A Subnet Based Intrusion Detection Scheme for Tracking down the Origin of Man-In-The-Middle Attack

S.Vidya[1] and R.Bhaskaran[2]

[1] Department of Computer Science, Fatima College
Madurai, Tamil Nadu, India

[2] School of Mathematics, Madurai Kamaraj University
Madurai, Tamil Nadu, India

Abstract

The Address Resolution Protocol (ARP), has proved to work well under regular circumstances, but it is not equipped to cope with malicious hosts. Several methods to mitigate, detect and prevent these attacks do exist for the gateways/routers and nodes. This work is focused towards developing our own tailor made Intrusion Detection technique at the subnet level and we present an algorithm that detects the source of ARP poisoning in the Man-in-the-Middle attack. It is designed to detect both the attack and the attacker. The algorithm uses filtering rules to capture the network traffic and pass the IP packets through four phases. After the first three phases, the algorithm is made to raise an alarm on potential ARP poisoning to the user, if one exists, and the fourth phase detects the source IP that has initiated the attack and raises another alarm. This method works successfully even if there is more than one MITM attacker in the subnet. There is a proof of concept implemented for this algorithm. As a result of this experiment, it was found that the Windows 7 Operating System is also vulnerable to ARP attacks as the earlier versions of Windows.

Keywords: *ARP Poison, Intrusion Detection System (IDS), Media Access Control (MAC), Man-in-the-Middle (MITM) attack, Stateless protocol*

1. Introduction

In a Local Area Network (LAN) environment, data transfer takes place by making use of Media Access Control (MAC) addresses. MAC address operates at the Data Link layer of TCP-IP protocol stack. Communications in LAN require IP addresses to be converted into MAC addresses. Address Resolution Protocol (ARP) does this network address translations. To get good performance, ARP maintains an internal cache of IP addresses and the corresponding MAC addresses. When a node wants to send data, it searches in the ARP cache to find out the MAC address of the destination IP [1].

A MAC address consists of 48 bits (6 octets), whereas IPv4 address consists of 32 bits (4 octets). ARP deals with the two kinds of packets – ARP request and ARP reply [2].

Fig.1 shows how a typical ARP protocol communication happens. The machine with IP address 192.168.0.1 sends a broadcast request, asking for the MAC address of the machine whose IP address is 192.168.0.3. Once the request is received by 192.168.0.3, it sends a reply to 192.168.0.1 that, 00:E0:FE:09:C2:11 is the MAC address [3].

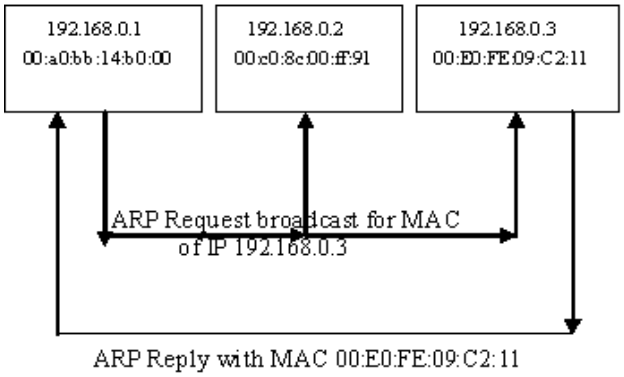

Fig.1 Communication in LAN using ARP

The potential vulnerabilities in this ARP way of communication are researched and presented here with an algorithm to detect the source of ARP poisoning in MITM attack. This paper is organized as follows : In Section 2 the problem is defined and throws light on the existing methods and attempts made by researchers, in Section 3 the proposed solution is presented and in Section 4 explains the lab environment used and the various observations that were made during the implementation of the proposed solution and finally Section 5 concludes the presentation.

2. Problem Defined

2.1 ARP Poisoning

In ARP spoofing attacks, attackers send a fake ARP response packet pretending that they own the MAC address that is wanted. This causes the requester to cache the fake data. As a result, the victim's machine, which has incorrectly cached information about the owner of the IP address, sends all traffic destined for that IP address to the attacker's node [4]. When done properly, the victim will have no idea that the information is redirected to the attacker. The process of updating a target computer's ARP cache with a forged entry is referred to as "poisoning" [5].

The trick is to associate the attacker's MAC address with the IP address of the victim instead. The network is thus made open to vulnerabilities like Man-in-the-Middle attack (MITM), Denial-of-Service (DoS) attack and such [6].

In Fig.2 the machine with IP address B does the MITM attack. So the ARP cache of machine with IP address A and that of IP address C are poisoned. All information transferred between machine A and machine C are now intercepted by machine B.

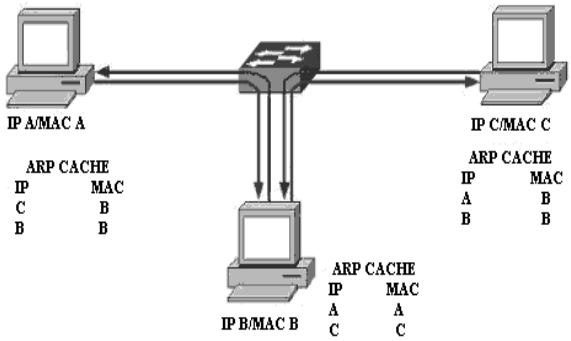

Fig.2 MITM with ARP Poisoning in a switched environment

This is extremely potent, when we consider that, not only can computers be poisoned, but routers/gateways and any device with an IP address as well.

2.2 Defense strategies against ARP poisoning

Sean Whalen [5] in his work has observed that there is no universal defense against ARP spoofing. In fact, the simple possible defense is the use of static ARP entries. Since static entries cannot be updated, spoofed ARP replies are ignored. But the overhead in deploying these tables, as well as keeping them up to date, is not practical for most LANs. Also, the other method of using Port Security does not prevent ARP spoofing.

Aside from these two methods, the only remaining defense is detection. Free tools like Arpwatch [7], Ettercap [8], are working based on defense by detection mechanism, and do not provide complete defense. Instead, at times they even seem to increase the work of the network administrator[9]. Allam Appa Rao et al [10] in their work have suggested software development based on Jpcap for IDS, involving various network vulnerabilities but not specifically meant for the vulnerabilities of ARP. The work of Wenjian Xing et al [11] is about a defense mechanism for ARP spoofing which mainly focus towards the attack of the network gateway. In it the packets are analysed against the IP of the gateway.

With the inherent vulnerabilities of the ARP protocol no complete solution is devised so far. Hence the obvious practical way to go is to devise subnet specific contextual solutions. The solution presented here applies to detecting ARP vulnerabilities due to intrusion in data link layer of the ISO-OSI stack.

3. Proposed Solution

Attacks against layer-2 of ISO-OSI stack, the data link layer, ranges from various ARP attacks like the cache poisoning for wired clients to de-authentication of wireless clients. Fairly simple to implement, these attacks can often go unnoticed by intrusion analysts since intrusion detection systems typically look at the network layer and above to detect attacks. Whatever method of attack an attacker uses to attack the data link layer, in all cases, an adversary attempts to compromise confidentiality, authentication or availability of information. The attacks succeed for the most part because of the lack of fine-grain controls for the data link layer. While layer 2 is considered a less novel platform for attacks, layer 2 attacks continue to trouble the networked systems. The implementation of each attack is unique [12]. This being the current scenario, led to the development of this new algorithm and software, proof to examine network traffic for data link layer attack and proactively respond to attacks.

The algorithm presented here detects the source of ARP poisoning in real time by monitoring traffic flow and raise alarm to the network administrators, detecting both the attack and the attacker.

3.1 Intrusion Detection Algorithm for MITM attacks

The algorithm has four phases as shown in Fig 3. The first alarm is raised when ARP poisoning is discovered. The second alarm is raised when the origin of poisoning, the IP address from which the poisoning is initiated, is discovered by the algorithm.

The first of the four phases is the process of Filtering, where the rules from the rule base are applied on the incoming IP packets and the packets are captured from the online network traffic. The entire Ethernet traffic is passed through the filtering stage and only the filtered set of packets based on the rule base is retained and passed on to the next stage for further process. In the second phase of Segregation, from the full packet with all the fields, specific fields needed for process are segregated.

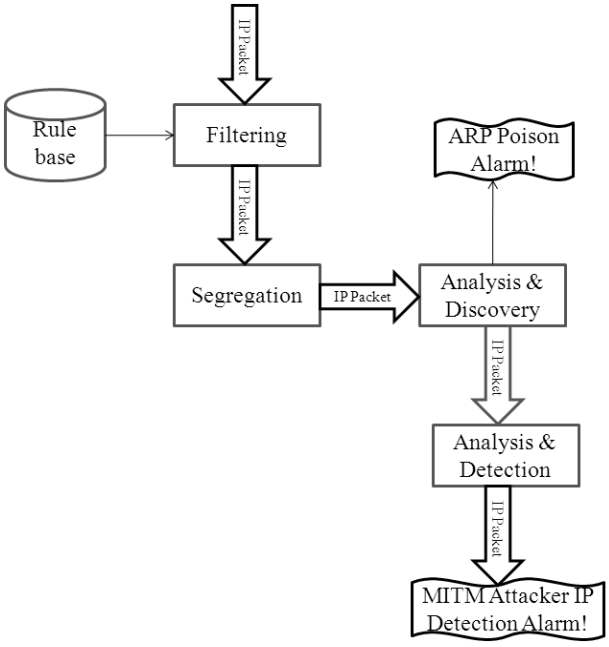

Fig.3 The Algorithm for detection of origin of MITM attack

The ARP Packet structure is as in Fig 4. All the fields in the packet are not needed for the process and hence, only the required fields, the Source IP address and the Source MAC address fields are segregated from the packet using the two functions of Jpcap.

The two functions are getSenderProtocolAddress() and getSenderHardwareAddress(). The segregated information is passed on to the third phase for analysis. As a result of analysis of the details from the captured packets, the ARP poison, if any, is discovered in the subnet and an alarm is raised.

As shown in the Fig.4, the actual ARP packet comprises of nine fields as Hardware type, Protocol type, Hardware size, Protocol size, Operation, Source and Target Hardware address and Source and Target IP address.

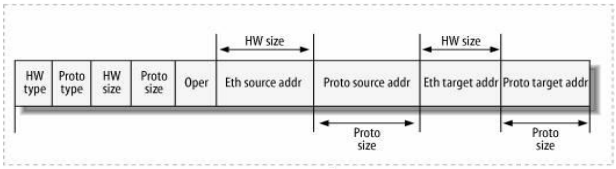

Fig.4 ARP Packet Structure

The last phase of the algorithm is to identify the culprit computer which has initiated the ARP poisoning of the victims and is performing the MITM attack. This last phase is called for, only when the earlier three phases detects an ARP poison in the subnet. In case, there is no poison detected, the last phase has no role to play and the process continues.

In the fourth phase, it captures more packets and analyses the source IP address and the MAC address after segregating the two fields from the whole ARP packet. By sufficient analysis of the information, the algorithm is able to conclude on the attacker machine's IP.

The pseudocode of the proposed Algorithm is as follows

```
Loop //Capture ARP Packets
  Split (ARP Packet ) // the IP, MAC,STATUS
    If  ARP Type != 'Unknown"  then
          Add (IP, MAC) to AddrArray
    Loop
      Split (Next ARP Packet)  //IP, MAC,STATUS
       If  NextARP Type != "Unknown" then
                  Add(IP,MAC) to NewArray
       Loop //Finding possible Poisoned IPs
         If AddrMAC=NextMAC then
           if  AddrIP != NextIP then
                    Raise Poison discovered Alarm
            Loop  //Collecting Poisoned IPs
           If ErrIP != AddrIP  and ErrMAC !=
                       NewMAC then
             Add (AddrIP,NewMAC) to ErrArray
           End if
           If ErrIP!=NewIP and ErrMAC !=NewMAC then
                    Add (NewIP,NewMAC) to ErrArray
           End if
         Else
              NOOP
         End if
    Else
      NOOP
```

```
    End if
      Loop  //Find Attacker
          If (ErrIP[Current]=ErrIP[Next]) then
              ErrIPCount++
          End if
          If  (ErrIPCount>1) then
                  Display (IP,MAC)
                  Raise Alarm attacker Found
          End if
    Else
        Fetch next ARP packet
    End if
Else
        Fetch next ARP packet
End if
```

3.2 Jpcap

The algorithm designed is implemented as a software written in JAVA using the Jpcap, a kind of wrapper class for winpcap/libpcap[13]. Jpcap is a Java library for capturing and sending network packets. It has been tested in Microsoft Windows (98/2000/XP/Vista) and Linux (Fedora, Mandriva, Ubuntu), MacOS X (Darwin), FreeBSD, and Solaris. **Now this work has shown that it works well in Windows 7 also**. It is open source and is licensed under GNU LGPL. It provides facilities to capture raw packets live from wire, save captured packets, filter the packets according to user-specified rules before dispatching them to the application. These facilities have made the use of Jpcap in this software development, trivial. Jpcap does not block, filter or manipulate the traffic generated by other programs on the same machine. Therefore, it does not provide the appropriate support for applications like traffic shapers, QoS schedulers and personal firewalls.

4. The Experiment and Results

4.1 The Experiment

The software was implemented and load tested in a simulated environment, with internet connection, from inside a University network used by the staff and students. The tool used to poison and carry out the MITM attack, was Cain and Abel [14]. The complete experiment involved Windows XP and Windows 2000 Operating Systems. The IP addresses, for the machines in the lab environment, connected to the internet, were dynamically assigned. The entire network was under a single windows domain.

If there are more than one MITM attacks in the subnet, all the attacks are detected by this algorithm based on the captured traffic flowing between poisoned machines and the attacker.

The algorithm was tested with following two experiments

1. A simple MITM attack with one attacker between two victim IPs.
2. The effect of two machines trying to attack each other at the same time.

In both the experiments the algorithm was able to detect the attacking IP/IPs creating the MITM attack.

Detecting the origin of the attack is possible only if there is active transfer of data between the victim machines. In the lab environment, minimal traffic such as a simple ping command between the victims triggers the detection of the second alarm.

The lab environment thus setup with the software developed was able to detect ARP poisoning attacks and the source of attack and raised the alarm, every time there was an attack. To cross check the authenticity of the program executed, it was run along with a free ARP attack detection tool, the XArp [15]. Every time there was a poison taking place, XArp would pop up. The program developed was also able to detect every time the attack was happening. The XArp tool does not directly point out the attacker, but shows a continous list of the changes that the MAC undergoes for each IP. Otherwise the **success rate was seen to be the same for XArp and the software developed.**

Since the detection algorithm is implemented in promiscuous mode, any computer implementing this software within the subnet can detect that there is poisoning in the subnet and identify the attacker/attackers.

4.2 Observations on Experiment 1

Contents of ARP cache of the victims and of the attacker, before and after the poison attack, in the lab is presented here. In all the screen shots below, the dots highlight the IP and MAC addresses that are involved in the MITM attack.

The Cain and Abel tool was executed from the attacker machine with IP 192.168.21.193 and the two victims involved in the attack were with the IP 192.168.21.187 and 192.168.21.178.

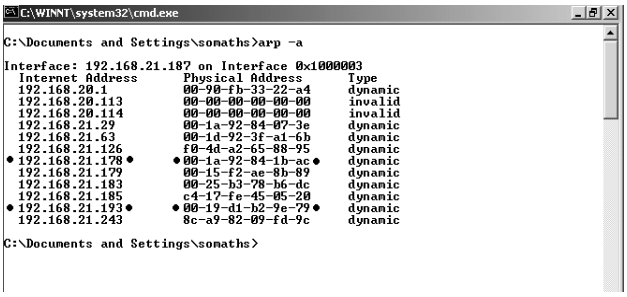

Fig.5 The ARP cache of the victim machine with IP 192.168.21.187 before MITM attack

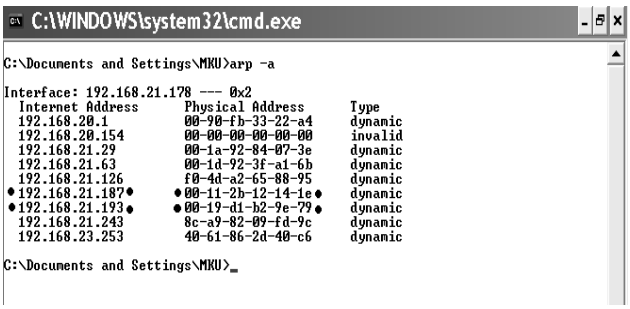

Fig.6 The ARP cache of the victim machine with IP 192.168.21.178 before MITM attack

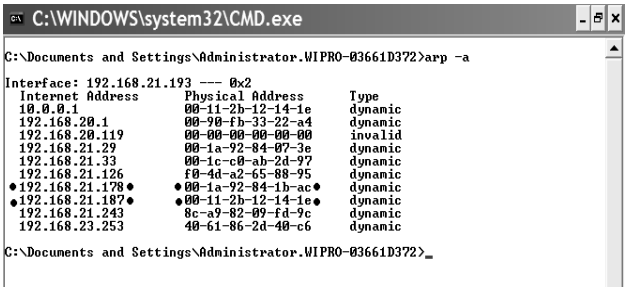

Fig.7 The ARP cache of the machine with IP 192.168.21.193 that implements the MITM attack, before the attack

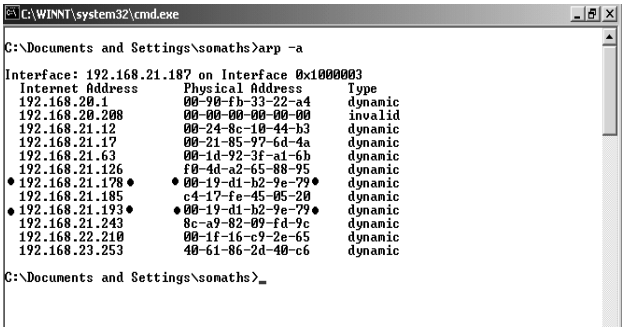

Fig.8 The Poisoned cache of the victim machine with IP 192.168.21.187

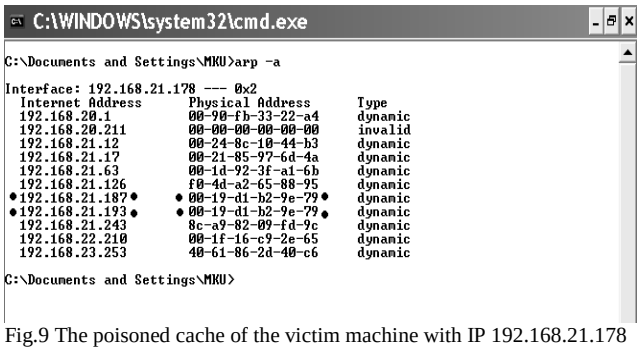

Fig.9 The poisoned cache of the victim machine with IP 192.168.21.178

4.3 Observations on Experiment 2

The IP addresses involved in the two simultaneous MITM attacks are, one set of addresses as in experiment 1 and the other set is with the attacker IP 192.168.21.187 and the two victims being 192.168.21.193 and 192.168.21.178.

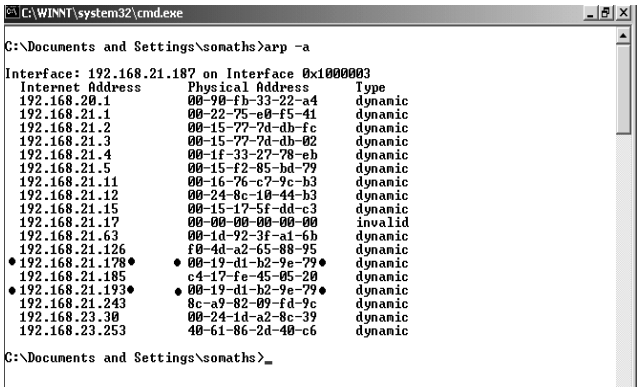

Fig.10 The ARP cache of IP 192.168.21.187 the attacker and the victim

The Fig.11 depicts the poisoned cache of the victim with both the attacker machines and their MAC address changed.

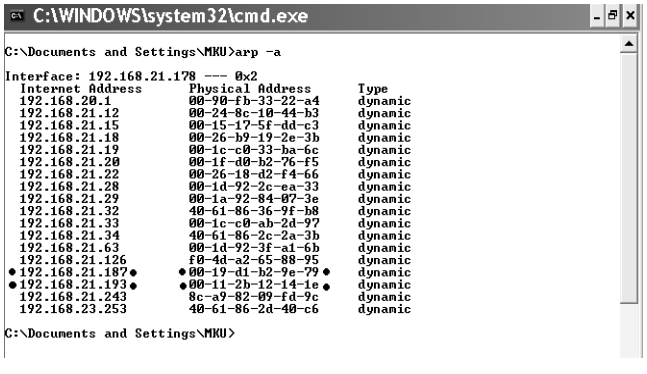

Fig.11 The ARP cache of IP 192.168.21.178 the victim of two simultaneous attacks

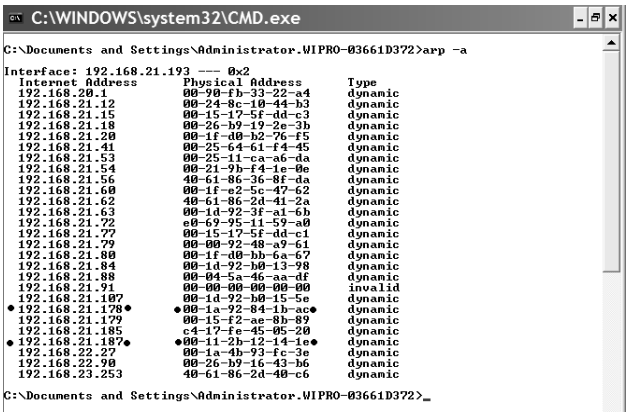

Fig.12 The ARP cache of IP 192.168.21.193, the attacker and the victim

4.4 Results

In Fig.13 and Fig.14 the result is enclosed in a box showing the discovery of poison and detection of the attacker. The dots show the packets of the victim and attacker machines captured for analysis

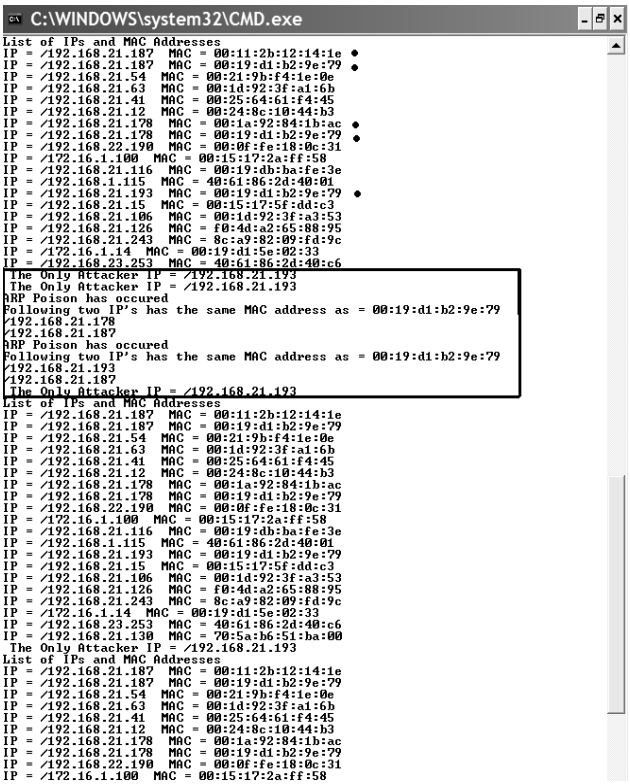

Fig.13 Screen shot of the first experiment

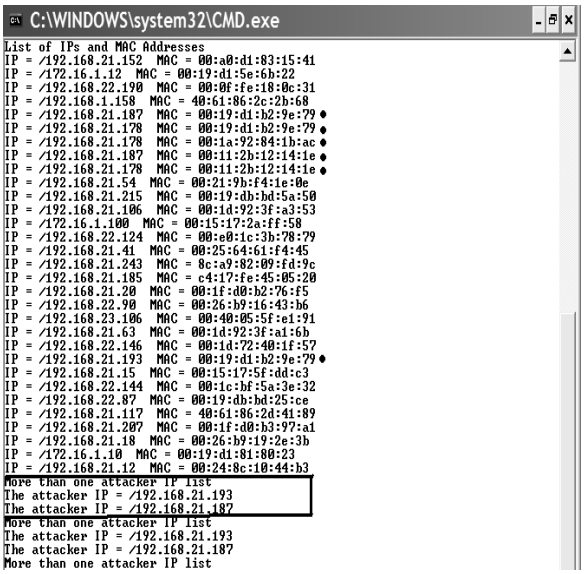

Fig.14 Screen shot of the second experiment. The results enclosed in the box shows the poison discovery and both the attackers identified

5. Finding

As a result of conducting this experiment, *it was found that Windows 7 was equally vulnerable to ARP attacks as the earlier versions of Windows.* It is documented earlier that the various Operating Systems that are vulnerable to ARP attacks are Windows 95/98/2000/NT [16], Windows XP [17]. The OS that is less vulnerable to ARP attacks is Sun Solaris System [18].

6. Conclusion

As this algorithm helps to detect attacks at an early stage, the software could be used as both continuous monitoring tool and as preventive maintenance tool. The alarms raised by the software are useful in taking necessary actions against the malicious IPs. The speed with which internetworking and malicious attacks on the internet grows, the cost of cleaning and bringing business critical systems back online is sky-rocketing. The current trend in the internetworking being so, with the automation and increase in speed and sophistication of attack tools, each year doubling of newly discovered vulnerabilities and administrators finding it difficult to keep up to date with patches, increasing permeability of firewalls with firewall friendly protocols that are designed to bypass firewall configurations, increasing threat from infrastructure attacks like Distributed DoS, Worms or attack on Domain Name Server cache and so on [19], finding generic solutions too is becoming infeasible. Hence, as a subnet based solution,

this algorithm provides a means of prevention by detection and thus provides an inexpensive method to protect systems. Also this algorithm works at data link layer while most of the known methods are based on other layers of ISO-OSI model.

References

[1] Behrouz A. Forouzan, "TCP/IP Protocol Suite", Fourth Edition, Tata McGraw Hill, pp. 220-223, 2010.
[2] D.Plummer, "An Ethernet Address Resolution Protocol", Nov. 1982, RFC 826
[3] S.Vidya, R.Bhaskaran, "ARP storm Detection and Prevention Measures", IJCSI International Journal of Computer Science Issues, Vol.8, Issue 2, March 2011, ISSN(Online): 1694-0814, pp-456-460.
[4] Gopi Nath Nayak and Shefalika Ghosh Samaddar, "Different flavors of Man-in-the-Middle attack, Consequences and feasible solutions", 978-1-4244-5540-9/10/$26.00 ©2010 IEEE.
[5] Sean Whalen, "An Introduction to ARP Spoofing", www.machacking.net/kb/files/arpspoof.pdf, April 2001.
[6] Wikipedia, The Free Encyclopedia, "ARP Spoofing", http://en.wikipedia.org/wiki/ARP_Spoofing, 2011.
[7] L. N. R. Group, "arpwatch, the Ethernet monitor program; for keeping track of Ethernet/ip address pairings," ftp://ftp.ee.lbl.gov/arpwatch.tar.gz, (Accessed on April 7,2011)
[8] Alberto Ornaghi, Marco Valleri, "Ettercap", http://ettercap.sourceforge.net/
[9] Andre P.Ortega, Xavier E. Marcos, Luis D. Chiang, Cristina L. Abad, "Preventing ARP cache poisoning Attacks: A Proof of Concept using OpenWrt", 978-1-4244-4550-9/09/$25.00 ©2009 IEEE
[10] AllamAppaRao, P.Srinivas, B.Chakravarthy, K.Marx, P.Kiran, " A Java Based Network Intrusion Detection System(IDS)", in the proceedings of the 2006 International Journal of Modern Engineering INTERTECH Conference , Session ENG 206-118
[11] Wenjian Xing, Yunlan Zhao, Tonglei Li, "Research on the defense against ARP spoofing attacks based on Winpcap", in Second International Workshop on Education Technology and Computer Science, 2010, DOI 10.1109/ETCS.2010.75, 978-0-7695-3987-4/10 $26.00 © 2010 IEEE
[12] TJ O'Connor, "Detecting and Responding to Data Link Layer Attacks", SANS Institute InfoSec Reading Room, Oct 13, 2010, http://www.sans.org/reading_room/whitepapers/detection/detecting-responding-data-link-layer-attacks_33513, 2010.
[13] Keita Fujii, "Jpcap – a Java library for capturing and sending network packets", http://netresearch.ics.uci.edu/kfujii/Jpcap/doc/, 2007.
[14] Massimilliano Montoro, "Cain and Abel", http://www.oxid.it/, 2011
[15] Chrismc de, "XArp", http://www.chrismc.de/development/xarp/,2010
[16] Roudha Khcherif, "ARP cache poisoning for the detection of sniffers in an Ethernet Network", www.docstoc.com/docs/38584195/arp-cache-poisoning-for-the-dete, 2010
[17] Vamshidhar Chillamcharla, "ARP Spoofing", www.scribd.com/doc/47803882/arp-spoofing,, 2011
[18] Cristina L.Abad, Rafael I.Bonilla, "An Analysis on the Schemes for Detecting and Preventing ARP cache poisoning attacks", 27th International Conference on Distributed Computing Systems Workshops (ICDCSW '07),0-7695-2838-4/07© 2007 IEEE
[19] CERT and CERT Coordination Center, "Overview of attack Trends",© Carnegie Mellon University, 2002, http://www.arcert.gov.ar/webs/textos/attack_trends.pdf.

S.Vidya completed M.Sc in Computer Science from SR College, Trichirappalli, TamilNadu, India in the year 1990 and M.Phil in Computer Science from Alagappa University, Karaikudi, TamilNadu, India in the year 2001. Working as Associate Professor in Computer Science in the Department of Computer Science, Fatima College, Madurai, TamilNadu, India since 1990. Areas of interest are Data Structures and Algorithms. She is currently pursuing research in Network Security. She is a Life Member of Computer Society of India and Member of ACM.

Dr.R.Bhaskaran completed M.Sc in Mathematics from IIT Chennai, TamilNadu, India in the year 1974 and got his Doctoral degree from Ramanujam Institute of Mathematical Sciences, Chennai, TamilNadu, India in the year 1979. Joined the School of Mathematics, Madurai Kamaraj University, Madurai in 1980. Currently he is the Chairperson of School of Mathematics. He has to his credit lots of publications including in IEEE conferences. His area of interest includes Linden Mayer System, Computer Applications, and developing software for learning Mathematics. He has guided students in both Mathematics and Computer Applications. He is a Life Member of the Indian Mathematical Society.