2018

DNS Spoofing Application

TESTING

DANNY LIEU

Table of Contents

Test outline

Test #	Description	Tools used	Expected results	Actual results	Status
1	Application parse all user input from command-line arguments.	Python	Get all input from user and print out.	Victim IP, router IP and redirected IP are printed out.	Passed. See below for more details.
2	ARP poisoning the victim machine	Python, Wireshark	The victim machine sees the attacker machine as the router	Running Wireshark, the victim machine receives ARP packet indicating the attacker machine is router	Passed. See below for more details.
3	DNS Spoofing a normal website such as ipecho.net	Python, Firefox	The victim will be lead to the fake website running on the attacker's Apache server.	The victim goes to the fake website.	Passed. See below for more details.
4	DNS Spoofing a high security website such as facebook.com	Python, Firefox	The victim will be lead to the fake website running on the attacker's Apache server.	Firefox shows the error indicating SSL is SSL_ERROR_RX_RECORD_TOO_LONG	Failed. See below for more details

Test description

Test #1

Application parse all user input from command-line arguments.

```
14:52:57(master)root@datacomm-16:Listings$ python DNSSpoofing.py -v 192.168.0.15
 -r 192.168.0.100 -re 192.168.0.17
```

- IP address of the victim is 192.168.0.15.
- IP address of the router is 192.168.0.100
- IP address of the machine that the victim will be redirected to is 192.168.0.17.

If the application run successfully, it will print out the MAC address of the victim and the router.

This is the result after pressing Enter:

```
14:52:57(master)root@datacomm-16:Listings$ python DNSSpoofing.py -v 192.168.0.15
 -r 192.168.0.100 -re 192.168.0.17
Victim IP: 192.168.0.15 Victim MAC: 98:90:96:d4:af:4f
Router IP: 192.168.0.100 Router MAC: 44:d9:e7:95:e4:9f
Forwarding DNS requests to: 192.168.0.17
Starting ARP poisoning to 192.168.0.15
```

The MAC address of the victim and the router are successfully printed out.

Test #2

ARP poisoning the victim machine

First of all, we will check what the actual router MAC address by running command `arping`:

```
15:00:20(-)root@datacomm-15:~$ arping -I eno1 192.168.0.100
ARPING 192.168.0.100 from 192.168.0.15 eno1
Unicast reply from 192.168.0.100 [44:D9:E7:95:E4:9F]  0.665ms
Unicast reply from 192.168.0.100 [44:D9:E7:95:E4:9F]  0.611ms
Unicast reply from 192.168.0.100 [44:D9:E7:95:E4:9F]  0.640ms
^CSent 3 probes (1 broadcast(s))
Received 3 response(s)
```

The MAC address of the router is `44:D9:E7:95:E4:9F`. After that, we run our application and confirm if we successfully poison the router.

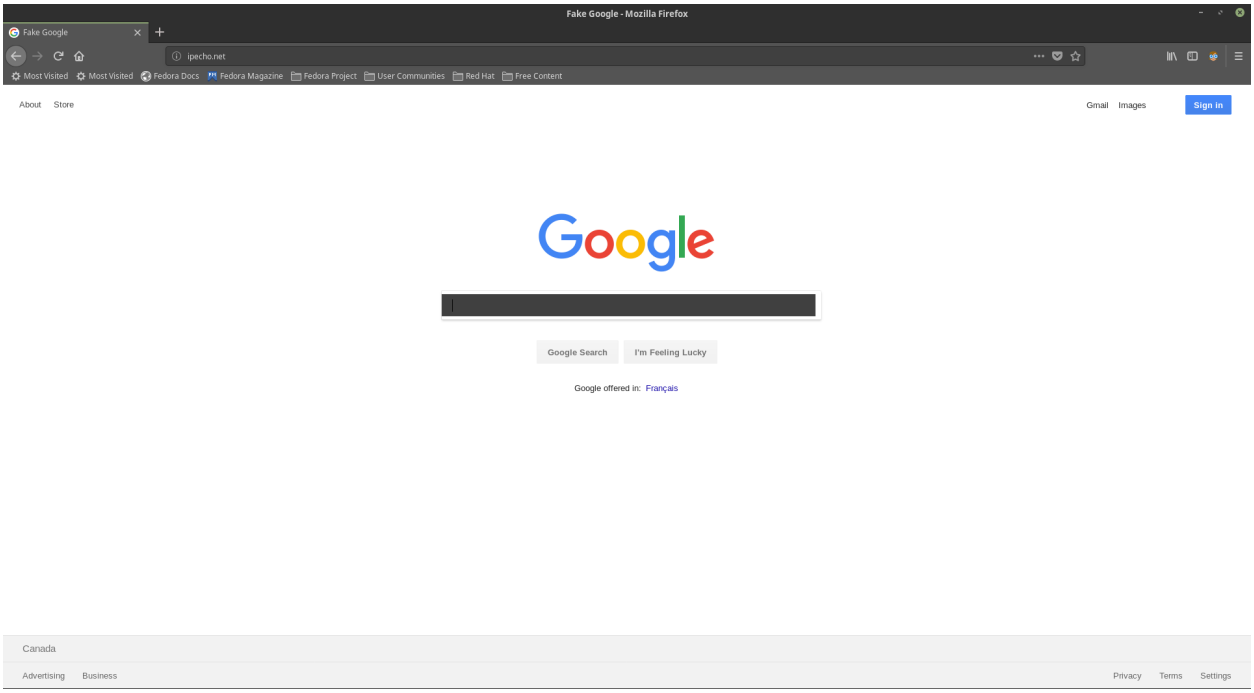
```
124 Standard query response 0x8d9e AAAA dete
 60 RST. Root = 32768/1/b4:14:89:a8:2c:80
 60 192.168.0.100 is at 98:90:96:d0:35:af
```

Now the MAC address of the router is `98:90:96:D0:35:AF` which is the MAC address of the attacker machine.

Test #3

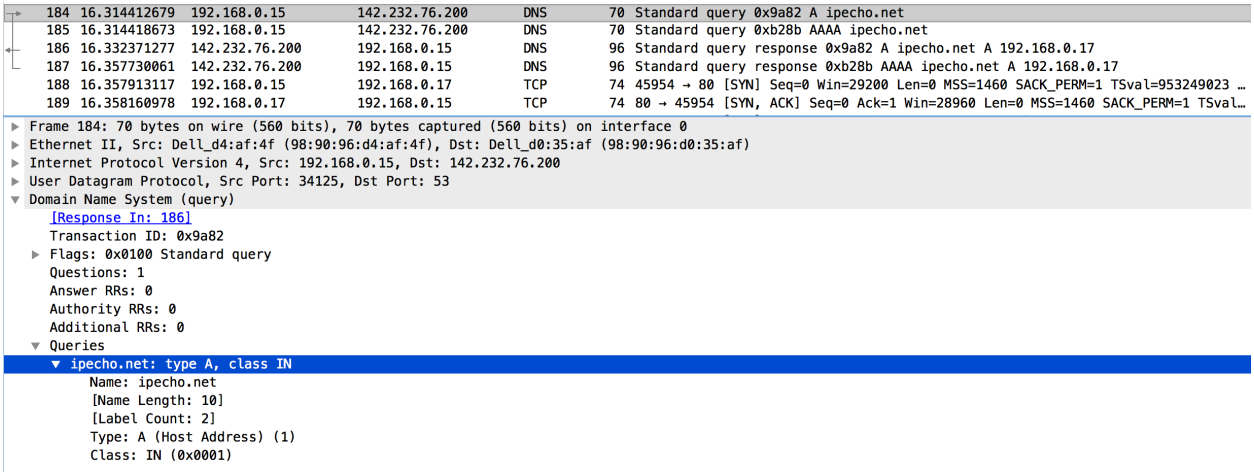
DNS Spoofing a normal website such as ipecho.net

The website that is going to be tested is http://ipecho.net. This website does not have SSL.

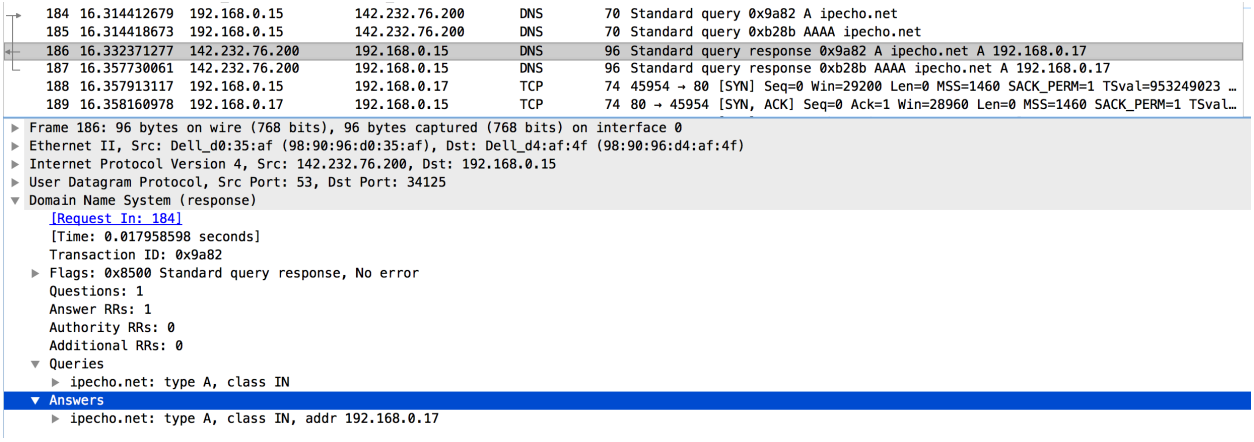


As we can see, the URL field is ipecho.net but the page is actually a fake Google hosted on the redirected machine using Apache. The title of the page is also Fake Google.

In Wireshark, the victim machine tried to send DNS request to the DNS server which is poisoned:



Because the router is poisoned and all DNS traffic to send to the attacker machine. Then, the attacker machine redirects it to the redirected IP.

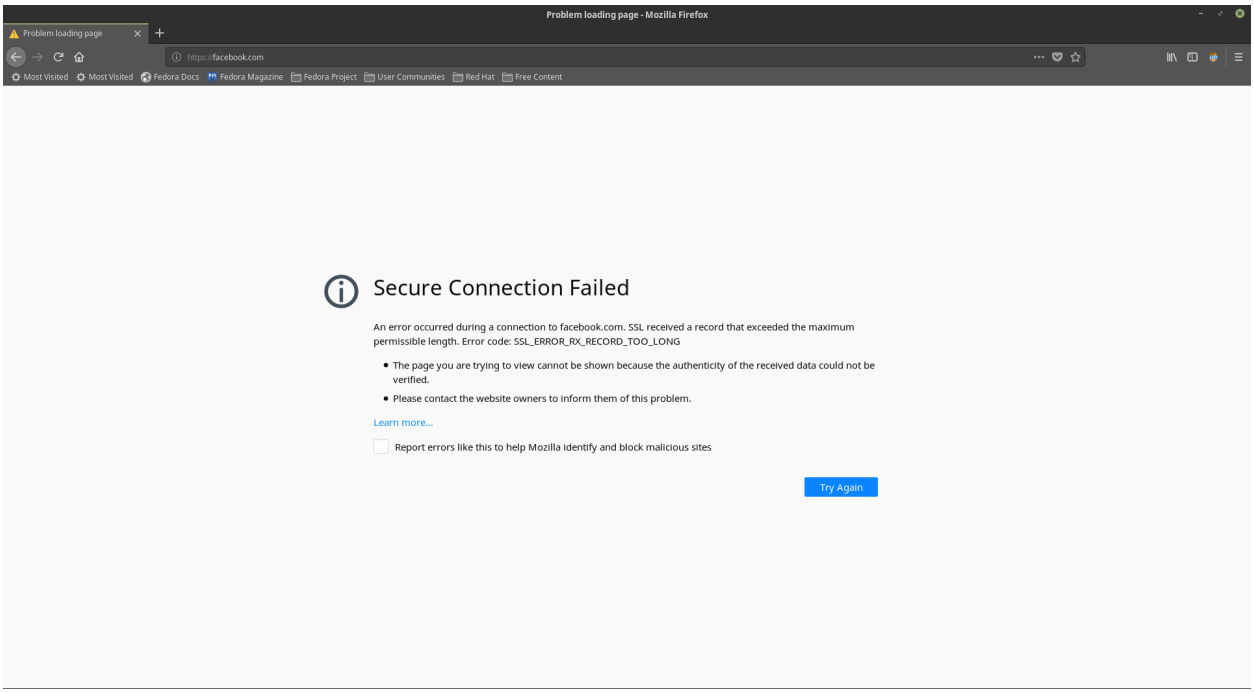


When looking at the DNS response packet, we can see that the answer is the IP address of the redirected machine. We successfully spoof DNS traffic of the victim machine.

Test #4

DNS Spoofing a high security website such as facebook.com

We are going to do the same test, but it will be Facebook website. This is the result of the test:



The browser show an error saying that SSL_ERROR_RX_RECORD_TOO_LONG. It shows it because Facebook has SSL layer, so we cannot spoof it.

The solution for this problem is implementing SSL stripping which will strip off SSL from the URL and then we can perform our DNS spoofing.