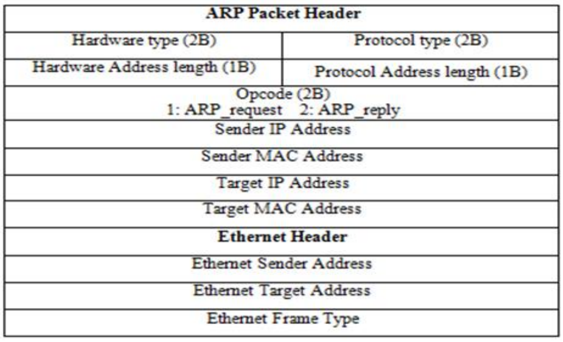
Big data security is an imperative stride for academicians and Researchers. In [3], the authors proposed a detection technique for DDoS attacks in the cloud, which also protects the Data center from unwanted traffic. Thandeeswaran et al. [4], proposed a secure cloud infrastructure (virtual) by using AES for enhanced security.

In large data centers (using either IPv4 or IPv6) with numerous hosts, the problem in Address resolution is faced. The address resolution protocol (ARP-RFC0826) is used for resolving the IP address to MAC address in IPv4 and Neighbour Discovery (ND-RFC4861) is used in IPv6 for address resolution within the large data center. In addition to issues the ARP protocol has in a normal situation, it has several issues with address resolution in the large data center as specified in RFC 6820 [5]. In this paper, we provide a solution for the issues faced by ARP in large datacentres viz., broadcast storms which are caused when the number of hosts in the data centers increases. This, in turn, increases the broadcast traffic devastating the entire network since all the devices in the broadcasting domain must act on this traffic.

## 2. An overview of ARP in large datacenters

ARP is the telecommunication protocol used for address resolution in LAN. Most noteworthy features of ARP are stateless and non-authenticated, which is the reason for most of the attacks caused by ARP. For any device to communicate with any other devices in LAN it must require Link layer address (Layer 2) called MAC address. If there is a device A willing to share its data with another device B, but it has only the IP address of device B, B can still communicate with A with the help of ARP. ARP maintains a cache in all devices containing the following details: ARP CACHE (IP-MAC PAIR, Network Interfaces, and ARP TYPE). So first A will look up its ARP cache for B’s MAC address, if found it will fetch the detail and fills the destination column in ARP header with B’s MAC address and sends the data otherwise device A sends a broadcast ARP-REQUEST message to all devices in the network and in turn a device with matching IP address (i.e., B) will update its MAC address and generate a unicast ARP-REPLY message for sender, i.e., A here. ARP packet format and header is depicted in Fig. 1.

The in-secured nature of ARP makes it vulnerable to ARP cache poisoning/ spoofing attack by the attacker can make an invalid entry in ARP cache and redirect the incoming and outgoing traffic to attacker’s system. Some of the advance attacks caused by ARP spoofing are Sniffing, Denial of Service, IP spoofing, Man-in-the-middle attack, host impersonation and so on [6]. Jeyanthi and Kumar [7] proposed a virtual firewall using Army node for avoiding DDoS attacks in the cloud which increase the Data center availability during the DDoS attack. The authors proposed a Recurrence Quantification Approach [8], which detects a DDoS attack in VoIP network at the beginning stage thereby preventing it further.



| ARP Packet Header | |
|---|---|
| Hardware type (2B) | Protocol type (2B) |
| Hardware Address length (1B) | Protocol Address length (1B) |
| Opcode (2B)<br>1: ARP_request 2: ARP_reply | |
| Sender IP Address | |
| Sender MAC Address | |
| Target IP Address | |
| Target MAC Address | |
| **Ethernet Header** | |
| Ethernet Sender Address | |
| Ethernet Target Address | |
| Ethernet Frame Type | |

Fig. 1. ARP packet header on Ethernet /IP

### 2.1. ARP in large datacentres

The technology like big data hosts multiple heterogeneous data and in a different format, so it should be dynamic and faster enough to retrieve the data. In such kind of data centers when the number of hosts increases, the traffic from link layer protocols like ARP will increase. In most of the situations like load balancing in the cloud where the dynamic relocation of servers will take place, the ARP should be able to update its cache when the hosts migrate [5]. Henceforth when the number of hosts keeps increasing and changing, a rapid broadcast will be more challenging which includes issues like:

- Are the ARP caches of the hosts recently updated? If updated, is the intended host alive in the network?
- Does any ARP broadcast request message will consume larger bandwidth?
- If not, then how long it will take for a single ARP communication?

This in the worst case leads to ARP broadcast storms when the broadcast traffic level crosses the tolerance capacity of the DCN (Data Center Networks). To avoid this most of the large DCNs, split the large one into small networks each working as its own Layer 3 subnet maintaining its own broadcast traffic using Top-of-the Rack (ToR) switch at each rack [5]. ToR switches will manage the devices on its rack and connect to routers (which are placed at the top of the racks). But still, it will be difficult in dynamic load balancing, where ARP table has to keep track of all changing IP addresses.

### 2.2. Broadcast storm

The broadcast storm is depicted in Fig. 2. Consider a network with four segments containing 4 multiway switches. Assume 5 PCS, of which PC1, wishes to communicate with PC3 but it does not know the MAC addresses of PC3. So the following steps take place:

1. PC1 generate an ARP broadcast message asking for MAC address of PC3.
2. Switches SW1, SW2, SW3 and SW4 receive the frame and flood it out.
3. Since it is cyclic loop, the frame runs in the loop exploiting all the resources.
4. At a given stage the switches will not respond.
5. All the traffic stops because switches keep transmitting the frames.

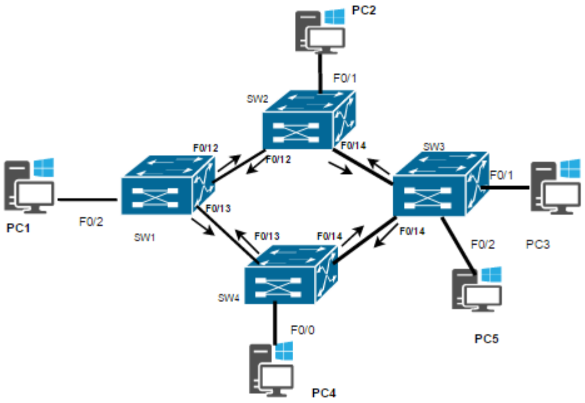


Fig. 2. ARP broadcast storm

This activity not only leads to broadcast storms but can even result in Denial of service to legitimate users in the network. The broadcast storm can occur due to a variety of reasons as specified in [9]. Though various methods for preventing the Broadcast message are available, they won't completely eradicate the storm instead it will reduce the risk level.

The notable feature of DCN is that it must be capable of enabling any server at any time to any service (most of the cases DCN uses flat addressing instead hierarchical addressing) [10]. In large DCNs the number of devices connected to the network will be dynamically scaled and migrate for load balancing, but still, the VM must have the same IP address. In such cases, this storm will exploit all the resources in the network and diminish the system performance.

Anurag Khandelwal, Jain and Kamara [18] Have proposed a source based filtering scheme which uses a directory system and restricts the unwanted traffic at the source. They use a directory which maintains the details of trusted VMs which maps the names to locators. The destination VM is allowed to communicate only if it is in directory otherwise packets are dropped. Huan Ma et al. [19] proposed a Bayes based ARP detection using SDN. SDN controller is used for constructing a global ARP cache, which also manipulates VMs ARP cache with custom ARP requests. They proved that it is better than the existing cryptographic solutions. But still, it is based on the centralized server which may lead to the central point of failure.

The various solutions of this ARP cache poisoning attacks are: Ticket based ARP [20], static ARP [21], stateful ARP, Cryptographic techniques like SARP [22], EARP [23] DES-based approach [24], centralized approaches like Active Server [25], Pandey's Probe-based technique [26] and various intrusion detection open-source tools available in the market.

## 3. Proposed solution

Most of the DCN prefer ToR switch which performs better in scaling situation. When the broadcast domain of the host is rapidly increasing with a large number of hosts, Layer 2 network is divided into multiple smaller L2 networks each rack is coordinated by a ToR switch functioning at Layer 3 subnet of IP [5]. This is why traditional ARP to reduce the overhead caused by broadcast traffic, IP subnet size has been limited to few hundred machines [11]. Most of the DCN disable the ARP because of overhead caused: VLB in [10], load spreading in [11], Pseudo MAC address and Fabric Manager in [12], 1-hop DHT in SEATTLE [13] and Switches map MAC addresses to remote switch [14]. Portland [12] is a fault-tolerant protocol supporting plug and play feature for DCNs. During the migration of VMs, hypervisors use Reverse-ARP, which provides IP address for given MAC address [15]. The architecture of the proposed system is depicted in Fig. 3.

The proposed system makes the following assumption: In large DCNs, to avoid overhead caused by ARP like protocols, when the number of hosts increases, it partitions the large physical network into a number of smaller networks (VLANs with a fewer number of devices) with ToR switch at each rack [11].

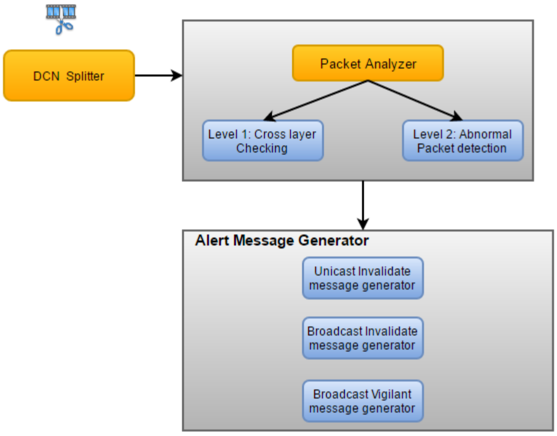

DCN Splitter
Packet Analyzer
Level 1: Cross layer Checking
Level 2: Abnormal Packet detection
Alert Message Generator
Unicast Invalidate message generator
Broadcast Invalidate message generator
Broadcast Vigilant message generator

Fig. 3. Architecture of the proposed system

When a VM migrates, consider its configuration file with extension .cfg or .vmx is also moved, i.e., its IP-MAC addresses will be changed and it is reachable. Layer 2 domains are restricted to have only 4000 servers so ARP limits IP subnet size to less number of servers [11].

The flowchart for the proposed system is specified in Fig. 7. On reception of request or response, a cross-layer check is performed to ensure authenticity. It has the following components:

### 3.1. Splitting the large DCNs with more number of smaller Layer 2 networks

To avoid the broadcast storms through loops, the switches can be configured to avoid them. In addition, the single larger physical network can be partitioned into many smaller Layer 2 networks coordinated by ToR switch at Layer 3 IP subnet. The DCN architecture composed of three-layered switches core layer, access layer, and aggregation layer [16]. The aggregation layer has multi-layered Layer 2 switches. One ToR per rack switch will be acting as a gateway for each subnet.
This feature will avoid loops; as well this will not create storms. In order to avoid the ARP request storms, on reception of an ARP request the scheme will behave

differently with the help of the new entries in the cache. The ARP cache will have two new entries viz., count of ARP requests, i.e., Creq received on same entry and timestamp when the entry was made.

Table 1. Modified ARP cache

| Type | IP address | MAC address | Interface | $TS_{up}$ | Creq |
|---|---|---|---|---|---|
| Static/ dynamic | 172.168.0.1 | 00:5:79:66:68:01 | Fast Ethernet 0/1 | 2016-07-14 04:32:26 | 3 |

Table 2. Invalid List Table

| Index | IP address | MAC address | Count | $TS_{lup}$ |
|---|---|---|---|---|
| 1 | 198.164.0.3 | 0034.4456.2234 | 1 | 2016-08-14 03:22:26 |
| 2 | 165.178.0.5 | 0098.9876.3456 | 10 | 2016-08-14 02:12:56 |

$TS_{lup}$ → last updated timestamp

```
        1. Generating ARP Requests
        Assumption:
        1. Consider a Host A wants to communicate with Host B in its
subnet, but it does not have an entry in ARP cache.
        2. A and B are on the same subnet
        3.ARP table tuple: ARP<Type, IP address, MAC Address, Interface,
$TS_{up}$, $C_{req}$>
        4. A knows the IP of B but not it's MAC
        Processing:
        If($IP_B$ AND $MAC_B$==Unknown) Then
        GEN: $ARP_{REQ}(OP_{REQ}, IP_A, MAC_A, IP_B, BDC_{MAC}, TS_{qg}, MSG)$
        SET:
        $OP_{REQ} \leftarrow 1, IP_{TAR} \leftarrow IP_B, MAC_{TAR} \leftarrow BDC_{MAC}, IP_{SRC} \leftarrow IP_A,$
$MAC_{SRC} \leftarrow MAC_A$ in $ARP_{REQ}$
        IP=$IP_B$ and MAC=NULL in $ARP_A$
        Else // $IP_B$-$MAC_B$ is in $ARP_A$
        $SEND_{PKT}$
```

Count → No of times request for this entry was made

```
                         Processing ARP requests
  2. Processing ARP Request
Assumption:
1. Host B has received the $ARP_{REQ}$ from host A.
2.$OP_{REQ}$=1
Processing:
If ($MAC_{TAR}$ == $BDC_{MAC}$) Then
    If($ARP_{MAC}$==$ETH_{MAC}$) Then
      If($IP_A$ and $MAC_A$  not  in $INV_{ListB}$) Then
         If($IP_A$-$MAC_A$ is in $ARP_B$) Then
           If(CLK- $ARP_{req}.TS_{qg}$ < 10 s)Then
             If($ARP_B.C_{req}$<4)Then
               GEN: $ARP_{REP}$
```

```
              $ARP_B.C_{req} \rightarrow ARP_B.C_{req}+1$
              Else Discard $ARP_{REQ}$
            Else $SEND_{ACK}$ to A
          Else
              ADD $IP_A$-$MAC_{A\,,}$ $TS_{up} \leftarrow TS_{qg}$ in $ARP_B$
              $C_{req}=1$
        Else
              ADD $IP_A$-$MAC_A$,$TS_{lup} \leftarrow TS_{qg}$ in $INV_{ListB}$
              Discard $ARP_{REQ}$
              GEN: $BDC_{VIG}$
      Else
              IF($INV_{LISTB}$.Count >3)
                SEND: $BDC_{VIG}$
              Else $INV_{LISTB}$.Count+=1
  Else ADD: $ARP_{REQ}$ in $INV_{LISTB}$
Else Discard $ARP_{REQ}$ and ADD $ARP_{REQ}$ in $INV_{LISTB}$
```

3.2. Packet analysis component

This component captures and decodes the packets. This can be done through network packet analyzers and capture tool like Wireshark [6] available. This component performs the *cross-layer checking, abnormal packet detection* based on Timestamp expiration. That is whenever an ARP Request or Reply message is generated it will be embedded with a Timestamp field, that indicates the validity of a lifetime packet [17].

**Processing ARP Replies**

**1. Generating ARP Reply**

**Assumption:**

1. $ARP_{REQ}$ from A is valid and it is successfully processed all the checks in processing ARP requests
2. $ARP_B$ is updated with host A details

**Processing:**

```
GEN: $ARP_{REP}$<$IP_B$,$MAC_B$,$IP_{A,}$$MAC_A$,$TS_{qg}$>
SET: $OP_{REP} \leftarrow 2$, $IP_{TAR} \leftarrow IP_A$, $IP_{SRC} \leftarrow IP_B$, $MAC_{TAR} \leftarrow MAC_A$,
$MAC_{SRC} \leftarrow MAC_B$, $TS_{qg} \leftarrow CLK$
```

**2. Processing ARP Reply**

**Assumption:**

1. Host A received $ARP_{REP}$ from B
2. $OP_{REP}=2$

**Processing:**

```
If ($MAC_{TAR}$== $UNI_{MAC}$ ) Then
  If ($ARP_{MAC}$==$ETH_{MAC}$) Then
    If ($IP_B$ and $MAC_B$ not in $INV_{ListA}$) Then
      If ($IP_B$ and $MAC_B \leftarrow$NULL in $ARP_A$) Then
        If (CLK- $ARP_{rep}.TS_{pg}$ < 10s) Then
```

```
            SET:
            $ARP_{A.}MAC_{B} \leftarrow ARP_{rep}.MAC_{SRC}$
            $ARP_{A}.TS_{up} \leftarrow TS_{pg}$
          Else
              Discard $ARP_{REP}$
             GEN: $UNI_{ALT}$ to A
         Else
            ADD $IP_{B}$-$MAC_{B}$, $TS_{lup} \leftarrow TS_{pg}$ in $INV_{ListA}$
            Discard $ARP_{REP}$
            GEN: $BDC_{VIG}$
        Else
            IF($INV_{LISTA}$.Count >3)
                SEND: $BDC_{VIG}$
            Else $INV_{LISTA}$.Count+=1
      Else ADD: $ARP_{REP}$ in $INV_{LISTA}$
Else Discard $ARP_{REP}$ and ADD $ARP_{REP}$ in $INV_{LISTB}$
```

The packets have to be processed before the time specified in Timestamp.

If ($OP_{REQ}$ == 1) then

ARP Request;

If ($OP_{REP}$==2) then

ARP Reply;

The message format of ARP request and reply messages are depicted in the Figs 8 and 9, respectively.

To enhance security, these packets can be modified by embedding a timestamp in all the messages [17]. The two new messages with opcode =26.28 are broadcast Alert message and opcode = 27 is unicast alert message has been introduced to enhance the security.

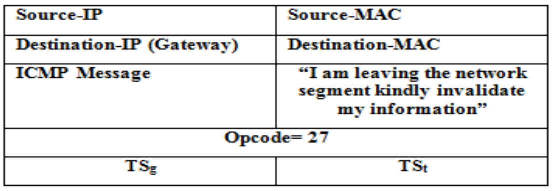
**Cross-Layer checking.** The Ethernet header has the following contents eth_header(Target_MAC, Source_MAC, Ethernet type, Ether_payload). Here the Ether_payload is going to be ARP_Header which can be a request or response which has the contents arp_header(H/w type, protocol_type, h/w size, protocol_field_size, opcode, data). The data field of arp_header contains the arp payload containing: arp_payload(Source_MAC,Source_IP, Target_MAC, Target_IP, Timestamp). The MAC in eth_header and ARP_ payload are cross-checked to avoid malicious user entry in ARP Cache.

**Timestamp expiration.** If the cross-layer checking is passed successfully, then the timestamps are evaluated as the difference of system clock time and timestamp time is found and if it is less than 10s, then the number of requests ($C_{req}$)from the same host in ARP table is greater than 4, the request is dropped.

The $C_{req}$ field is extracted from the ARP cache of the corresponding host.

### 3.3. Generate unicast invalidate message

This component will be functioning when VM migration takes place in DCN. Whenever a host /VM want to migrate to balance the load, it must generate a unicast alert message as depicted in the Fig. 4. This message is intended to the gateway telling that the VM is leaving the network segment and update this to all the other hosts in the network.



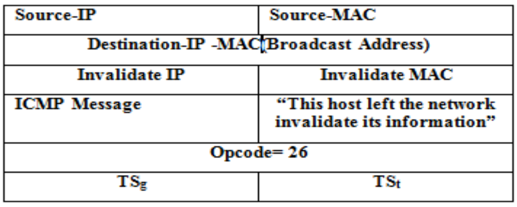
| | |
|---|---|
| **Source-IP** | **Source-MAC** |
| **Destination-IP (Gateway)** | **Destination-MAC** |
| **ICMP Message** | **"I am leaving the network segment kindly invalidate my information"** |
| **Opcode= 27** | |
| **$TS_g$** | **$TS_t$** |

Fig. 4. Unicast invalidate message

Once it has been migrated, it will generate Gratuitous ARP packet a special ARP request, to introduce itself in the new network segment. In turn, the gateway will invoke the next module.

The $TS_{qg}$, $TS_{qt}$, TSpg, and $TS_{pt}$ and $TS_t$ refer to Time Stamp of request generation time, request Expiration time, reply generation time and expiration time respectively. $TS_g$ and $TS_t$ refer to timestamp generation and expiration times of other messages.

The difference of both will give the total alive time of an ARP packet. The opcode is 4 since it is a special time of the unicast alert. The timestamps are used to trace out the fake ARP entries and avoid multiple ARP requests being generated. *On receiving this host can either* move this entry to invalid or clear the cache. Though the latter one is better for security reasons, it may incur a number of ARP messages to be generated. This message format is depicted in the Fig. 5. The opcode is 26 since it is a special type of broadcast alert.



| | |
|---|---|
| **Source-IP** | **Source-MAC** |
| **Destination-IP -MAC(Broadcast Address)** | |
| **Invalidate IP** | **Invalidate MAC** |
| **ICMP Message** | **"This host left the network invalidate its information"** |
| **Opcode= 26** | |
| **$TS_g$** | **$TS_t$** |

Fig. 5. Broadcast invalidate message

### 3.4. Generate Broadcast_Vigilant_message

This is the broadcast alert message to the entire host in the network segment whenever an invalid entry is about to be made in the ARP cache. This is based on the counter. If a fake entry has been found then it is updated in the invalid list as follows:

```
IF (ETH_Header! = ARP_Header) Then
IF (entry is on the invalid_list) Then
Increment the count;
If (Count >2) Then
Generate Broadcast Vigilant Message (); //to alerts the hosts in the network segment.
```

Table 3. Features of the proposed system

| SNo | Features | Components |
|---|---|---|
| 1 | Makes ARP Stateful | Components 1 with timestamp, Alert messages |
| 2 | Cryptographic techniques | Not Used |
| 3 | Tables maintained | Modified ARP Cache, Invalid list table |
| 4 | Avoids Broadcast storms without affecting virtualization | Component 1, component 3, and 4 |
| 5 | Mitigates Cache poisoning attacks | Components 2 through 5 |
| 6 | MITM, DoS and /host impersonation can be avoided | All are ARP based cache poisoning attacks |
| 7 | ARP Type | Static and Dynamic |

<table>
<tr><td colspan="2">H/W Type (2B)</td><td colspan="2">Protocol-Type (2B)</td></tr>
<tr><td>H/W length (1B)</td><td colspan="2">Protocol-length (1B)</td><td>Op-code(2B) = 26</td></tr>
<tr><td colspan="4">Source-IP of Gateway (4B)</td></tr>
<tr><td colspan="4">Source –MAC of Gateway(6B)</td></tr>
<tr><td colspan="4">Destination-MAC-Broadcast address (6B)</td></tr>
<tr><td colspan="4">Invalidate (IP-MAC entry) (10B)</td></tr>
<tr><td colspan="4">TS<sub>g</sub> (4B)</td></tr>
<tr><td colspan="4">TS<sub>t</sub> (4B)</td></tr>
</table>

Fig. 6. Broadcast invalidate message format

The message format of the two new messages is depicted in the Figs 7 and 8 respectively.

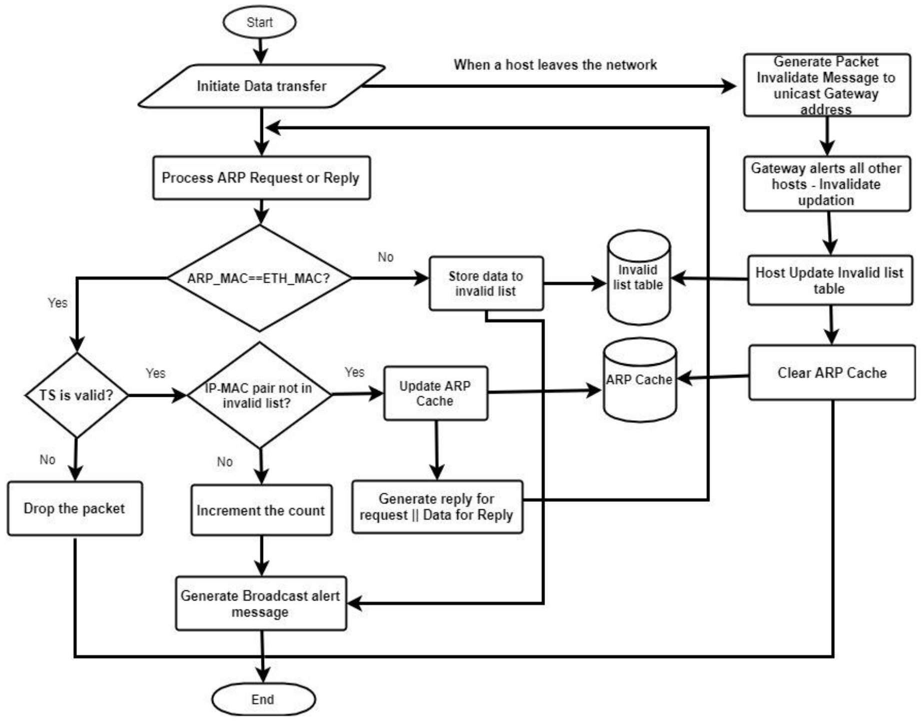

Start
Initiate Data transfer
When a host leaves the network
Generate Packet Invalidate Message to unicast Gateway address
Process ARP Request or Reply
Gateway alerts all other hosts - Invalidate updation
ARP_MAC==ETH_MAC?
No
Yes
Store data to invalid list
Invalid list table
Host Update Invalid list table
TS is valid?
Yes
IP-MAC pair not in invalid list?
Yes
Update ARP Cache
ARP Cache
Clear ARP Cache
No
No
Drop the packet
Increment the count
Generate reply for request || Data for Reply
Generate Broadcast alert message
End

Fig. 7. Flowchart of the proposed system

The message is depicted in Fig. 6. The features and characteristics of the proposed system are depicted in Table 1. The ARP cache will be modified to have one more entry namely Timestamp field: **ARP_CACHE (ARP TYPE, IP, MAC, INTERFACE, TIMESTAMP**) The ARP entries must be cleared for every 10 minutes, in case of dynamics, it can be based on the information being sent and received.

<table>
<tr><td colspan="3">H/W Type (2B)</td><td colspan="3">Protocol-Type (2B)</td></tr>
<tr><td colspan="2">H/W length (1B)</td><td colspan="2">Protocol-length (1B)</td><td colspan="2">Op-code(2B) = 27</td></tr>
<tr><td colspan="6">Source-IP of Host (4B)</td></tr>
<tr><td colspan="6">Source –MAC of Host(6B)</td></tr>
<tr><td colspan="6">Destination-MAC of Gateway (6B)</td></tr>
<tr><td colspan="6">Destination IP of Gateway (4B)</td></tr>
<tr><td colspan="3">Message</td><td colspan="3">Invalidate my details</td></tr>
<tr><td colspan="6">$TS_g$ (4B)</td></tr>
<tr><td colspan="6">$TS_t$ (4B)</td></tr>
</table>

Fig. 8. Unicast alert message format

<table>
<tr><td colspan="3">H/W Type (2B)</td><td colspan="3">Protocol-Type (2B)</td></tr>
<tr><td colspan="2">H/W length (1B)</td><td colspan="2">Protocol-length (1B)</td><td colspan="2">Op-code(2B) = 28</td></tr>
<tr><td colspan="6">Source-IP of host(4B)</td></tr>
<tr><td colspan="6">Source –MAC of host(6B)</td></tr>
<tr><td colspan="6">Destination-MAC-Broadcast address (6B)</td></tr>
<tr><td colspan="3">Invalid IP (4B)</td><td colspan="3">Invalid-MAC (6B)</td></tr>
<tr><td colspan="6">$TS_g$ (4B)</td></tr>
<tr><td colspan="6">$TS_t$ (4B)</td></tr>
</table>

Fig. 9. Broadcast vigilant message format

## 4. Timeline charts

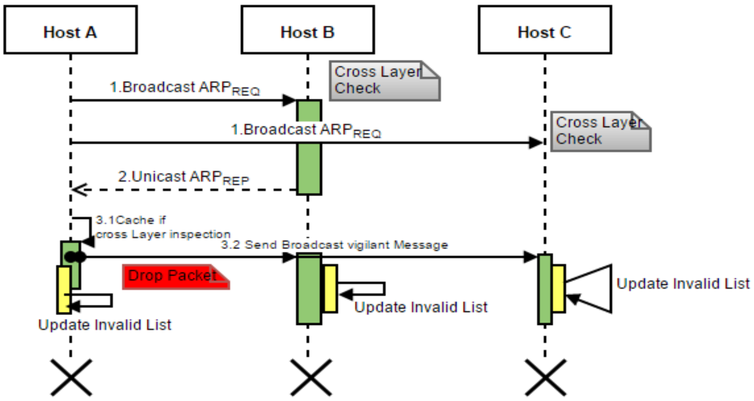


Fig. 10. ARP request response scenario

The timeline charts show the working of the proposed system. The splitting of LDCNs takes place automatically. The packets that are being transmitted are analysed by the packet analyser.

### 4.1. Request response scenario

Consider that host A and Host B are willing to share their information. The normal ARP request-response scenario with cross-layer checking invalid packet detection is depicted in Fig. 10.

### 4.2. Gateway update during VM sprawl or Host migration

Imagine a host named C migrates from this network and join other. It will intimate the gateway with a unicast message about its migration. The gateway, in turn, can send broadcast invalidate message to all the other hosts in the network. The working of this scenario is depicted in Fig. 11.

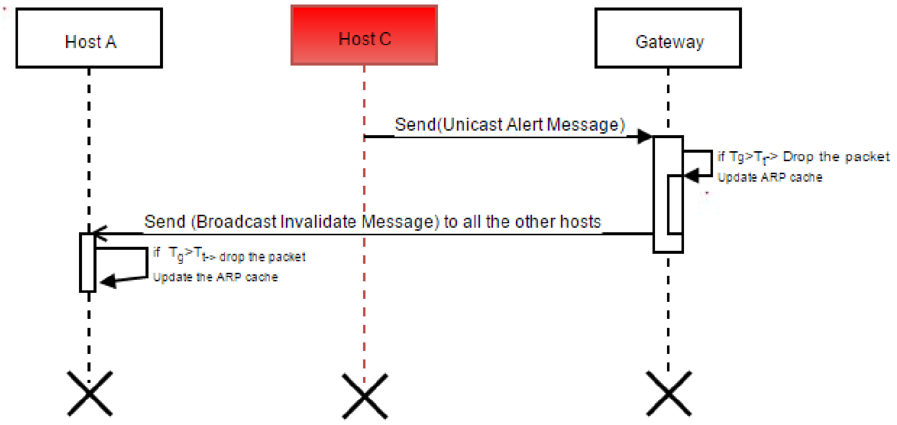
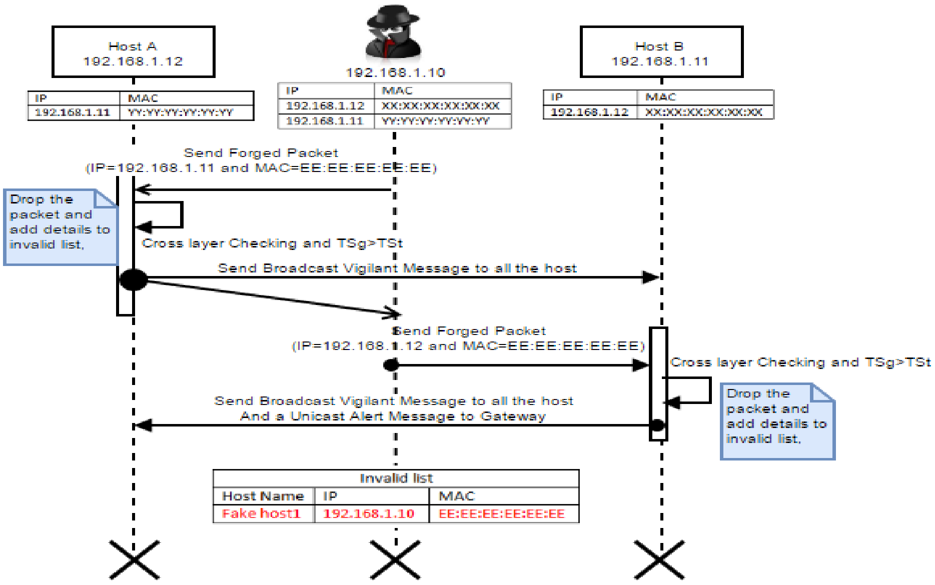

Host A
Host C
Gateway
Send(Unicast Alert Message)
if Tg>$T_r$-> Drop the packet
Update ARP cache
Send (Broadcast Invalidate Message) to all the other hosts
if $T_g$>$T_{t\text{->}}$ drop the packet
Update the ARP cache

Fig. 11. Host C migrates from the network

### 4.3. Defense against MiTM attack


Host A
192.168.1.12
192.168.1.10
Host B
192.168.1.11
Send Forged Packet
(IP=192.168.1.11 and MAC=EE:EE:EE:EE:EE)
Drop the packet and add details to invalid list,
Cross layer Checking and TSg>TSt
Send Broadcast Vigilant Message to all the host
Send Forged Packet
(IP=192.168.1.12 and MAC=EE:EE:EE:EE:EE)
Cross layer Checking and TSg>TSt
Drop the packet and add details to invalid list,
Send Broadcast Vigilant Message to all the host
And a Unicast Alert Message to Gateway

| IP | MAC |
|---|---|
| 192.168.1.11 | YY:YY:YY:YY:YY:YY |

| IP | MAC |
|---|---|
| 192.168.1.12 | XX:XX:XX:XX:XX:XX |
| 192.168.1.11 | YY:YY:YY:YY:YY:YY |

| IP | MAC |
|---|---|
| 192.168.1.12 | XX:XX:XX:XX:XX:XX |

| Invalid list | | |
|---|---|---|
| Host Name | IP | MAC |
| Fake host1 | 192.168.1.10 | EE:EE:EE:EE:EE:EE |

Fig. 12. Defense against MiTM attack

When a host say, host, C has been compromised and targets the victims say host A and host B, will try to send a forged ARP reply to host A and host B. The host A and B will perform the cross-layer check and timestamp validation. On invalidation, the packet is dropped and the host added to invalidate list. This is the worst case. Now the victims are no more victims, they alert the other hosts about this forged attacker and add the host details to invalid list. This scenario is depicted in Fig. 12.

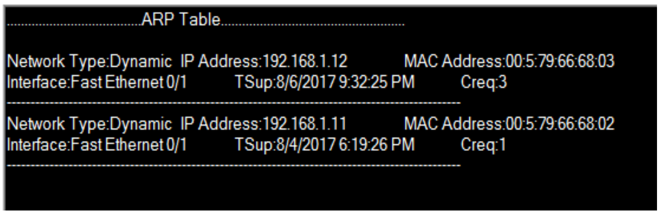
## 5. Comparative analysis of proposed system

As per the six security requirements for defenses against ARP attacks stated by Al-Hemairy, Amin and Trabelsi [27], the proposed system satisfies all of them. Table 4 provides the detailed comparison of various solutions. From the table, it is clear that the proposed system satisfies 5 security solutions out of 6.

Table 4. Comparison with Existing solutions

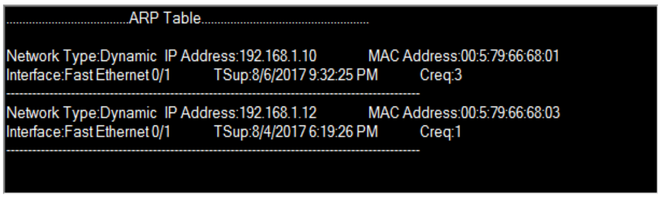
| Features | SDN based | SARP [18] | TARP [16] | GARP [23] | Host IDS [22] | Proposed |
|---|---|---|---|---|---|---|
| Cross-layer Inspection | No | No | No | No | No | Yes |
| ARP Stateful | Yes | Yes | Yes | Yes | Yes | Yes |
| ARP storm Prevention | Yes | No | No/P | YES | Yes | Yes |
| ARP Scan detection | No | No | No | No | No | No |
| Static (S) and Dynamic (D) entries | D | D | D | S&D | S&D | S&D |
| Non-Cryptographic | Yes | No | No | Yes | No | Yes |

## 6. Implementation and results



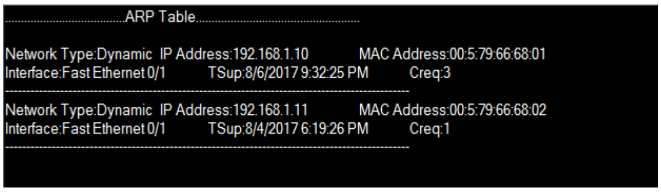
```
.................................ARP Table.................................................

Network Type:Dynamic  IP Address:192.168.1.12          MAC Address:00:5:79:66:68:03
Interface:Fast Ethernet 0/1          TSup:8/6/2017 9:32:25 PM          Creq:3
--------------------------------------------------------------------------------
Network Type:Dynamic  IP Address:192.168.1.11          MAC Address:00:5:79:66:68:02
Interface:Fast Ethernet 0/1          TSup:8/4/2017 6:19:26 PM          Creq:1
--------------------------------------------------------------------------------
```



```
.................................ARP Table.................................................

Network Type:Dynamic  IP Address:192.168.1.10          MAC Address:00:5:79:66:68:01
Interface:Fast Ethernet 0/1          TSup:8/6/2017 9:32:25 PM          Creq:3
--------------------------------------------------------------------------------
Network Type:Dynamic  IP Address:192.168.1.12          MAC Address:00:5:79:66:68:03
Interface:Fast Ethernet 0/1          TSup:8/4/2017 6:19:26 PM          Creq:1
--------------------------------------------------------------------------------
```



```
.................................ARP Table.................................................

Network Type:Dynamic  IP Address:192.168.1.10          MAC Address:00:5:79:66:68:01
Interface:Fast Ethernet 0/1          TSup:8/6/2017 9:32:25 PM          Creq:3
--------------------------------------------------------------------------------
Network Type:Dynamic  IP Address:192.168.1.11          MAC Address:00:5:79:66:68:02
Interface:Fast Ethernet 0/1          TSup:8/4/2017 6:19:26 PM          Creq:1
--------------------------------------------------------------------------------
```

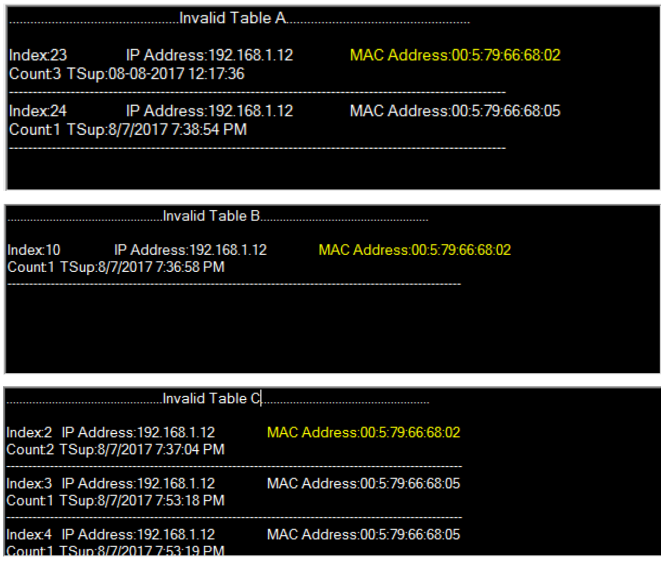
Fig. 13. ARP tables of nodes A, B, and C (from the top to the bottom)

The ARP part of the proposed technique has been implemented and the simulated environment consists of two scenarios:

1. Same network: The three nodes A, B and C bearing 192.169.1.10-00:5:79:66:68:01, 192.169.1.11-00:5:79:66:68:02, 192.169.1.12 -00:05:79:66:68:03 IP – MAC addresses respectively.

The initial ARP table of A, B and C are depicted in the Fig. 13.

2. Different network (Fig. 14).



```
..........................................Invalid Table A..............................................

Index:23          IP Address:192.168.1.12          MAC Address:00:5:79:66:68:02
Count:3  TSup:08-08-2017 12:17:36
-------------------------------------------------------------------------------------------
Index:24          IP Address:192.168.1.12          MAC Address:00:5:79:66:68:05
Count:1  TSup:8/7/2017 7:38:54 PM
-------------------------------------------------------------------------------------------
```

```
..........................................Invalid Table B..............................................

Index:10          IP Address:192.168.1.12          MAC Address:00:5:79:66:68:02
Count:1  TSup:8/7/2017 7:36:58 PM
-------------------------------------------------------------------------------------
```

```
..........................................Invalid Table C..............................................

Index:2  IP Address:192.168.1.12          MAC Address:00:5:79:66:68:02
Count:2  TSup:8/7/2017 7:37:04 PM
-------------------------------------------------------------------------------------
Index:3  IP Address:192.168.1.12          MAC Address:00:5:79:66:68:05
Count:1  TSup:8/7/2017 7:53:18 PM
-------------------------------------------------------------------------------------
Index:4  IP Address:192.168.1.12          MAC Address:00:5:79:66:68:05
Count:1  TSup:8/7/2017 7:53:19 PM
```

Fig. 14. Invalid tables of nodes A, B and C (from the top to the bottom)

Table 5. Nomenclature

| Symbols/Notation | Description |
|---|---|
| $ARP_{REQ}$, $ARP_{REP}$ | ARP Request and Reply Packet |
| $OP_{REQ,}$ $OP_{REP}$ | ARP Request and Reply opcodes takes a value 1and 2, respectively |
| $ARP_{MAC}$, $ETH_{MAC}$ | MAC address in ARP header and Ethernet header |
| $IP_{SRC}$, $IP_{TAR}$ | IP address of Source and destination host |
| $IP_{A}$, $IP_{B}$ | IP address of host A and host B |
| $MAC_{SRC}$, $MAC_{TAR}$ | MAC address of Source and destination host |
| $MAC_{A}$, $MAC_{B}$ | MAC address of host A and host B |
| $BDC_{IP\,,}$ $BDC_{MAC}$ | Broadcast IP and MAC address takes the values 255.255.255.255 and ff:ff:ff:ff:ff:ff |
| $MUL_{IP}$, $MUL_{MAC}$ | Multicast IP and MAC addresses |
| $UNI_{MAC}$ | Unicast MAC address |
| $ARP_{A}$, $ARP_{B}$ | ARP tables of host A and host B |
| $SEND_{ACK}$ $SEND_{PKT}$ | Unicast message to A about TS expiry and start data communication |
| TS | Timestamp |
| $INV_{ListA}$, $INV_{ListB}$ | Invalid list table of host A and host B |
| $BDC_{VIG}$ | Broadcast vigilant message |
| $UNI_{ALT}$ | Unicast alert message |
| CLK | System clock time |
| NULL $_{IP}$ | NULL IP address |
| $IP_{VAL}$, $MAC_{VAL}$ | Valid IP and MAC address |

It has three nodes A, B, C and E, F, G in subnet 1 and subnet 2, respectively. The IP-MAC pairs of E, F and G are 192.168.0.14-00:5:79:66:68:05, 192.168.0.15-00:5:79:66:68:06 192.168.0.16 - 00:5:79:66:68:07

When host A wants to communicate with host C it will be done normally as the details of C are available in A's ARP table. But after some time the ARP caches will be cleared A has to send ARP request for updating its cache. When a host is sending a request to A with 192.168.1.12-00:5:79:66:68:02, invalid tables of all hosts will be updated with this information because MAC address of this node is not equal to its Eth-MAC address.

## 7. Performance evaluation of proposed system

The ARP portion of the proposed system has been implemented and it complies with the security requirements stated by Trabelsi et al [23]. The algorithm performs the cross-layer inspection with ARP header (28 bytes) and Ethernet header (64 bytes and may vary) which requires a fixed number of steps to carry out this pre-processing say $O(1)$ since the fields in the headers are constant. The ARP request processing algorithm performs one sequential scan in the invalid list and ARP table. If size the tables are m and n then it requires $O(m)$ and $O(n)$. Then it performs one comparison before generating a reply which is $O(1)$.

The ARP reply processing algorithm performs two sequential searches as request processing which takes $O(m)$ and $O(n)$ respectively. Both algorithms perform incrementing/decrementing counter's values and generate packets requiring to be sent. This cost may vary depending on the availability of the hosts. As a whole the total time complexity of Algorithms will be $O(m) + O(n)$; depending on ports $m$ and $n$ may be negligible.

**Performance evaluation of proposed system against ARP, SARP, TARP, KARP, and GARP.** The proposed ARP is capable of handling following malicious types packets:

**Abnormal/malicious packets related to ARP Request**

PKT1 →Source ($IP_{INV}$ $MAC_{VAL}$)

PKT2 →Target ($UNI_{MAC}$)

PKT3 →Source ($ETH_{MAC}$ != $ARP_{MAC}$)

PKT4 →Source($NULL_{IP}$ ||$BDC_{IP}$|| $MUL_{IP}$)

PKT5 →Source($NULL_{MAC}$ ||$MUL_{MAC}$|| $BDC_{MAC}$) or multicast address not within the subnet

**Abnormal/malicious packets related to ARP Reply**

PKT6 →Source($IP_{VAL}$ – $MAC_{INV}$)

PKT7 →Source($ETH_{MAC}$ != $ARP_{MAC}$)

PKT8 →Target ($BDC_{MAC}$)

PKT9 →Target ($IP_{INV}$ – $MAC_{VAL}$)

PKT10 →Target ($ETH_{MAC}$ != $ARP_{MAC}$)

PKT11 →Source($NULL_{IP}$ || $BDC_{IP}$|| $MUL_{IP}$)

PKT12 →Source($NULL_{MAC}$ ||$MUL_{MAC}$|| $BDC_{MAC}$) or multicast address not within the subnet

**Abnormal/malicious packets related to ARP Reply/Request**

PKT13 → $ARP_{REQ}$ or $ARP_{REP}$ by the host that has left the network

Of these 13 malicious packets PKT1, PKT6, PKT4, PKT9, and PKT13 are suspicious to corrupt the traditional ARP table but not the proposed one. Fig. 13 depicts the malicious packets detection and prevention of traditional ARP, SARP, TARP, GARP and Proposed Solution. The timestamp feature of the Proposed ARP avoids the replay attacks. As the difference between the timestamp and the local clock can be fixed not more than the 20 s, replay attacks can be prevented. This difference depends on the network administrator.

By detecting PKT9, PKT12, PKT3 the host impersonation and MiTM can be exploited. While the packets PKT1, PKT4, PKT10 help to avoid IP spoofing, the packets dealing with reply will encounter replay attacks and other avoid ARP cache poisoning. The proposed system is compared to other systems based on the packet detection rate which is calculated as:

**Detection rate (%) = (# abnormal packets detected/ #packets received)**

The abnormal packets detection rate is compared to other systems and the result is depicted in the graph Fig. 15. The proposed system is compared to ARP, SARP [18], GARP [24], TARP [16] and EARP [19].

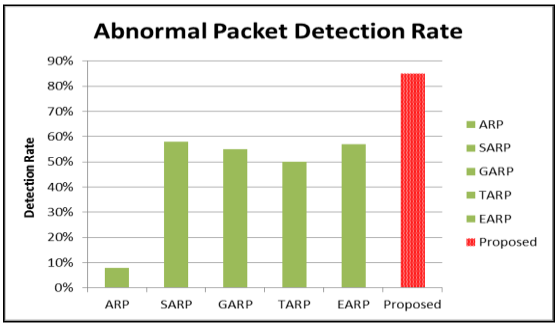


Fig. 15. Abnormal Packet detection rate

## 8. Conclusions and future work

Today's large Data Centre networks are subjects to broadcast storms due to improper configuration of the devices or maybe because of loops in the network. Because of dynamic nature of these DCNs, it is difficult for the Layer 2 protocols to cope up with the heavy broadcast traffic. So in this paper Layer 2 ARP protocol is taken and a brief note on big data and datacentres is provided to illustrate the problem they face by using this ARP protocol in large DCNs. Also, the proposed technique overcomes the issues related to stateless nature of ARP and mitigates the attacks like MiTM, DoS and host impersonation. The proposed system has been compared to various existing solutions and proved to be better than others. The future study is to detect ARP Scanning and explore the issues with neighbor discovery feature of IPv6; also to simulate the proposed method in LDCN.

# References

1. Laney, D. 3D Data Management: Controlling Data Volume, Velocity and Variety. – Gartner. Issued: 6 February 2001, accessed December 2015.
**http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf**
2. Damiani, E., C. A. Ardagn, F. Zavatarelli, E. Rekleitis, L. Marinos. Big Data Threat Landscape and Good Practice Guide. European Union Agency for Network and Information Security (ENISA), January 2016.
3. Jeyanthi, N., N. S. N. Iyengar, C. S. N. Iyengar. Escape-on-Sight: An Efficient and Scalable Mechanism for Escaping DDoS Attacks in Cloud Computing Environment. – Cybernetics and Information Technologies, Vol. **13**, 2013, No 1, pp. 46-60.
4. Thandeeswaran, R., S. Subhashini, N. Jeyanthi, M. A. S. Durai. Secured Multi-Cloud Virtual Infrastructure with Improved Performance. – Cybernetics and Information Technologies, Vol. **12**, 2012, No 2, pp. 11-22.
5. Narten, T., M. Karir, I. Foo. Address Resolution Problems in Large Data Center Networks. No RFC 6820, 2013.
6. Prabadevi, B., N. Jeyanthi. A Framework to Mitigate ARP Sniffing Attacks by Cache Poisoning. – International Journal of Advanced Intelligence Paradigms, 2016 (in Press).
7. Jeyanthi, N., M. P. C. Kumar. A Virtual Firewall Mechanism Using Army Nodes to Protect Cloud Infrastructure from DDoS Attacks. – Cybernetics and Information Technologies, Vol. **14**, 2014, No 3, pp. 71-85.
8. Jeyanthi, N., R. Thandeeswaran, J. Vinithra. RQA Based Approach to Detect and Prevent DDoS Attacks in VoIP Networks. – Cybernetics and Information Technologies, Vol. **14**, 2014, No 1, pp. 11-24.
9. Dineen, T. Broadcast Storm Mitigation on Ethernet Networks. – Industrial Commuication, Schneider Electric Industries SAS. White paper.
**http://tinyurl.com/jq9l3n6.**
10. Greenberg, A., J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, S. Sengupta. VL2: A Scalable and Flexible Data Center Network. – ACM SIGCOMM Computer Communication Review, Vol. **39**, 2009, No 4.
11. Greenberg, A., P. Lahiri, D. A. Maltz, P. Patel, S. Sengupta. Towards a Next Generation Data Center Architecture: Scalability and Commoditization. – In: Proc. of ACM Workshop on Programmable Routers for Extensible Services of Tomorrow, ACM, 2008, pp. 57-62.
12. Mysore, N., R. A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, A. Vahdat. Portland: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric. – ACM, SIGCOMM Computer Communication Review, Vol. **39**, 2009, No 4, pp. 39-50.
13. Joe, T., R. Perlman. Transparent Interconnection of Lots of Links (TRILL): Problem and Applicability Statement. No RFC 5556, 2009.
14. Kim, M. C. C., J. Rexford. Floodless in SEATTLE: A Scalable Ethernet Architecture for Large Enterprises. – In: Proc. of ACM SIGCOMM Conference on Data communication (SIGCOMM’08), 2008.
15. Umar, K., M. K. Gardner, E. J. Brown, W.-C. Feng. Seamless Migration of Virtual Machines Across Networks. – In: 22nd International Conference on Computer Communication and Networks (ICCCN’13), IEEE, 2013, pp. 1-7.
16. Mauricio, A., M. Portolani. Data-Center Network Architecture. U.S. Patent 7,643,468, Issued 5 January 2010.
17. Prabadevi, B., N. Jeyanthi. A Mitigation System for ARP Cache Poisoning Attacks. – In: ICC-ACM 2nd International Conference on Internet of Things Data, and Cloud Computing, Churchill, 2017 (Accepted).
18. Khandelwal, A., N. Jain, S. Kamara. Attacking Data Center Networks from the Inside. EECS.
**https://people.eecs.berkeley.edu/~anuragk/papers/dcn.pdf**

19. Ma, H., H. Ding, Y. Yang, Z. Mi, J. Y. Yang, Z. Xiong. Bayes-Based ARP Attack Detection Algorithm for Cloud Centers. – Tsinghua Science and Technology, Vol. **21**, 2016, No 1, pp. 17-28.

20. Lootah, W., W. Enck, P. McDaniel. Tarp: Ticket-Based Address Resolution Protocol. – Computer Networks, Vol. **51**, 2007, No 15, pp. 4322-4337.

21. Dessouky, M. M., W., Elkilany, N. Alfishawy. A Hardware Approach for Detecting the ARP Attack, in Informatics and Systems (INFOS). – In: Proc. of 7th International Conference on Informatics and Systems, Cairo, Egypt, 2010, pp. 1-8.

22. Bruschi, D., A. Ornaghi, E. Rosti. S-Arp: A Secure Address Resolution Protocol. – In: Proc. of 19th Annual, Computer Security Applications Conference, 2003, pp. 66-74.

23. Nam, S. Y., D. Kim, J. Kim. Enhanced Arp: Preventing Arp Poisoning-Based Man-in-the-Middle Attacks. – Communications Letters, IEEE, Vol. **14**, 2010, No 2, pp. 187-189.

24. Kumar, S., S. Tapaswi. A Centralized Detection and Prevention Technique Against Arp Poisoning. – In: International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec'12), Kuala Lumpur, Malaysia, 2012, pp. 259-264.

25. Neminath, H., S. Biswas, S. Roopa, R. Ratti, S. Nandi, F. Barbhuiya, A. Sur, V. Ramachandran. A DES Approach to Intrusion Detection System for Arp Spoofing Attacks. – In: 18th Mediterranean Conference on Control & Automation (MED'10), Marrakech, Morocco, 2010, pp. 695-700.

26. Barbhuiya, F., S. Biswas, N. Hubballi, S. Nandi. A Host Based DES Approach for Detecting Arp Spoofing, Computational Intelligence in Cyber Security (CICS). – In: IEEE Symposium on, Paris, France, 2011, pp. 114-121.

27. Al-Hemairy, M., S. Amin, Z. Trabelsi. Towards More Sophisticated ARP Spoofing Detection/Prevention Systems in LAN Networks. – In: Proc. of 2009 International Conference on the Current Trends in Information Technology (CTIT'09), 2009.

28. Dangol, S., S. Selvakumar, M. Brindha. Genuine ARP (GARP). – ACM SIGSOFT Softw. Eng. Notes, Vol. **36**, 2011, No 4, p. 1.