Assignment 3 - DNS Attacks

Siddhant Deshmukh

February 18, 2016

1 Details of Machine

1. Processor
   - 2.7GHz dual-core Intel Core i5 processor with Turbo Boost up to 3.1GHz
   - 3MB shared L3 cache
2. Memory
   - 8GB of 1866MHz LPDDR3 (RAM)
   - 256GB PCIe-based flash storage (Internal Memory)
3. Graphics
   - Intel Iris Graphics 6100

2 Task 1 : Host file attack

1. Exact steps: I set up a lab environment with client, server and attacker machines all on the same network. Since the users machine is already compromised and I am on the user machine I can modify the ”host file”. I modified the host file and added the IP address of CNN for ”www.example.com”. As we can see in the screen-shots, when the user types ”www.example.com” in the browser the CNN website opens.
2. How a real-world attacker could leverage this sort of attack : An attacker can enter rouge entries in the host file and direct users to malicious websites. For example he could add an entry for Bank of America with the wrong IP address. He could then create a website that looks like Bank of America and steal your personal information after directing you to his website. This could lead to severe consequences and possibly a loss of bank account details. The frightening part is that the user will not suspect anything and everything will look normal to him.
3. Is this a viable attack : If the attacker is already on the users machine the user is already in grave danger. However, under normal circumstances it is very hard for the attacker to get on the users machine without the carelessness of the user(eg: setting easy passwords,etc). If the users machine is well protected by the user this attack becomes very difficult and practically unfeasible in most cases.
4. Screenshots:

Figure 1: Attack 1: Command to modify host file

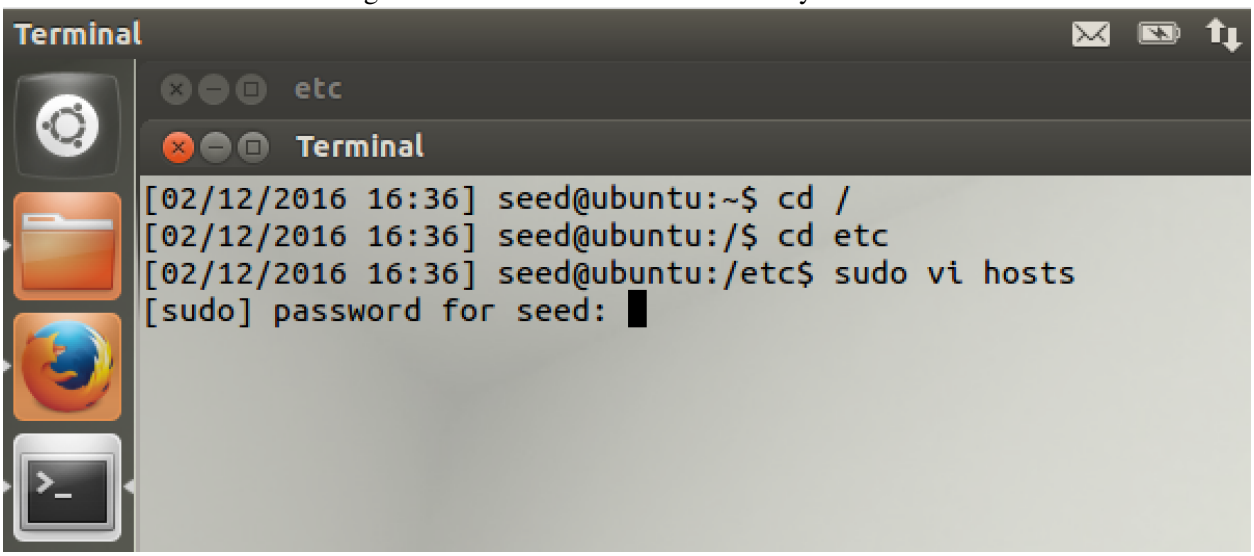


Figure 2: Attack 1: Adding IP address of CNN-157.166.226.25 for www.example.com in host file

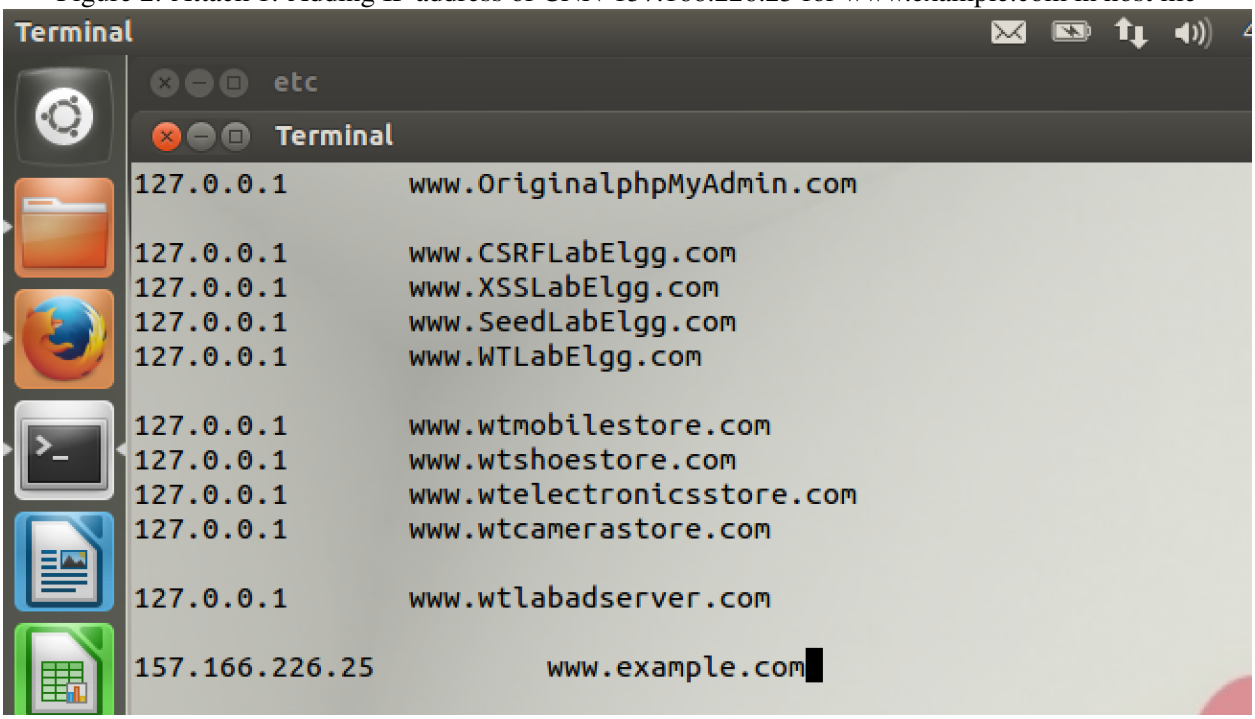


Figure 3: Attack1: User gets redirected to CNN after typing www.example.com in the browser

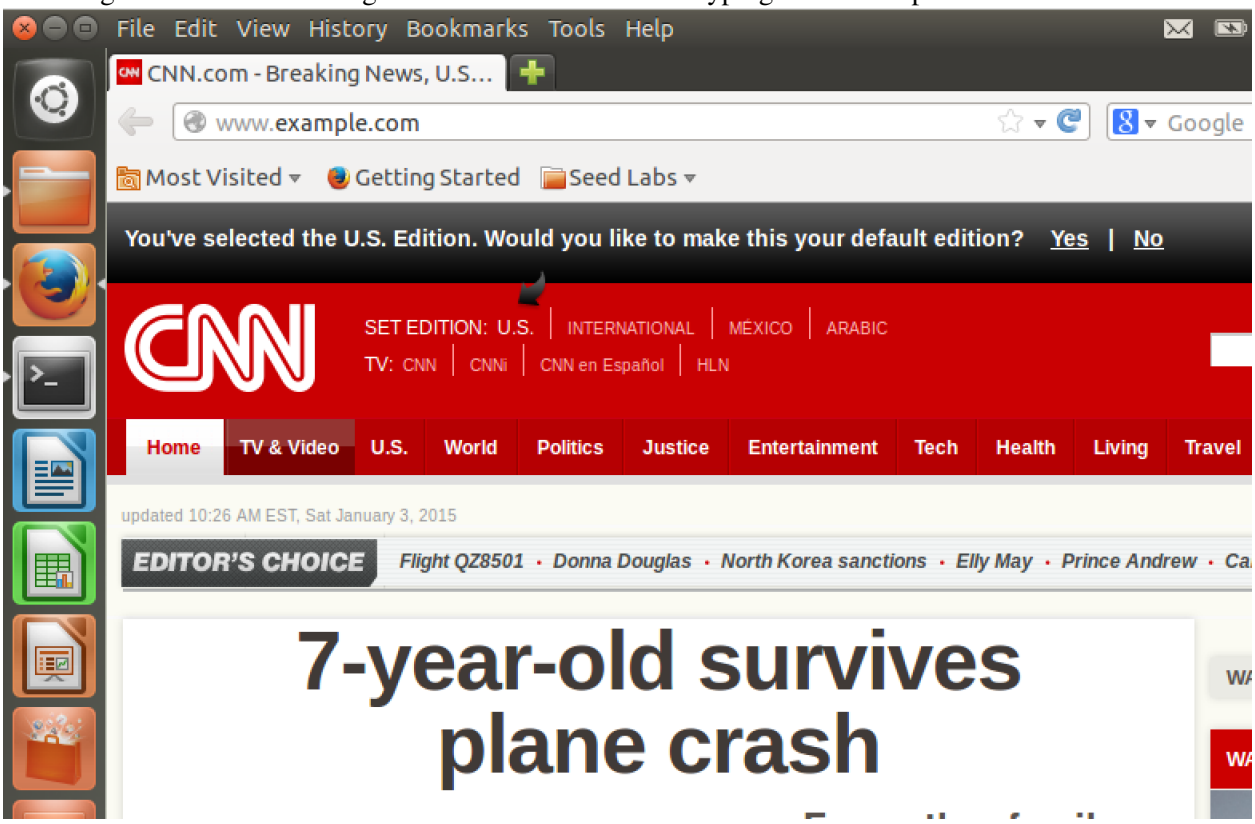


Figure 4: Attack1: PING to www.example.com also returns CNN IP address

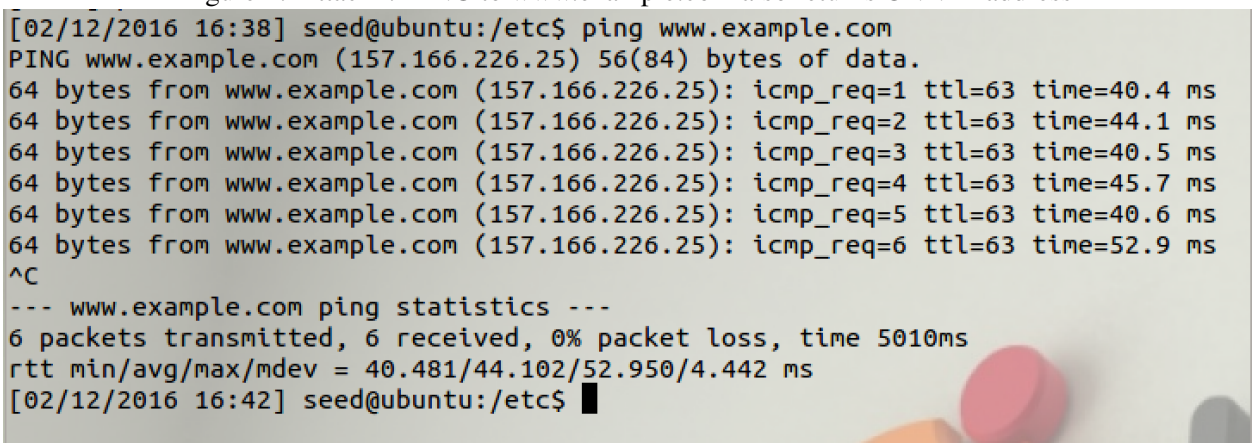
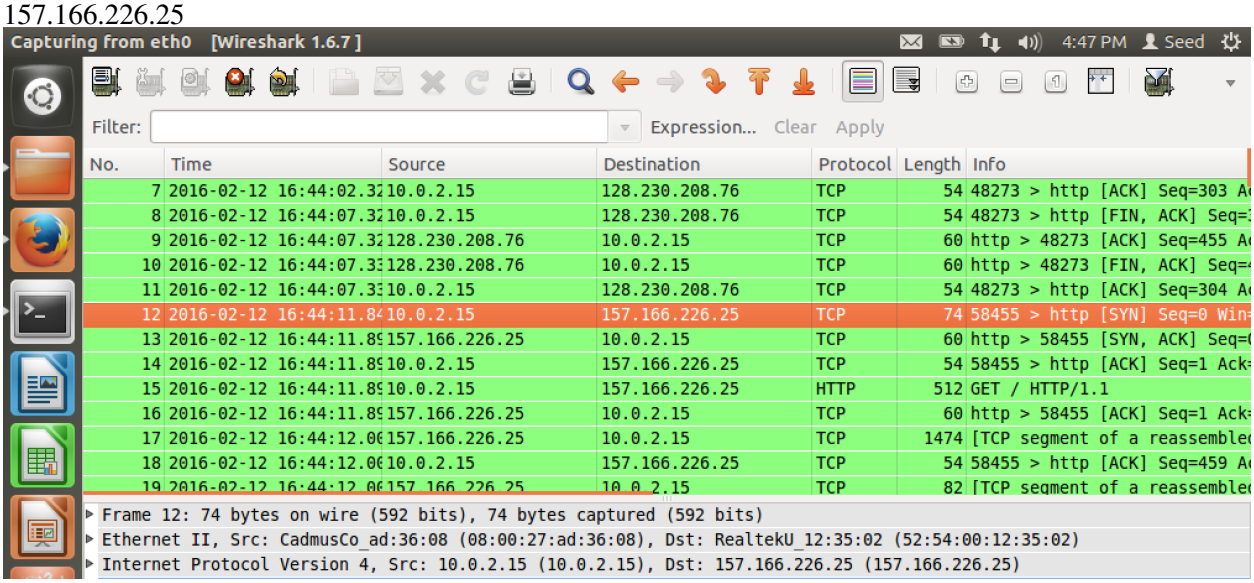


Figure 5: Attack 1: Wireshark shows the user-10.0.2.15 being directed to CNN's website with IP address-157.166.226.25



3 Task 2 : Host-Level Response Spoofing

1. Exact steps: I used the same lab environment with client, server and attacker machines all on the same network. I set up all the 3 VMs on 2 network adapters: NAT and host-only. Host-only enables all 3 VMs to communicate with each other and NAT connects them to the Internet. IP address of server: 192.168.56.101, IP address of client: 192.168.56.102, IP address of attacker: 192.168.56.103. I ran Netwag on the attacker machine and tried to sniff the packets from client to server. I used the Sniff and send option on Netwag. When the client send request to the server we have to try to send a spoofed response back to the client with a spoofed IP address. I kept the spoofed address as the IP address of CNN-157.166.226.25. After issuing a dig command from client and simultaneously running Netwag on the attacker I successfully sent a spoofed packet to client as we can see from the figures below.

2. How a real-world attacker could leverage this sort of attack : If an attacker is already on the victims network he can use a tool similar to Netwag and send a spoofed response back to the client as he knows the id of the request. This can have similar consequences to the previous attack. The attacker can direct the user to the wrong Bank of America site and steal his passwords and credit card details. This attack is general conducted by insiders; someone who we think we can trust but they end up betraying us.

3. Is this a viable attack : The more secure the network protocols, the harder it becomes for the attacker to pull-off such an attack. The attacker has to preform the attack for each request from the user and has to keep monitoring the packets. He has to ensure that the response is sent to the user before the actual response form the server. This can be extremely difficult to achieve in the real world. A more permanent and damaging attack is cache poisoning, which has long term effects once achieved; till the cache is updated.

4. Screenshots:

Figure 6: Attack 2: We can see the attack is successful as the answer section contains the IP address of CNN. The right hand side of the figure shows the Netwag configuration used.

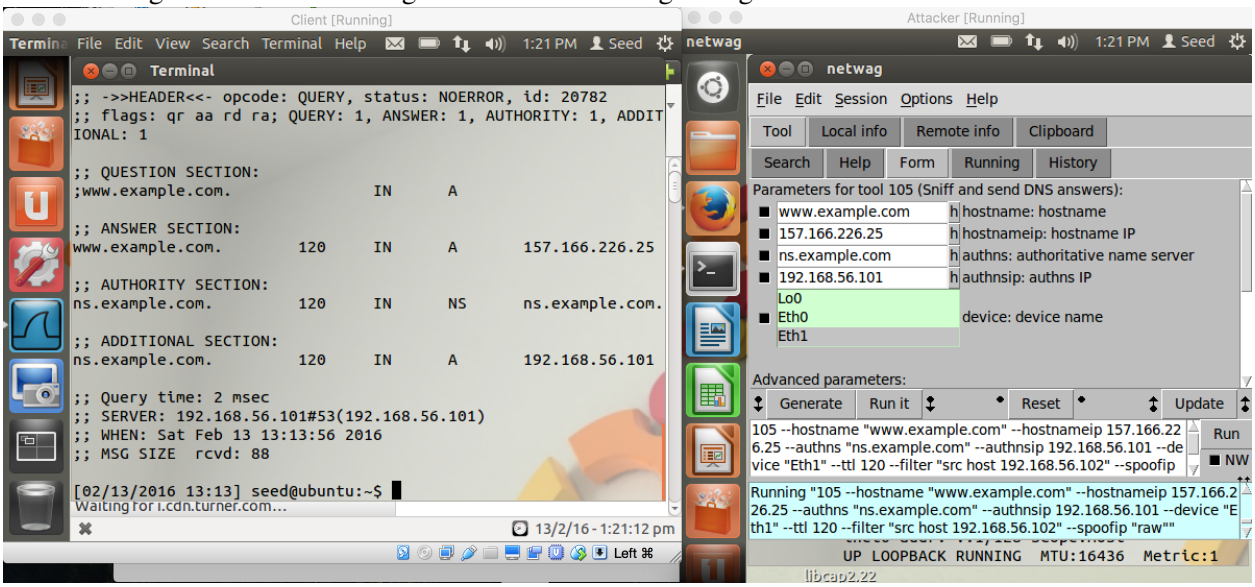


Figure 7: Attack 2: Netwag Configuration used for the attack

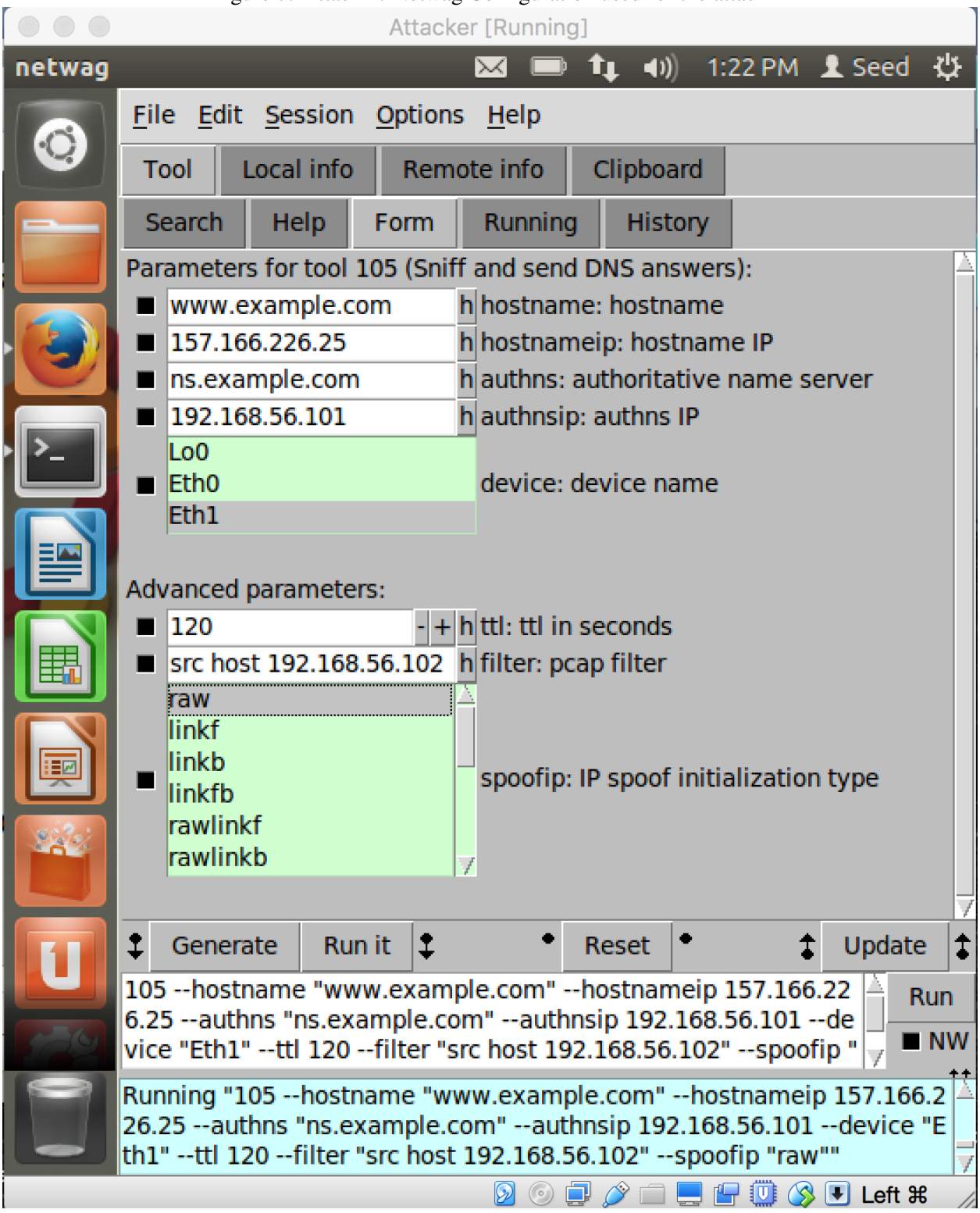


Figure 8: Attack 2: We can see the attack is successful as the answer section contains the IP address of CNN. The right hand side shows the captured packet on Netwag. As we can see both the response and captured packet have the same id=64835.

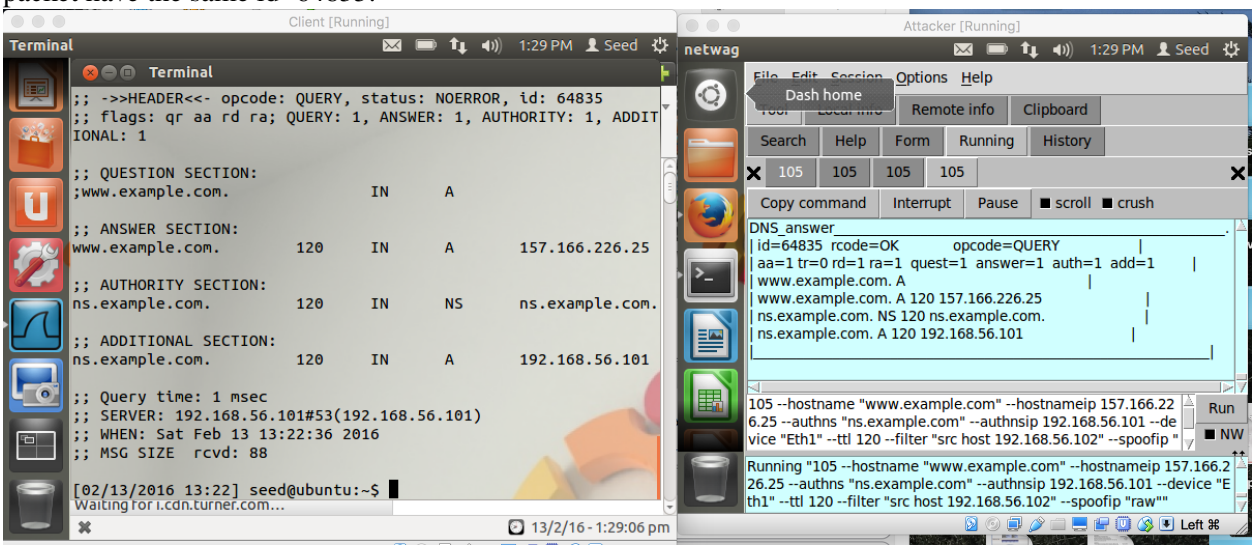
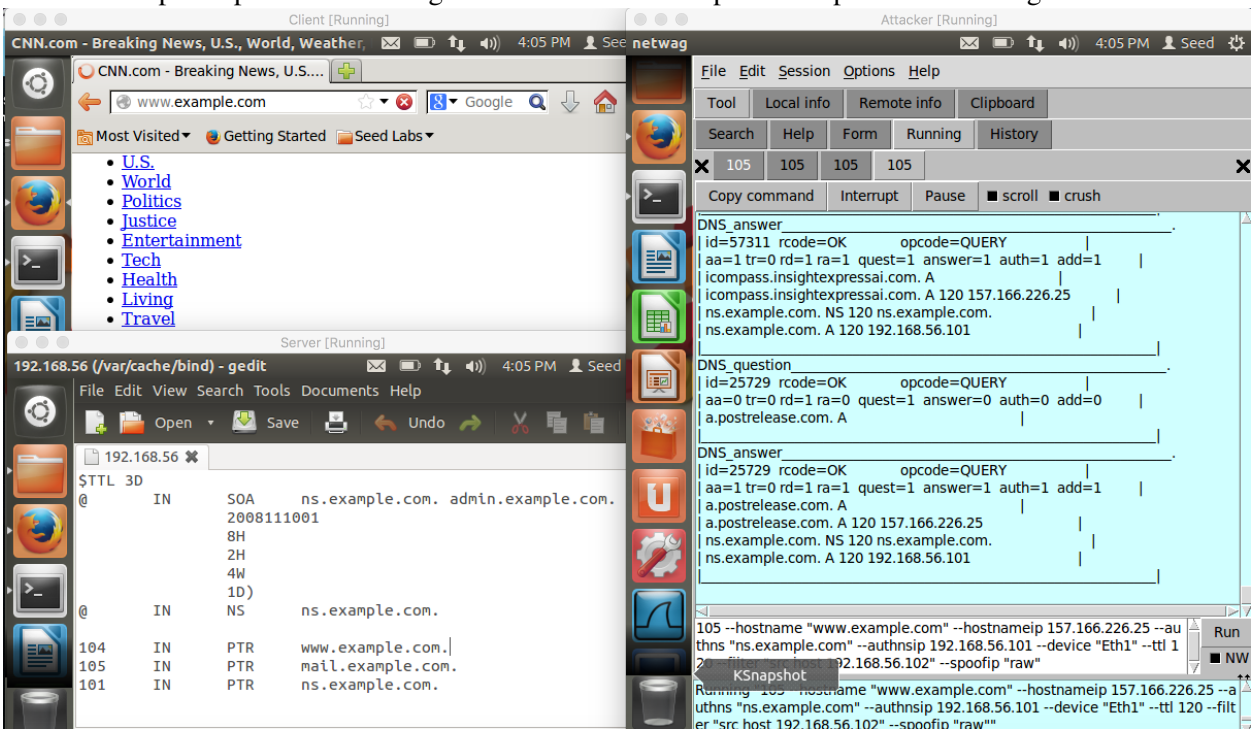


Figure 9: Attack 2: The same results show when www.example.com is typed in the browser; the CNN website opens. We can see the IP address stored on the server for www.example.com is 192.168.56.104 but the site that opens up is CNN. The right hand side shows the packets captured on Netwag.



4 Task 3 : Server-Level Response Spoofing

1. Exact steps: I used the same lab environment with client, server and attacker machines all on the same network. IP address of server: 192.168.56.101, IP address of client: 192.168.56.102, IP address of attacker: 192.168.56.103.First I made sure that all the 3 VMs are connected to the same network. I sent requests from the client to server and observed how the cache was filling up. I flush and dump the cache before I start the attack. I run Netwag on the attackers machine but this time target the server instead of the client. I send a request for "www.go-rts.com" to the server and send a spoofed response to server containing spoofed IP address for the site. As we can see in the figures the cache gets poisoned on the server and the response to the client is the spoofed response : 1.2.3.4.

2. How a real-world attacker could leverage this sort of attack : A real world attacker could poison the cache of a DNS server with a spoofed IP address and affect all the users that request for that particular address from the compromised name server. This has a much more widespread affect compared to the previous 2 attacks and if performed successfully could destroy a network or a DNS server. All the same threats mentioned in the previous 2 attacks apply here but the effects are amplified.

3. Is this a viable attack : Insiders can perform such an attack but it is difficult for an outsider to enter the network and perform this attack. Since the attacker has to be in the same network as the server and client this attack is very difficult to perform and less likely to be successful. If the attacker and dns server are on the same network the transaction ids can easily be duplicated and sent back as requests. However, to perform this attack from a remote machine requires a lot more effort and luck. Dan Kaminsky proposed an a DNS cache poisoning attack which is the last and final attack for the assignment.

4. Screenshots:

Figure 10: Attack 3: The left hand side shows the response received after the cache is poisoned. The right hand side shows the cache dump on the server with a malicious entry for www.go-rts.com. Hence we have successfully poisoned the cache.

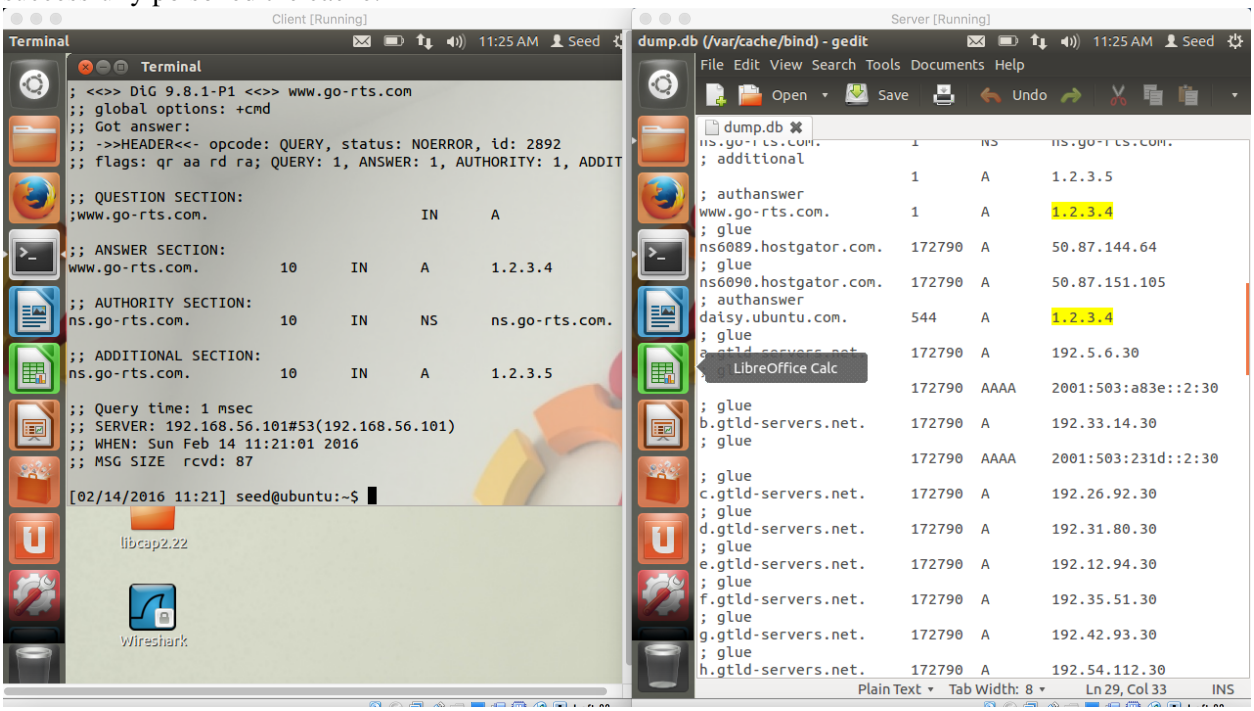


Figure 11: Attack 3: Netwag Configuration used for the attack

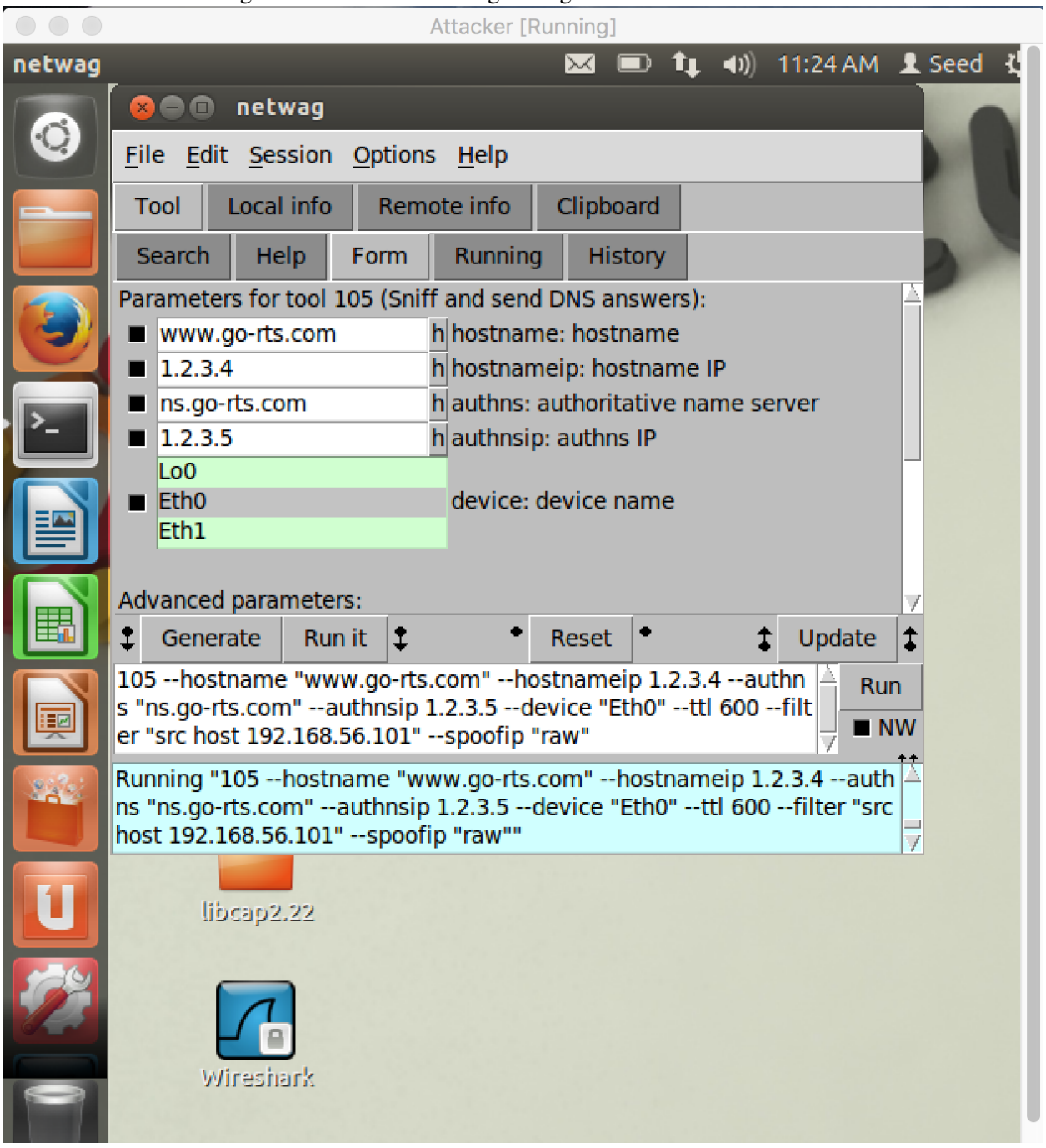
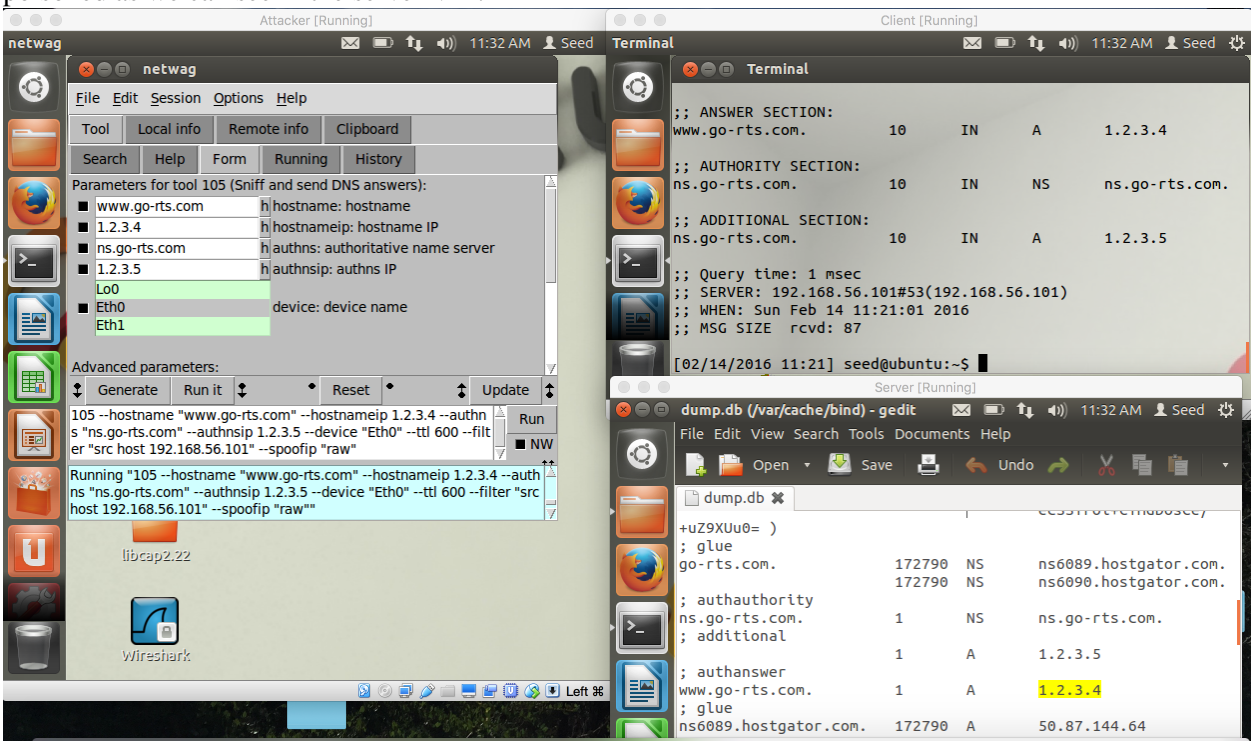


Figure 12: Attack 3: We can see the attacker, client and server all running after the attack is successful. Attacker is running Netwag and I gave a ”dig www.go-rts.com” command on the client. The server cache is poisoned as we can see in the server VM.



5 Task 4 : Kaminsky attack

1. Exact steps: I used the lab environment with forwarding server, server and attacker machines all on the same network. This time I configured all the VMs to 'host-only' adapter. IP address of server: 10.0.2.8, IP address of forwarding server: 10.0.2.10, IP address of attacker: 10.0.2.9. I added a delay on the server using "sudo tc qdisc add dev eth6 root delay 10ms". I set up the server to send any requests to dnsphishinglab.com to the forwarding server. The forwarding server contained the records with the actual address of dnsphishinglab.com. I modified pacgen to repeatedly send one request to the server, followed by a 1000 spoofed responses.

2. Structure of the attack tool written : I modified pacgen to repeatedly send one request followed by 1000 spoofed responses. Each time I changed the id of the responses based on a random function, hoping the id would match the initial request. I also added a random host name to the request and response and kept changing it after each iteration. I did this in order to continuously attack the server without worrying about caching of the previous requests. As we can see in the figures the cache is successfully poisoned with the spoofed address : 6.6.6.6.

3. Success: Initially I only sent one request followed by 1000 responses. I realized that the chances of success were too low. Hence I addeda while loop and ran the entire program continuously till the cache was poisoned. I had to delay the response from forwarding server but eventually achieved success.

4.

5. How a real-world attacker could leverage this sort of attack : An attacker can enter rouge entries in the host file and direct users to malicious websites. For example he could add an entry for Bank of America with the wrong IP address. He could then create a website that looks like Bank of America and steal your personal information after directing you to his website. This could lead to severe consequences and possibly a loss of bank account details. The frightening part is that the user will not suspect anything and everything will look normal to him.

6. Is this a viable attack : If the attacker is already on the users machine the user is already in grave danger. However, under normal circumstances it is very hard for the attacker to get on the users machine without the carelessness of the user(eg: setting easy passwords,etc). If the users machine is well protected by the user this attack becomes very difficult and practically unfeasible in most cases.

7. Precautions taken : I used host-only network to prevent packets from flowing onto the internet. I disconnected the NAT network used in the previous attacks.

8. Real world attack: Real world attacker could remotely poison the cache. of a DNS server with a spoofed IP address and affect all the users that request for that particular address from the compromised name server. This has a much more widespread affect compared to the first 2 attacks and if performed successfully could destroy a network or a DNS server. All the same threats mentioned in the previous 3 attacks apply here but the effects are amplified. This attack is more lethal as the attacker does not have to be on the same network and can continuously attack the server without worrying about caching.

9. This is a very viable attack as most people do not use dns-sec. If a machine does not have dns-sec a persistent attacker can successfully use this attack on them. We can see more about this attack and how it is performed in the images below.

10. Screenshots:

Figure 13: Attack 4: Wireshark running on the Forwarding server shows us that a request is send from attacker-10.0.2.9 to server-10.0.2.8 for e7e.dnsphishinglab.com. A spoofed response packet is sent using pacgen which sends a spoofed IP address of 6.6.6.6. After opening the spoofed dns packet we see that the response is for e7e.dnsphishinglab.com. This source for this packet is shown as fowarding server-10.0.2.10 and destination is shown as server-10.0.2.8.

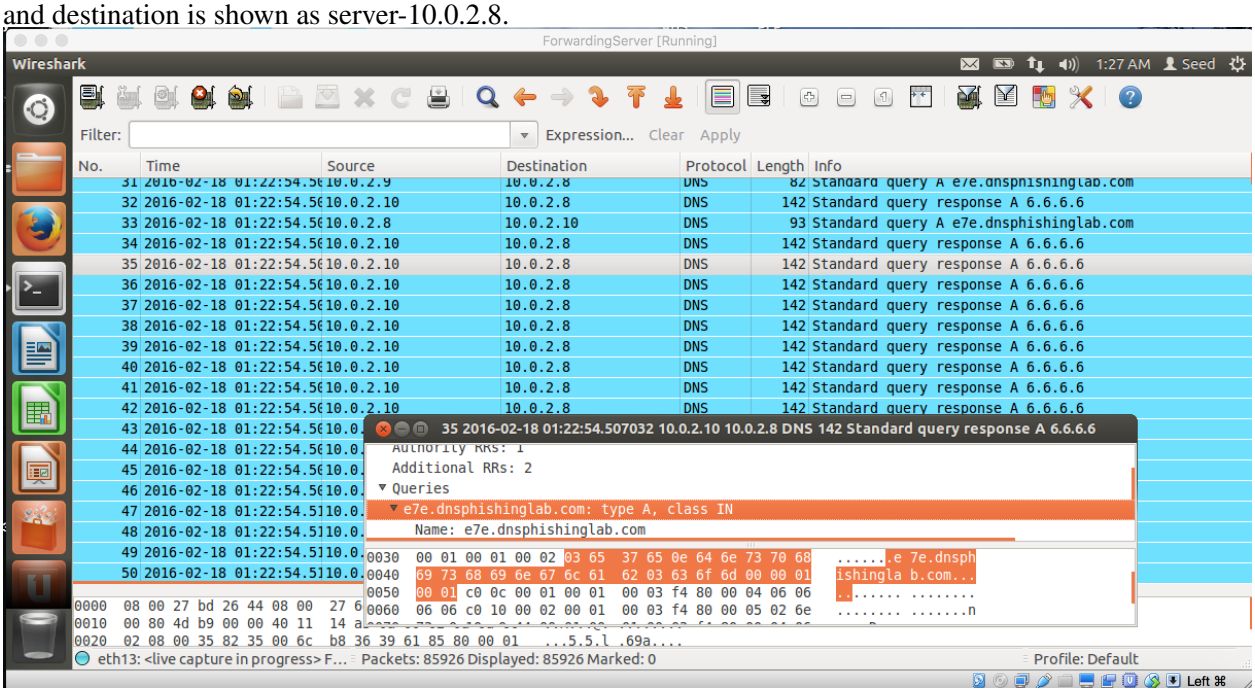


Figure 14: Attack 4: Left hand side shows pacgen running on the attacker and right hand side shows the poisoned cache. In this case only the IP address of the request was changed but a malicious attacker could have changed the name server from ns.dnsphishinglab.com to a malicious name server.

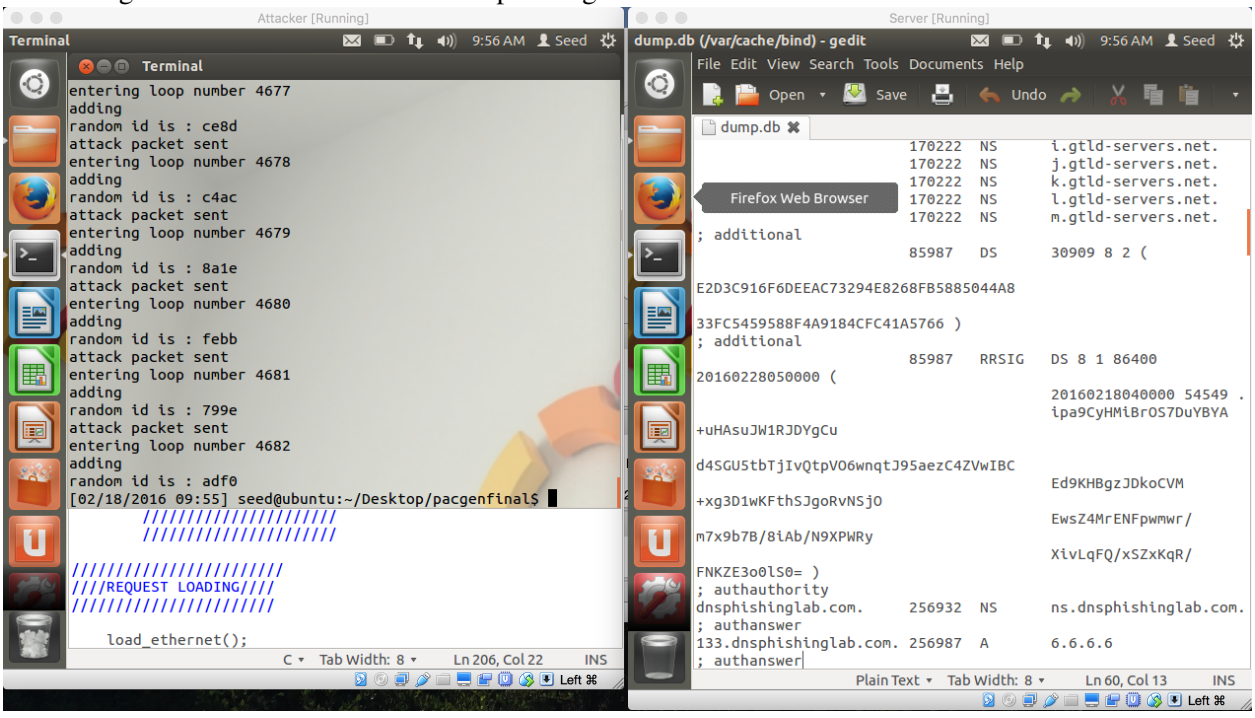


Figure 15: Attack 4: A dig from attacker machine now gives the spoofed IP address with which we had poisoned the cache.

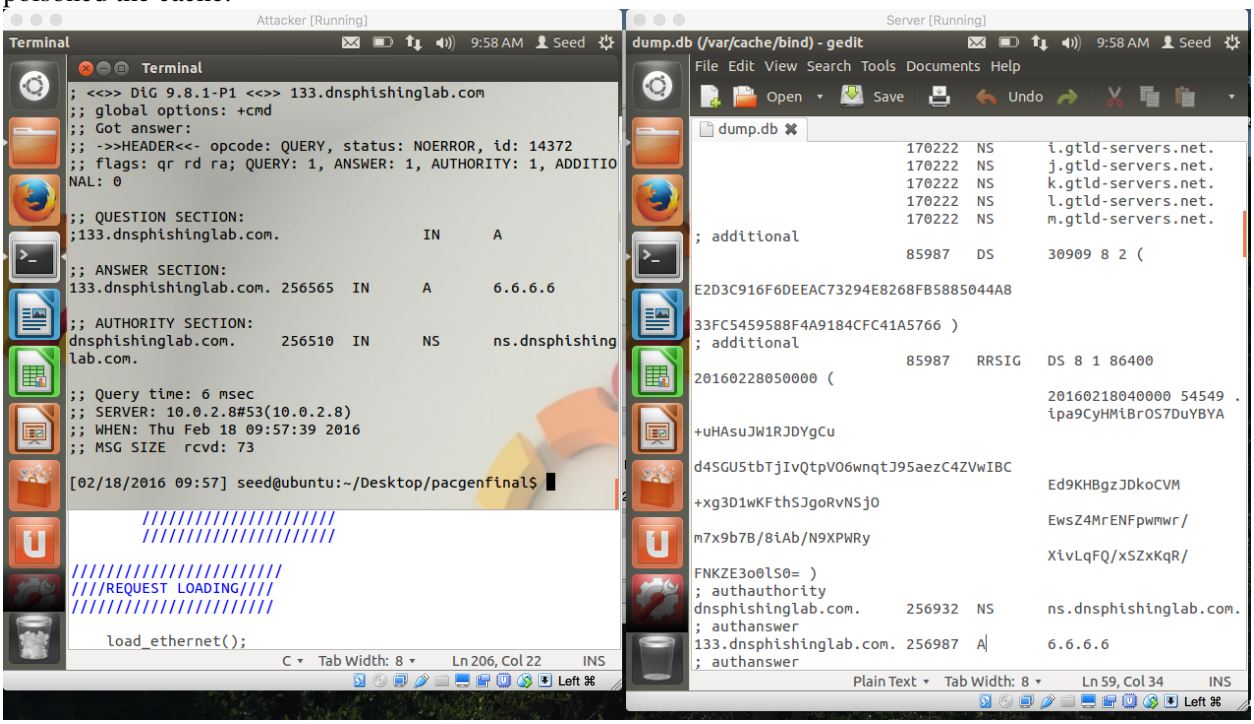


Figure 16: Attack 4: We see the server, attacker and forwarding server all running after the attack was successful. The forwarding servers shows the actual ip address for *.dnsphishinglab.com-1.1.1.100. Since we successfully poisoned the cache the response back to the attacker is the spoofed IP addresss-6.6.6.6