COMP 8505 Assignment 4

DNS Spoof

Xinghua Wei & Tao Yuan

A00978597 & A00952007

Introduction

The purpose of this assignment is to become familiar with DNS spoofing and ARP poisoning. ARP spoofing is when the attacker sends out ARP messages from a non-authoritative server to change the Gateway, the DNS server of the victim. Once the victim's DNS server has been changed, the attacker can start the DNS spoofing attack that the victim machine will be redirected to the attacker's machine.

Design

DNS spoof will be done by using ARP spoofing. I will perform ARP spoofing by connecting to the authenticated IP address and receiving information sent to the victim machine. Anything the victim machine receives will first go through the attacker machine.

For DNS spoofing, I will establish a connection with the victim machine and the router and receive all packets sent to the victim. Instead of sending the packets that the victim machine requested, I will send updated packets that will lead the machine back to my choice. In this case, I will open an Apache server on the attacker machine and redirect requests to my Apache server website. If the user goes to some not cached website, I can receive the router's response and send my packet to the victim machine to redirect the victim machine to my website.

Diagram

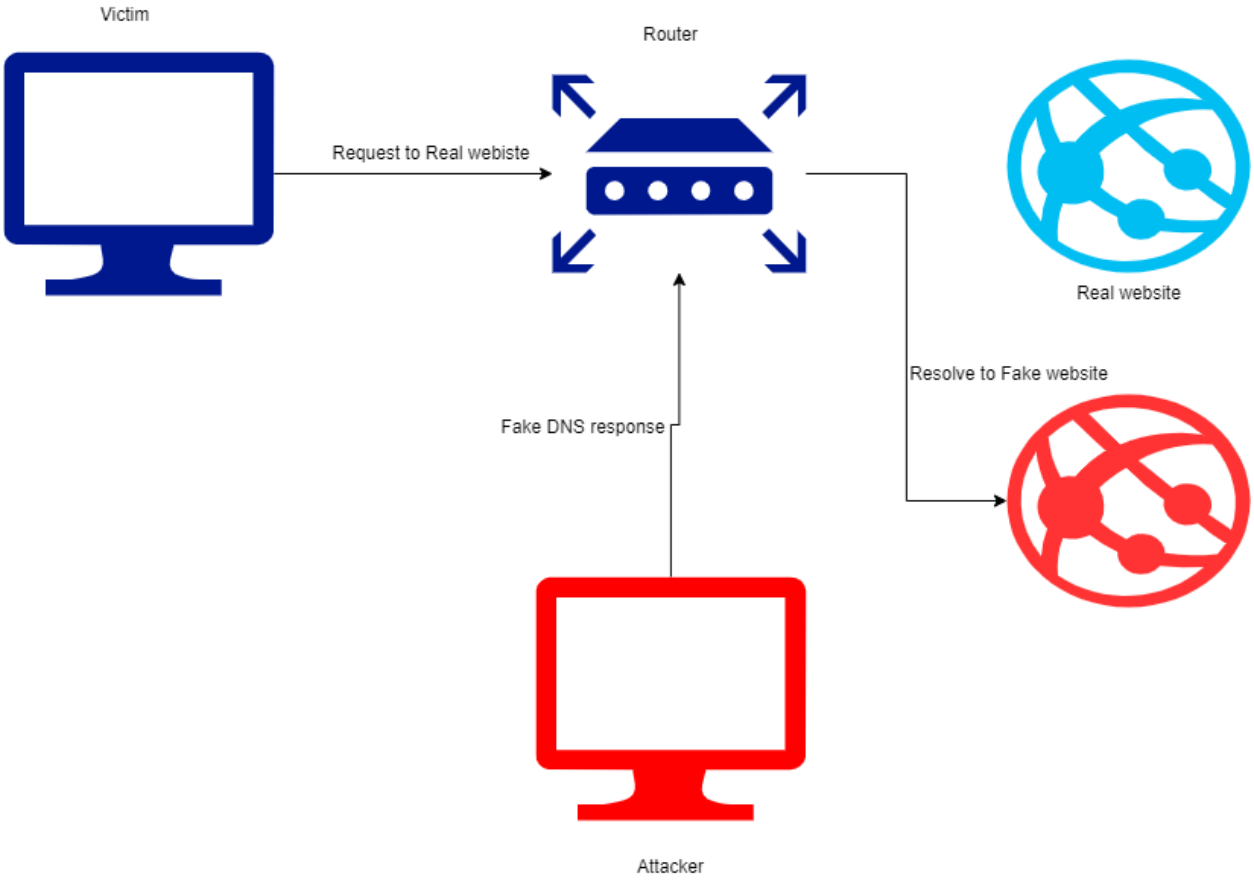


Figure 1 DNS spoof

Code Design

For this exercise, I need three files:

- ARP spoof
- DNS spoof
- IP addresses and Mac addresses file

Environment

The following tools will be used for this exercise:

- Python 3.8
- Scapy
- Multiprocessing

IP addresses and Mac addresses file

In this file, it will contain all IP addresses and Mac addresses used for this assignment. Each line will be read by orders.

Attacker machine Mac address	**xx:b4:d2:2b:ed:xx**
Attacker machine IP address	10.0.0.33
Victim machine Mac address	xx:5e:45:cf:38:xx
Victim IP address	10.0.0.8
Router Mac address	xx:56:11:8f:e7:xx
Router IP address	10.0.0.1

ARP spoof

ARP spoofing takes all the requests and responses from the DNS spoof and creates a packet based on the information past through. This includes the IP and Mac address of the router and the victim machine.

ARP spoof will craft two ARP headers for the victim machine's packet and the router's ARP packet:

- ARP header for victim machine:
    - Hwsrc: Attacker machine Mac address
    - Hwdst: Victim machine Mac address
    - Psrc: Router IP address
    - Pdst: Victim IP address
    - Op: 2
- ARP header for the router:
    - Hwsrc: Attacker machine Mac address
    - Hwdst: Victim machine Mac address
    - Psrc: Victim IP address
    - Pdst: Router IP address
    - Op: 2

DNS spoof

DNS spoof is to read and intercept the packets and craft the packets, sending packets back to the router and the victim machine. Then the packets sent to the victim machine will redirect the victim machine to my Apache server website.

- Configuration():
  - This function is to read and extract information in IP address and Mac address file:
  - The victim machine IP, Mac address
  - The router IP, Mac address
  - The attacker machine IP, Mac address
- ReadPacket(packet):
  - This function is to take the packet that was read and parse it and then generate a spoofed DNS response.
  - If the source IP is the victim machine IP
  - packetResponse = (Ether()/IP(dst=packet[IP].src, src=packet[IP].dst)/\
    UDP(dport=packet[UDP].sport, sport=packet[UDP].dport)/\
    DNS(id=packet[DNS].id, qd=packet[DNS].qd, aa=1, qr=1, \
    an=DNSRR(rrname=packet[DNS].qd.qname, ttl=10, rdata=attackIP)))
- spoof():
  - This function is to call ARP spoof function in ARP spoof file using multiprocessing. Then it will sniff packets and call ReadPacket function to listen for incoming packets and receiving packets to parse.
  - Sniff filter: UDP and port 53
- Main()
  - This function is to start the program and ensure the forwarding in the attack machine is enabled. And set the firewall rules which will drop all forwarding packets to the victim machine to ensure the speed of the response would not affect spoofing.
  - Enable forwarding by put 1 into ip_forward file
  - Firwall rule to drop forwarding packets:
    - iptables -A FORWARD -p UDP --sport 53 -d 10.0.0.8 -j DROP

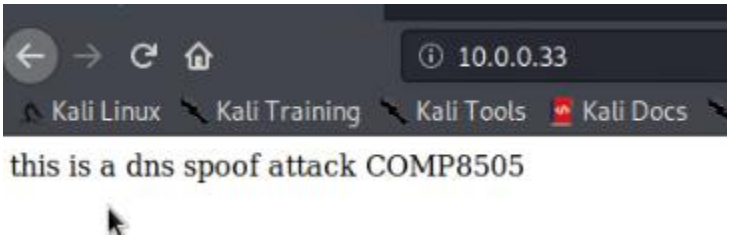
Demo

Run Apache server and exam it

Command to start Apache2 server:

```
# service apache2 start
```

Then exam the server by typing localhost as URL:



Start DNS spoofing

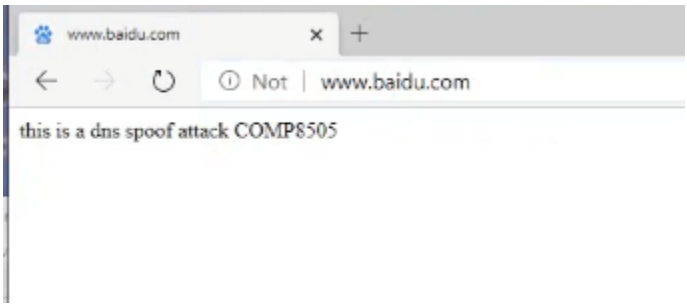
Then I start dnsAttack.py to perform DNS spoofing attack and it will show the redirect source IP and destination IP.

```
root@kali:/home/kali/Desktop/8505# python dnsAttack.py
f0:b4:d2:2b:ed:a7 10:56:11:8f:e7:c9
DNS spoof start
10.0.0.8 64.59.144.92
Redirecting
10.0.0.8 64.59.144.92
Redirecting
10.0.0.8 64.59.150.138
Redirecting
10.0.0.8 64.59.150.138
Redirecting
```

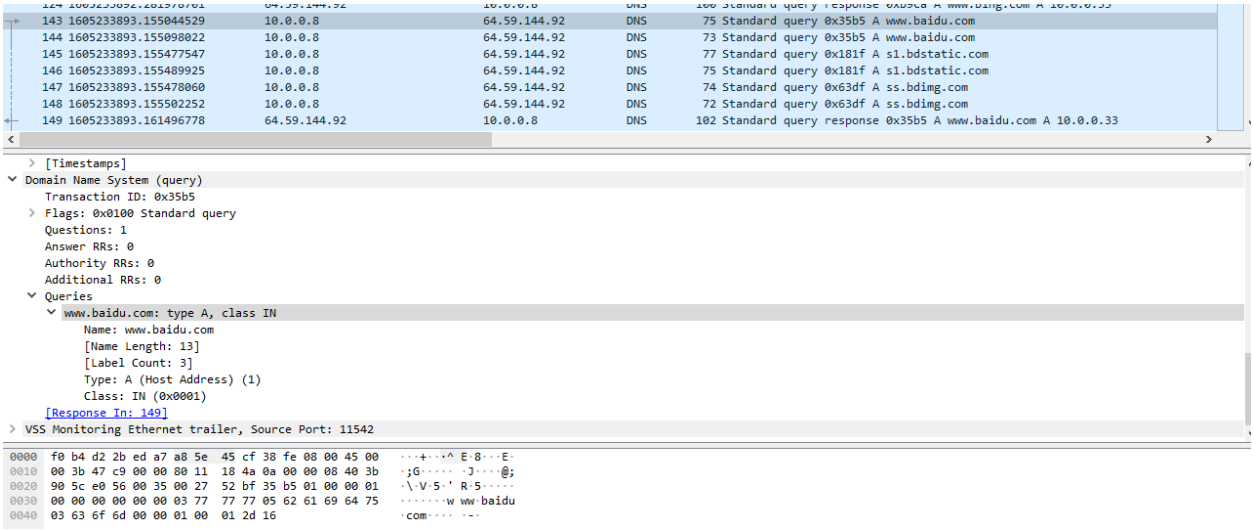
Checking the victim machine

I use Edge for this exercise. Then I open a website, and initially, it looks like:

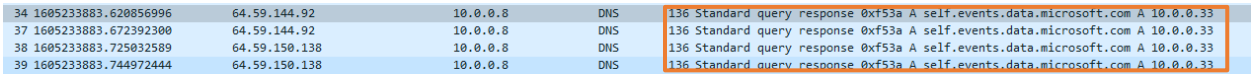
After being attacked by DNS spoofing attack, it looks like:



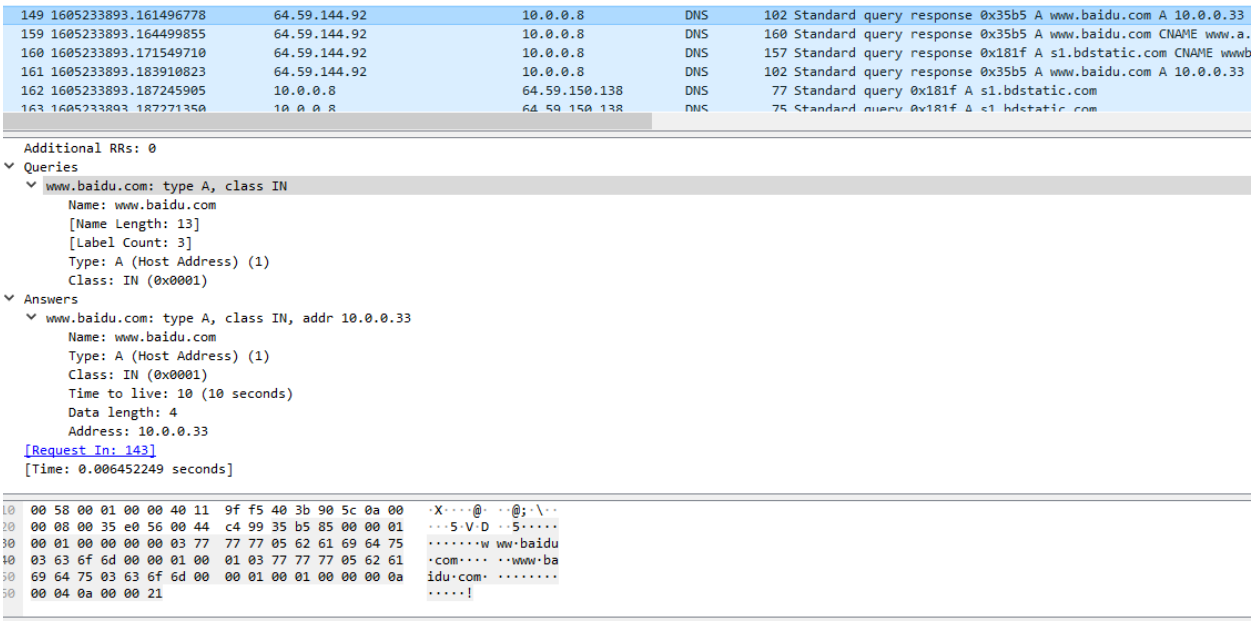
Wireshark



- Name: www.baidu.com
- Type: Set to A, which indicates the host IP address is provided
- Class: Set to IN (0x0001)



This shows redirecting the victim machine packets work.



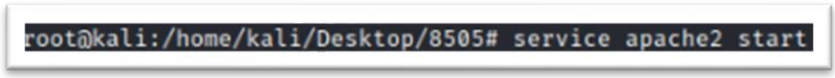
- Time to Live: 10 seconds as configurated in the packets
- Answer address: 10.0.0.33, the attacker IP address
- Type: A

Test Cases

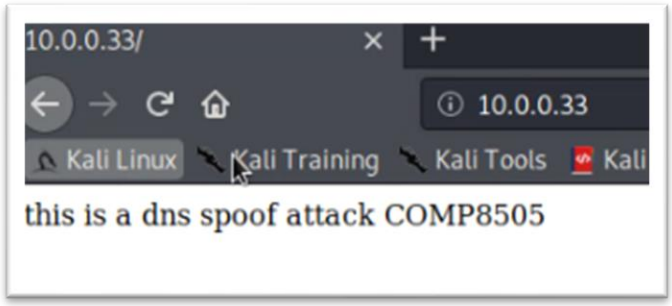
Test Number	Description	Expect Result	Result
1	Attack machine will start an Apache2 server website	Go to website 10.0.0.33 and see modified index.html	Pass
2	The victim goes to www.baidu.com	The victim is spoofed to my Apache2 website	Pass
3	The victim goes to www.haha.com	The victim is spoofed to my Apache2 website	Pass
4	The victim goes to www.qq.com	The victim is spoofed to my Apache2 website	Pass
5	DNS spoof program should automatically set up a firewall rule to DROP all forwarding packets	Use iptables command. There should be dropped packets in the iptables	Pass

Tests

Test Case 1



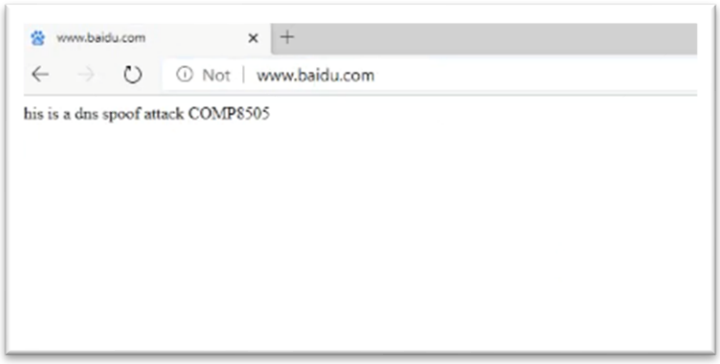
Start the Apache2 server.



The Apache2 server website performs as expected.

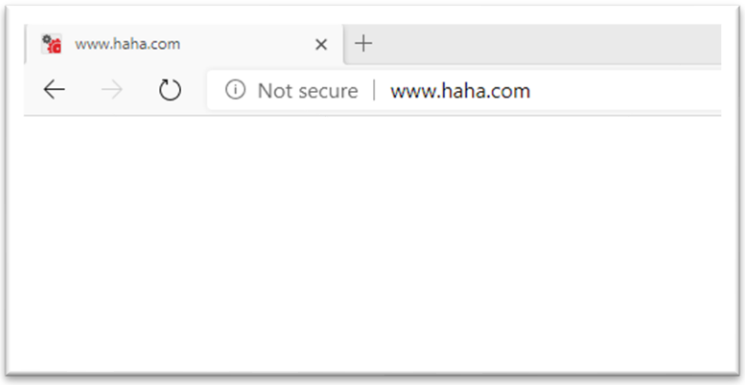
Test Case 2

This is the original www.baidu.com.

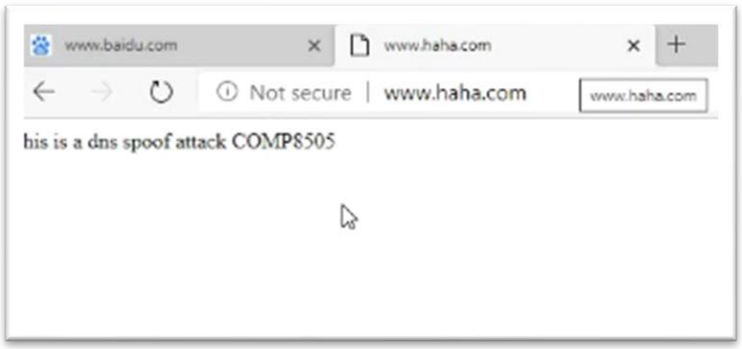


This is the same website after attacked by the DNS spoofing attack. DNS spoofing performs as expected.

Test Case 3



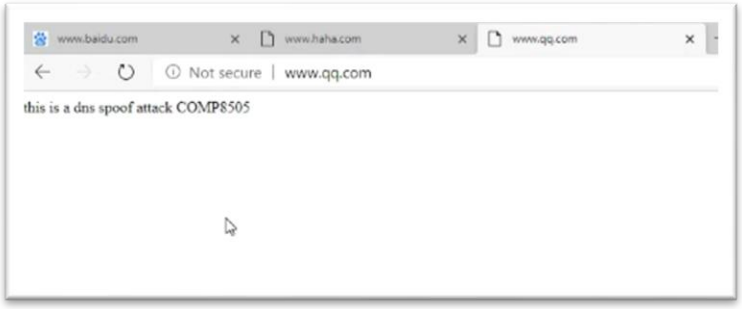
This is the original www.haha.com. It is an empty website.



This is the same website after attacked by the DNS spoofing attack. DNS spoofing performs as expected.

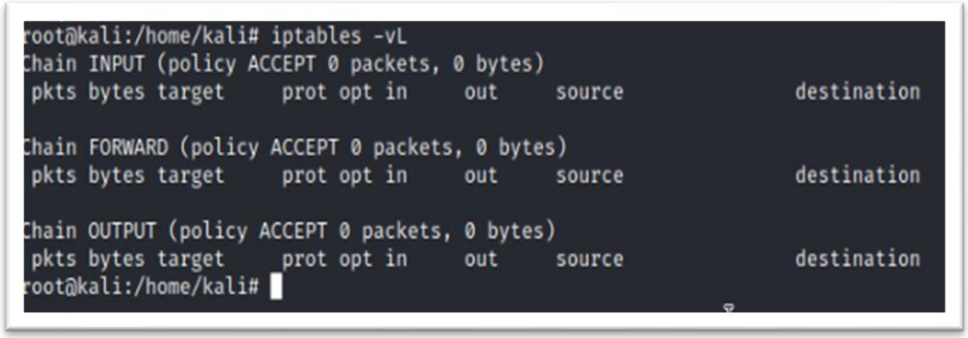
Test Case 4

This is the original www.qq.com. It is the index webpage for Tencent.

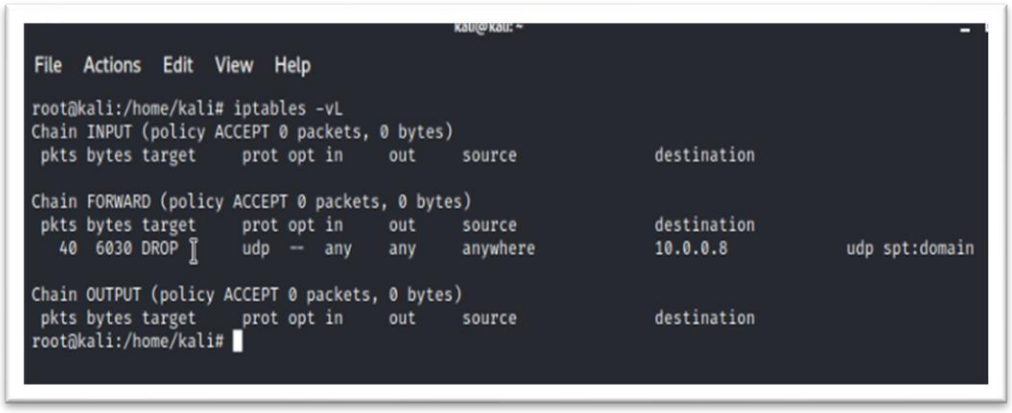


This is the same website after attacked by the DNS spoofing attack. DNS spoofing performs as expected.

Test Case 5



This is the firewall setting on the attacker machine. It has no rules set, and no packets are dropped.



This is the firewall rules after the DNS spoofing program is executed. The forwarding rule is modified that all UDP packets forwarding to 10.0.0.8 (the victim machine IP address) are dropped. It shows that 40 packets and 6030 bytes are dropped.

The DNS spoofing program setting the firewall rules performs as expected.