

=> Implemented in Python and Library used Scapy

=> **How to compile:**

```
sudo python dnsinject.py  
sudo python dnsdetect.py
```

General Design

=> dnsinject:

In main function command line arguments are parsed and bpf filter along with interface is given to the sniffer function. This function sniffs for packets.

The callback function dns_spoof takes in a packet. It first sets the ip address of spoofer to 172.16.1.63 (my local machines IP). It checks if the packet has DNS layer and it is a query. Then a check is included to see if hostnames have been specified for which spoofing should be done. If yes and the hostname is not in the list then return. Else, move on and make a spoofed packet and send it. In the DNS layer of the packet set the IP to the ip address of spoofer.

=> dnsdetect:

In main function the command line arguments are parsed. If tracefile is not specified it sniffs on the interface else on the tracefile.

To detect spoofed packets a dictionary is maintained, seenResponses. In the callback function dns_detect if a query is seen then the DNS id is used as a key and its value is set to None. If a response is seen and if the value is None then for that DNS id the value is set to a list [packet_payload, dns_rdata (IP address)]. If a retransmission has occurred then in the value the ip address is appended. Otherwise a DNS poisoning attempt is detected and the output is printed. It contains the as Answer 1 the IP address of the packet that came in late and a list of IP address of the packets that came before in Answer 2. It is a list because in case retransmission occurred all IP addresses of the packets are printed.

False Positives:

If the payload of the the new packet is same as the packet stored in the seenResponses dict data structure (key is the DNS id) then it's a retransmission.

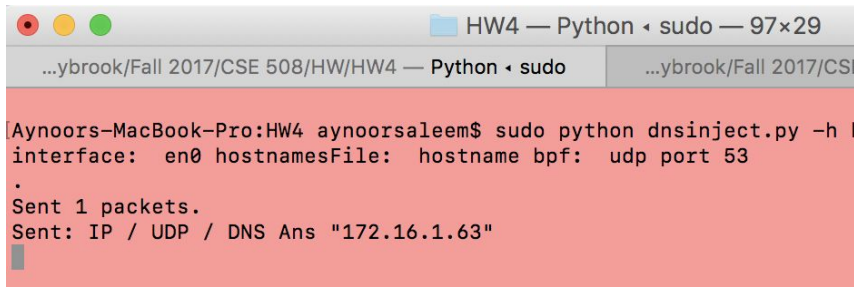
OS Version:

macOS Sierra version 10.12.6

Working example:

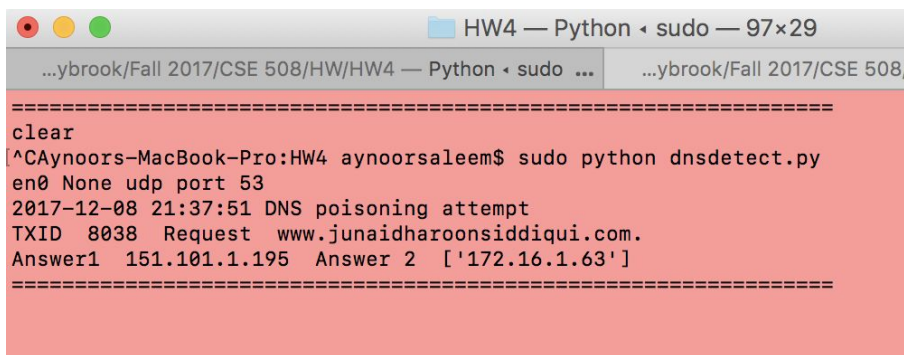
Example from tracefile submission.pcap:

For command: dig www.junaidharoonsiddiqui.com

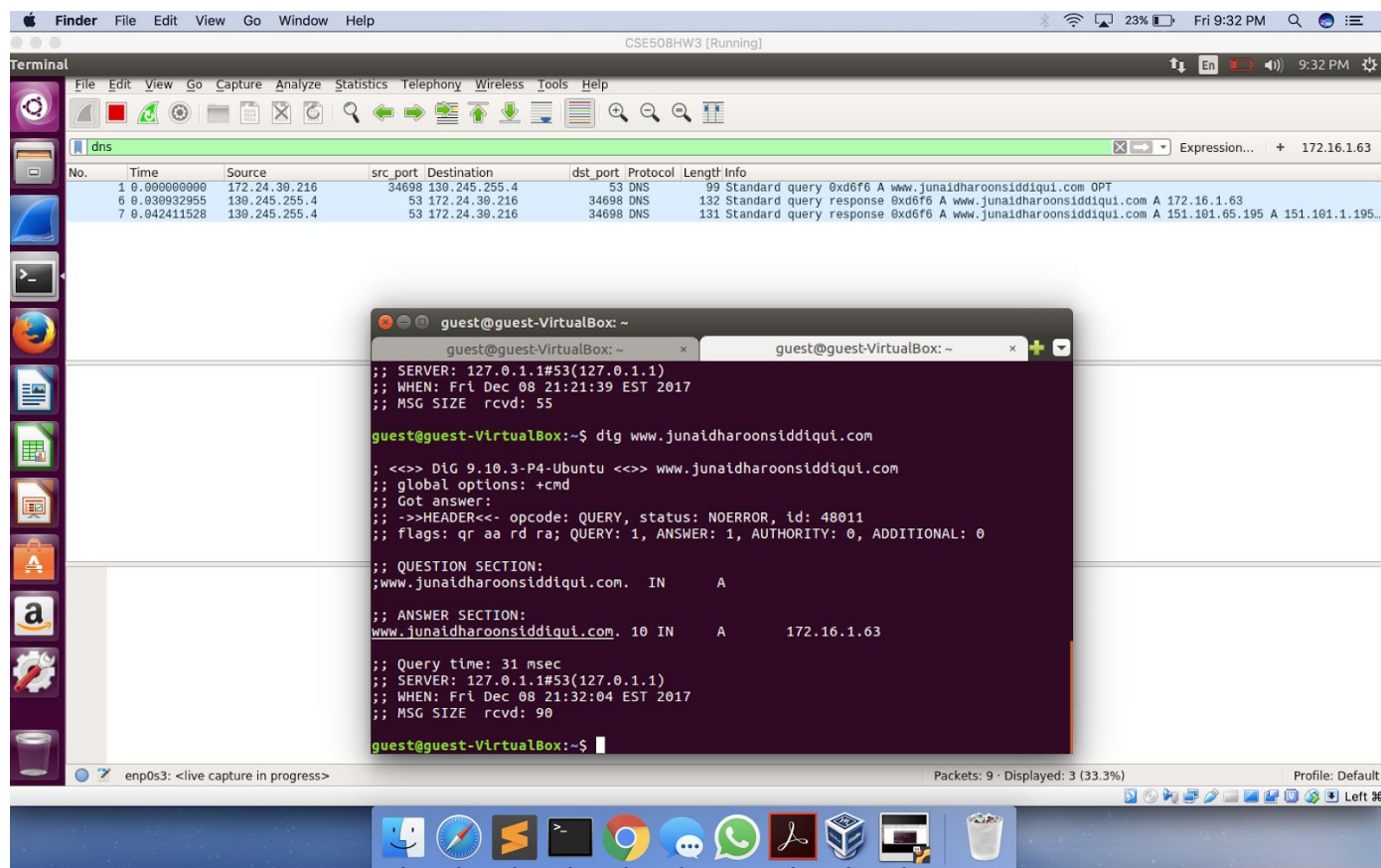
A terminal window titled "HW4 — Python • sudo — 97x29" showing the execution of a Python script. The user runs "sudo python dnsinject.py -h" and the output shows the help text: "interface: en0 hostnamesFile: hostname bpf: udp port 53". The user then runs the script without arguments, and the output shows "Sent 1 packets." and "Sent: IP / UDP / DNS Ans '172.16.1.63'".

```
Aynoors-MacBook-Pro:HW4 aynoorsaleem$ sudo python dnsinject.py -h
interface: en0 hostnamesFile: hostname bpf: udp port 53
.
Sent 1 packets.
Sent: IP / UDP / DNS Ans "172.16.1.63"
```

Detection output:

A terminal window titled "HW4 — Python • sudo — 97x29" showing the execution of a Python script. The user runs "sudo python dnsdetect.py" and the output shows a clear command, the interface "en0", the port "53", the date and time "2017-12-08 21:37:51", the event "DNS poisoning attempt", the TXID "8038", the request "www.junaidharoonsiddiqui.com.", and the answer "Answer1 151.101.1.195 Answer 2 ['172.16.1.63']".

```
=====
clear
^CAynoors-MacBook-Pro:HW4 aynoorsaleem$ sudo python dnsdetect.py
en0 None udp port 53
2017-12-08 21:37:51 DNS poisoning attempt
TXID 8038 Request www.junaidharoonsiddiqui.com.
Answer1 151.101.1.195 Answer 2 ['172.16.1.63']
=====
```



Citations:

Took help from the following links:

1. <http://www.cs.dartmouth.edu/~sergey/netreads/local/reliable-dns-spoofing-with-python-scapy-nfqueue.html#selection-813.9-1157.2>
2. <https://null-byte.wonderhowto.com/how-to/build-dns-packet-sniffer-with-scapy-and-python-0163601/>