DNS - Spoofer

COMP 8505 – Assignment 4

Submitted by
Filip Gutica (A00781910)
Gary Khoo (A00564204)

Table of Contents

Introduction

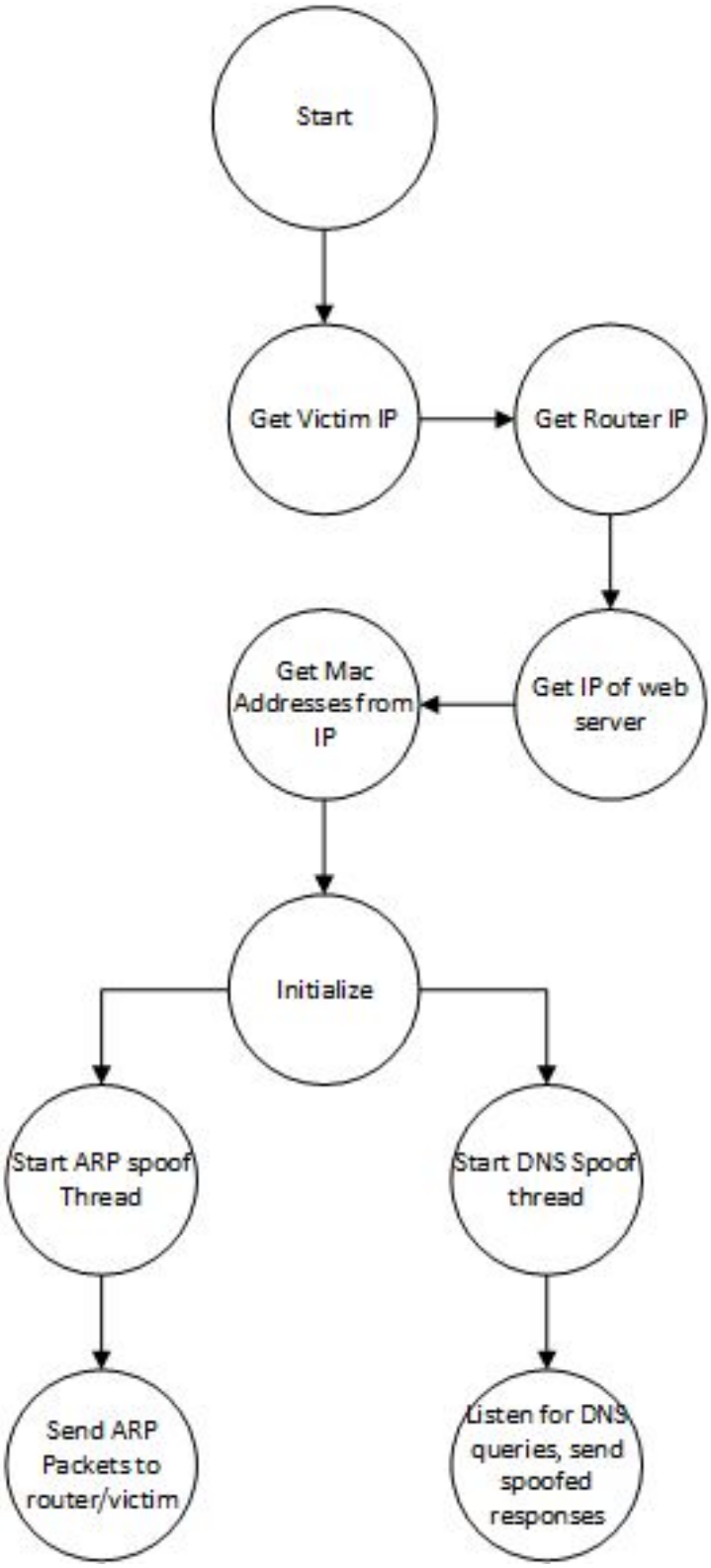
Manipulation of DNS traffic is a very dangerous attack. Knowing how it works, and how to write code to perform this spoofing is very important in understanding how to protect from it. For our assignment we have implemented a very simple ARP and DNS spoofer in python that will send spoofed DNS responses to a victim, redirecting all of their web requests to an ip address you specify.

Design

Our application has these main components:

1. Start → Get user input of ip addresses needed for spoofing
2. We need to enable IP forwarding, and add an iptables rule as to not send back the legit DNS responses to our victim.
3. Initialize → here is where we craft our ARP packets then start our threads
4. ARP Thread → This thread sends out the spoofed arp packets to the router and victim
5. DNS Thread → Here we sniff for incoming DNS requests, and send back spoofed responses redirecting the victim to our spoofing webservice

Design - Diagram



Testing

The following requirements were given for a successful DNS spoof implementation:

- Your application will simply sense an HTML DNS Query and respond with a crafted Response answer, which will direct the target system to a your own web site.
- You will test this POC on a LAN on your own systems only. This means that you are not to carry out any DNS spoofing activity on unsuspecting client systems.
- You are required to handle any arbitrary domain name string and craft a spoofed Response.

Based on the requirements above, we came up with the test cases below to test the application against. Our results and discussion of each test case are presented in the following sections.

#	Scenario	Tools Used	Expected Behavior	Actual Behavior	Status
1	Sense HTML DNS Queries	Wireshark, Scapy, Python	Victim's DNS Queries appear on attacker's machine	Victim's DNS Queries appear on attacker's machine	Pass
2	Send back spoof DNS responses	Python, Scapy, Wireshark	DNS Responses appear on both attacker and target machines as DNS responses	DNS Responses appear on both attacker and target machines as DNS responses	**Pass**
3	Victim is redirected to our web service	Node.js Chrome	User is redirected to our "You have been spoofed site" when they try to navigate to any A record URL	User is redirected to our "You have been spoofed site" when they try to navigate to any A record URL	**Pass**
4	Handle any arbitrary domain name string and craft a spoofed Response.	Python, Scapy, Wireshark	We send spoofed packets on any DNS request from the victim	We send spoofed packets on any DNS request from the victim	**Pass**

An example of how we started our application:

```
Enter Victim IP: 192.168.2.129
Enter Router IP: 192.168.2.1
Enter your webserver IP: 192.168.2.50
```

Test 1 Sense HTML DNS Queries

DNS Queries as they appear on the attackers machine.

645	112.3195350(	192.168.2.129	192.168.2.1	DNS	134	Standard query response 0x657c A 192.168.2.50
646	112.3444410(	192.168.2.1	192.168.2.129	DNS	134	Standard query response 0x7801 A 192.168.2.50
647	112.3815710(	192.168.2.1	192.168.2.129	DNS	134	Standard query response 0x7801 A 192.168.2.50
648	112.4165480(	192.168.2.129	192.168.2.1	DNS	134	Standard query response 0x7801 A 192.168.2.50
653	115.0996630(	192.168.2.129	192.168.2.1	DNS	77	Standard query 0xcb31 A milliways.bcit.ca
654	115.0997590(	192.168.2.129	192.168.2.1	DNS	77	Standard query 0xcb31 A milliways.bcit.ca
655	115.1163890(	192.168.2.1	192.168.2.129	DNS	110	Standard query response 0xcb31 A 192.168.2.50
656	115.1258050(	192.168.2.1	192.168.2.129	DNS	129	Standard query response 0xcb31 CNAME milliways.scas.bcit.ca A 142
657	115.1466760(	192.168.2.1	192.168.2.129	DNS	110	Standard query response 0xcb31 A 192.168.2.50

```
▾ Domain Name System (query)
    Transaction ID: 0xcb31
  ▾ Flags: 0x0100 Standard query
      0... .... .... .... = Response: Message is a query
      .000 0... .... .... = Opcode: Standard query (0)
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... .0.. .... = Z: reserved (0)
      .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ▾ Queries
    ▸ milliways.bcit.ca: type A, class IN
```

DNS Queries as they appear on our spoofer application.

```
###[ IP ]###
     version   = 4L
     ihl       = 5L
     tos       = 0x0
     len       = 63
     id        = 29574
     flags     = DF
     frag      = 0L
     ttl       = 63
     proto     = udp
     chksum    = 0x4255
     src       = 192.168.2.129
     dst       = 192.168.2.1
     \options   \
###[ UDP ]###
        sport     = 33320
        dport     = domain
        len       = 43
        chksum    = 0x38a9
###[ DNS ]###
           id        = 37238
           qr        = 0L
           opcode    = QUERY
           aa        = 0L
           tc        = 0L
           rd        = 1L
           ra        = 0L
           z         = 0L
           rcode     = ok
           qdcount   = 1
           ancount   = 0
           nscount   = 0
           arcount   = 0
           \qd        \
            |###[ DNS Question Record ]###
            |  qname     = 'milliways.bcit.ca.'
            |  qtype     = AAAA
            |  qclass    = IN
           an        = None
           ns        = None
           ar        = None
Spoofed DNS Response Sent
```

Test 2 Send back spoof DNS responses

Here is a response that our spoofer has sent, as seen by the attacker's machine.

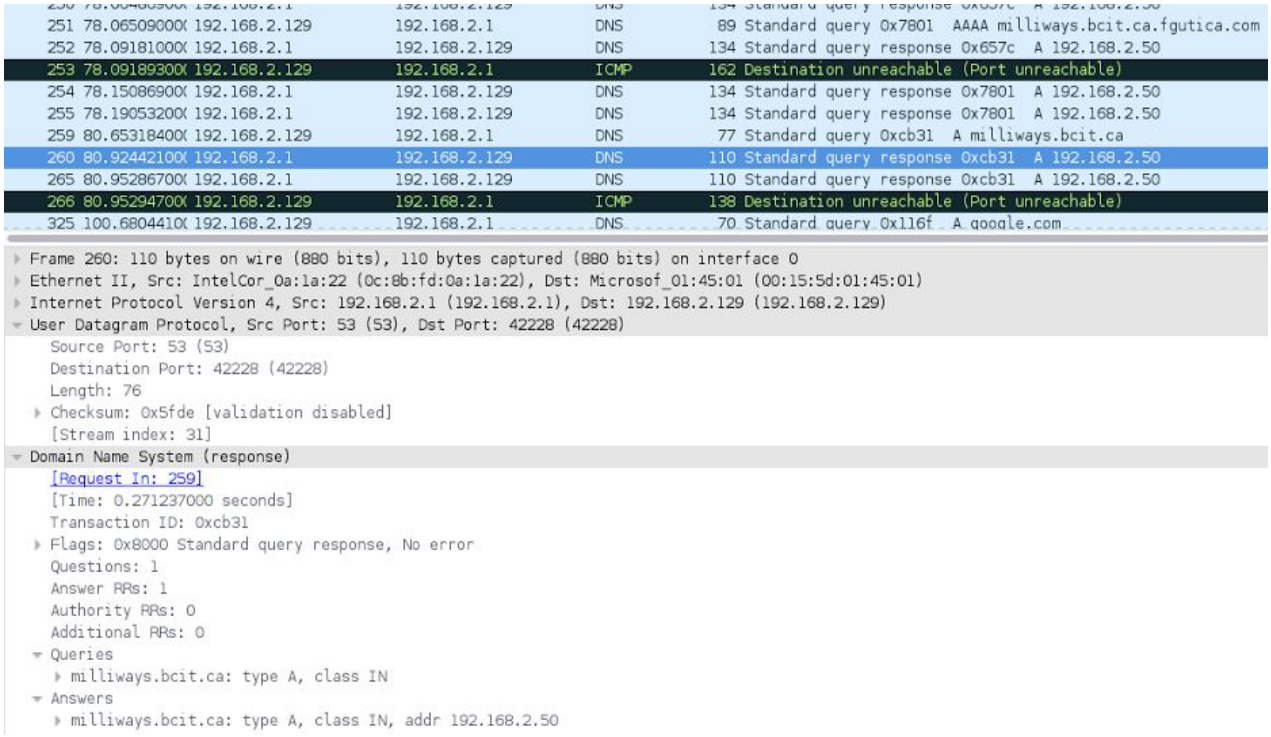
```
    645 112.3195350( 192.168.2.129          192.168.2.1          DNS        134 Standard query response 0x657c  A 192.168.2.50
    646 112.3444410( 192.168.2.1            192.168.2.129        DNS        134 Standard query response 0x7801  A 192.168.2.50
    647 112.3815710( 192.168.2.1            192.168.2.129        DNS        134 Standard query response 0x7801  A 192.168.2.50
    648 112.4165480( 192.168.2.129          192.168.2.1          DNS        134 Standard query response 0x7801  A 192.168.2.50
    653 115.0996630( 192.168.2.129          192.168.2.1          DNS         77 Standard query 0xcb31  A milliways.bcit.ca
    654 115.0997590( 192.168.2.129          192.168.2.1          DNS         77 Standard query 0xcb31  A milliways.bcit.ca
    655 115.1163890( 192.168.2.1            192.168.2.129        DNS        110 Standard query response 0xcb31  A 192.168.2.50
    656 115.1258050( 192.168.2.1            192.168.2.129        DNS        129 Standard query response 0xcb31  CNAME milliways.scas.bcit.ca A 14
```

```
    [Request In: 654]
    [Time: 0.016630000 seconds]
    Transaction ID: 0xcb31
  ▾ Flags: 0x8000 Standard query response, No error
      1... .... .... .... = Response: Message is a response
      .000 0... .... .... = Opcode: Standard query (0)
      .... .0.. .... .... = Authoritative: Server is not an authority for domain
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...0 .... .... = Recursion desired: Don't do query recursively
      .... .... 0... .... = Recursion available: Server can't do recursive queries
      .... .... .0.. .... = Z: reserved (0)
      .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
      .... .... ...0 .... = Non-authenticated data: Unacceptable
      .... .... .... 0000 = Reply code: No error (0)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  ▾ Queries
    ▸ milliways.bcit.ca: type A, class IN
  ▾ Answers
    ▸ milliways.bcit.ca: type A, class IN, addr 192.168.2.50
0000  00 15 5d 01 45 01 0c 8b  fd 0a 1a 22 08 00 45 00   ..].E... ..."..E.
0010  00 60 00 01 00 00 40 11  f4 b9 c0 a8 02 01 c0 a8   .`....@. ........
0020  02 81 00 35 a4 f4 00 4c  5f de cb 31 80 00 00 01   ...5...L _..1....
0030  00 01 00 00 00 00 09 6d  69 6c 6c 69 77 61 79 73   .......m illiways
0040  04 62 63 69 74 02 63 61  00 00 01 00 01 09 6d 69   .bcit.ca ......mi
0050  6c 6c 69 77 61 79 73 04  62 63 69 74 02 63 61 00   lliways. bcit.ca.
```

File: "/var/tmp/wireshark_pcapn... Packets: 1170 · Displayed: 237 (20.3%) · Dropped: 7 (0.6%)

An answer as seen by the victim machine:

As you can see the spoofed response is for milliways.bcit.ca but the address is shown as 192.168.2.50

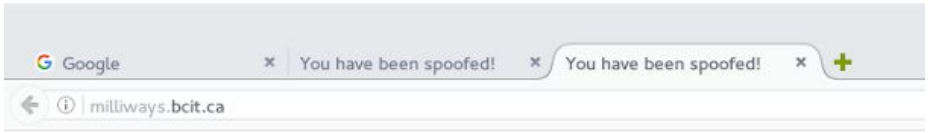


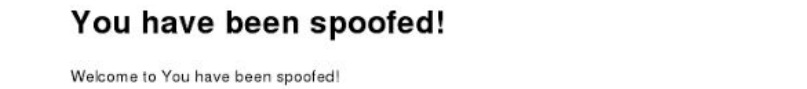
Nslookups as seen by the victim machine:
Here we demonstrate how all nslookups turn up as our spoofed address of 192.168.2.50 where our web server is running.

```
[root@localhost ~]# nslookup milliways.bcit.ca
Server:         192.168.2.1
Address:        192.168.2.1#53

Non-authoritative answer:
Name:   milliways.bcit.ca
Address: 192.168.2.50

[root@localhost ~]# nslookup google.com
Server:         192.168.2.1
Address:        192.168.2.1#53

Non-authoritative answer:
Name:   google.com
Address: 192.168.2.50

[root@localhost ~]# nslookup bcit.ca
Server:         192.168.2.1
Address:        192.168.2.1#53

Non-authoritative answer:
Name:   bcit.ca
Address: 192.168.2.50
```

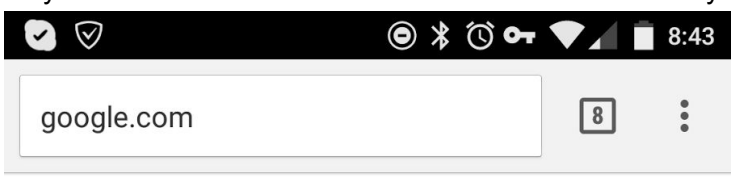
Test 3 Victim is redirected to our web service

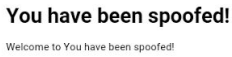
When victim navigates to any web page, they will be redirected to our “you have been spoofed webpage”





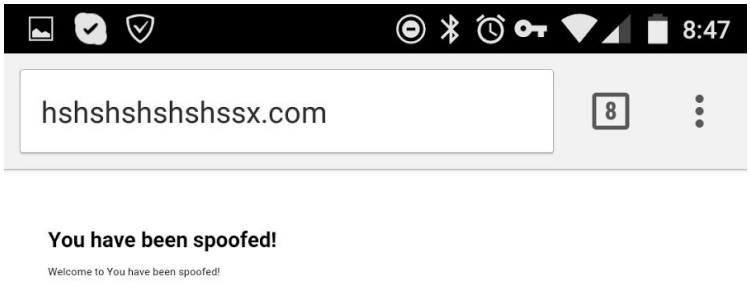
As you can see even mobile devices can be affected by this spoof:





Test 4 Handle any arbitrary domain name string and craft a spoofed Response

Any arbitrary request will be redirected:



Conclusion

After doing this assignment we have realised how easy it is to perform these type of man in the middle attacks on unsuspecting networks. It really demonstrates the necessity to protect ourselves and our information when we are navigating the web on any network.

By writing this application we better understand the ARP and DNS protocols and will be much more effective as security admins out in the field.