Detecting and Combating ARP Spoofing

Chai Ming Xuan
National University of Singapore
mingxuan@u.nus.edu

Er Xue Hui
National University of Singapore
xuehuier@u.nus.edu

Wu Wenqi
National University of Singapore
wenqi.wu@u.nus.edu

Zhu Chunqi
National University of Singapore
chunqi@u.nus.edu

ABSTRACT
Since the early days of the computer, the ARP cache has been an easy target for ARP attacks like ARP spoofing. Even until today, it is still relatively easy to carry out an ARP spoofing attack, and remains rather difficult to defend against it.

Thus, our project aims to create a program for users to actively detect if they are victims of ARP spoofing, and to offer alternative ways to protect themselves.

Categories and Subject Descriptors
C.2.0 [**Computer-Communication Networks**]: General - Data Communications, Security and Protection

; D.4.6 [**Security and Protection**]: Authentication, Verification

General Terms
Network Security, Intrustion Detection System (IDS), Address Resolution Protocol (ARP), spoofing

Keywords
ARP spoofing, network security, DHCP resolution, IDS, ARP protection

1. INTRODUCTION
The Address Resolution Protocol (ARP) is an important protocol in computer network communications.
Without this protocol, it would be difficult for computers to communicate with each other, because we would not know the MAC address of the computer we are communicating with. Thus, it would be difficult to establish an IP-to-MAC address mapping.

However, it is also one of the easier protocols to carry out attacks on, because of the lack of viable solutions to protect the ARP cache. Some common attacks that can occur after an ARP Spoof are man-in-the-middle (MITM), Denial-of-Service (DOS), or even session hijacking. In such cases, the security principles of Confidentiality, Integrity and Availability may be violated. We will be illustrating how an ARP Spoof occurs in Section 2.

Some common programs which users can use to carry out such an attack include 'Ettercap' and 'Cain and Abel'. As these programs are open source, almost anyone can easily carry out ARP Spoofing with some basic technical knowledge.

2. BACKGROUND
In order to understand how ARP spoofing occurs, we will need to cover the basics on how computers communicate with each other. In this section, we will be explaining how computers do so, and introduce the basic notions of how an ARP spoof is carried out.

2.1 The TCP/IP Model
In computer networking, the communication protocols are categorised and organised into different layers. In the TCP/IP Model, there are 5 layers:

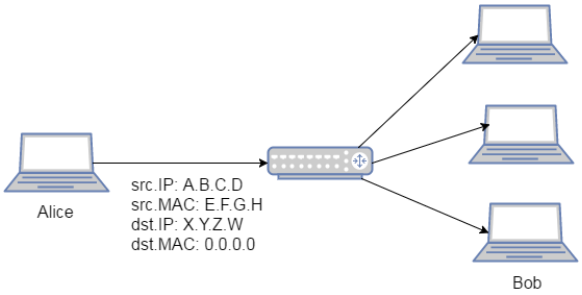
1. Application Layer
2. Transport Layer
3. Internet Layer
4. Link Layer
5. Physical Layer

DHCP is an application layer protocol and ARP is a link layer protocol.

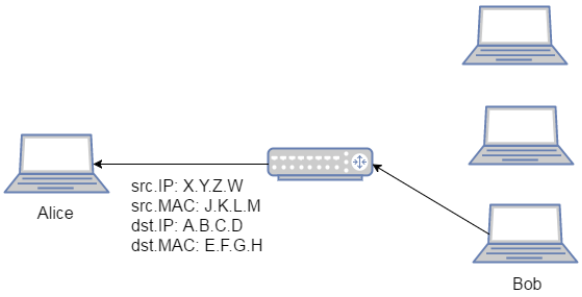
2.2 Computer Communications
In a typical modern computer, all IP addresses are resolved dynamically through the use of the Dynamic Host Configuration Protocol (DHCP). This is carried out in the following steps:
(to include picture)
1. DHCP Discover
2. DHCP Offer
3. DHCP Request
4. DHCP Ack

After that, assuming everyone has already had their IPs assigned to them via DHCP, suppose the the user Alice wishes to send some information to Bob. The following steps are then carried out:

1. ARP Request: Alice's computer sends out an ARP request to find out which MAC address has Bob's IP.



2. When Bob's computer receives the request, he sends back an ARP response packet.



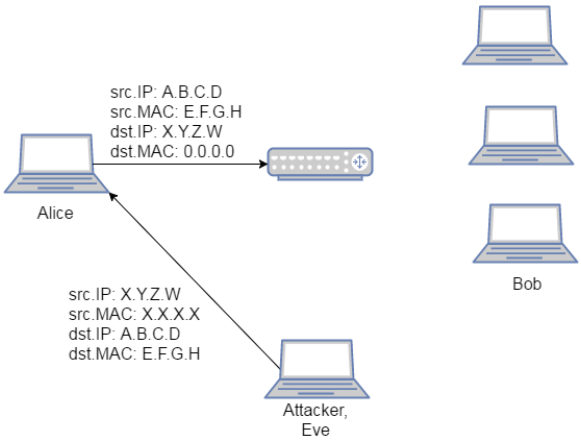
3. Alice gets the response and stores the corresponding IP-to-MAC entry into the ARP cache.

2.3 Poisoning the ARP Cache

If an attacker, Eve, wishes to carry out an MITM attack on the ARP, this is typically what happens:

1. Alice's computer sends out an ARP request to find out what is Bob's MAC address.

2. Before Bob's computer can reply, the attacker, Eve, sends a spam of packets to Alice's computer, claiming to be Bob. The ARP cache then becomes poisoned.



3. CURRENT SOLUTIONS AND MITIGATIONS

There are many solutions in the market to combat ARP Spoofing, such as Agnitum Outpost Firewall. However, many of these solutions are described to have major issues. From [8], [1] and [6], these are the summaries and drawbacks of current implementations in the market:

1. Use of Cryptographic Techniques
Examples: S-ARP

2. Passive Detection
Examples: Agnitum Outpost Firewall

3. Kernel-based patches
Examples: Anticap, Antidote

4. Making MAC entries static

4. GOALS

In taking up this project, we hoped to achieve the following goals:

1. Provide users with a means to actively combat ARP spoofing
2. Make any network more secure.
3. Provide a GUI for users to see what is happening in their network in real-time.

5. PROOF OF CONCEPT

5.1 Implementation System

Operating System:
OSX El Capitan, Arch-Linux [Manjaro 15.12 Capella]

Software:
Python 2.7 with the following libraries: scipy, scapy, matplotlib,
Ettercap
Wireshark

5.2 The Attacker

In our solution, we used Ettercap to stimulate an ARP Spoofing attack.

5.3 The Defender Module
(ming xuan's code)

5.4 The Spoof Detection Engine
(wenqi's code)

5.5 The GUI
to insert a picture once GUI is completed

6. OUR SOLUTION VERSUS CURRENT SOLUTIONS
to insert stuff

7. LIMITATIONS AND FUTURE WORK
Our project has several flaws which we were unable to resolve within a reasonable timeframe:

1. Our solution will not work on a network that employs WPA-Enterprise level of encryption. This is because the structure of WPA-Enterprise is such that only each user can see his incoming or outgoing network connections.

2. Our solution assumes that the user does not have any form of defence installed on his computer. (eg. no firewall that can prevent ARP spoofing, a network that does not use any enterprise level encryption, etc.)

3. The attacks have only been tested to work for various users across the wireless network. While it might work on a network that uses a hub, it might not be possible to find a user to spoof on a network that uses switches, unless a wiretap is attached to the victim computer's LAN port.

We hope to improve our solution for a more varied set of systems in the future.

8. CONCLUSION
Through this project, we have learnt that ARP Spoofing is not easy to defend against. Even though there are solutions on the market that have been out for a while, unless one is using a network encrypted by WPA-Enterprise, it is otherwise easy to fall prey to such attacks.

Furthermore, as active detection of ARP spoofing is not widely implemented yet, our solution may not be entirely foolproof. Thus, we will need to put in more work to make the solution more viable for a wider variety of situations.

(to modify conclusion and include more stuff)

9. ACKNOWLEDGEMENTS
First and foremost, we would like to thank Prof. Hugh Anderson for his guidance and patience with us throughout the semester. Our initial project was to investigate the hacking of Hello Barbie. However, we eventually decided to change the topic for various reasons, and he very kindly allowed us to do so. This is despite the topic being self-proposed and changed quite late in the semester.

Our thanks also goes out to him for loaning us the Hello Barbie even though we eventually dropped the project.

In addition, our appreciation also goes out to our friends who have loaned us various items for our project, such as a DLink Dir-825 router which runs on DD-RWT, so that we could work on the project in school.

10. REFERENCES
[1] N. Behboodian and S. A. Razak. Arp poisoning attack detection and protection in wlan via client web browser. pages 20–24, December 2011.

[2] G. Kaur and J. Malhotra. A review of latest techniques to secure wireless networks against arp poisoning attacks. *International Journal of the Future Generation Communication and Networking*, 8(2):225–232, 2015.

[3] V. Ramachandran and S. Nandi. Detecting arp spoofing: An active technique. pages 1–13.

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

10.1 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material is found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed dvi output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed dvi output of this document.

Table 2: Some Typical Commands

Command	A Number	Comments
`\alignauthor`	100	Author alignment
`\numberofauthors`	200	Author enumeration
`\table`	300	For tables
`\table*`	400	For wider tables

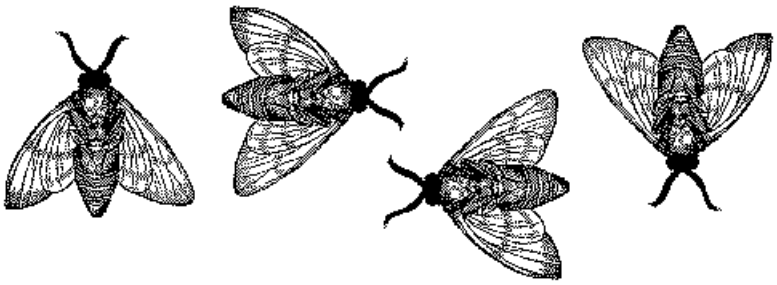


Figure 1: A sample black and white graphic (.eps format) that needs to span two columns of text.