



Al-Najah National University
Department of Engineering and Information technology
Computer Network and Information Security

Experiment #7

Local DNS Attacks

Mohammad Bahjat Abdulhuq
Abdullah Harb
Instructor: Dr. Othman Othman

1 Abstract

In this experiment a couple of DNS Pharming attacks will be experienced to show how resolved domain-name IPs can be spoofed either by modifying on files manually or by using a specific tool such as netwox 105 as will be seen.

2 Introduction

The Domain Name System (DNS) is a distributed internet system that maps human-readable names to IP addresses. DNS servers translate queries from clients for domain-names into IP addresses, controlling which server an end user will reach when they type a domain name into their web browser.

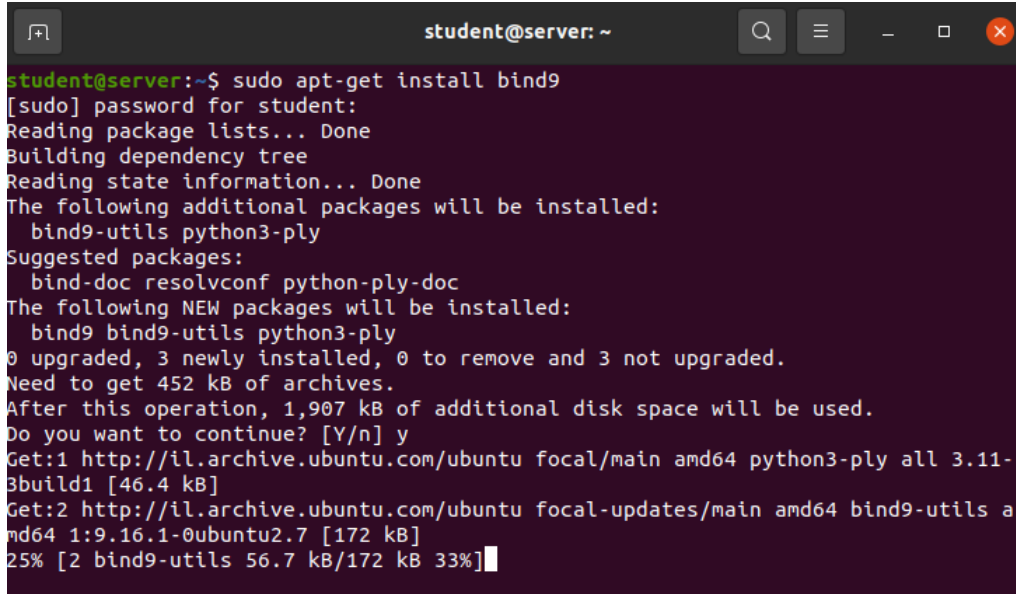
Types of DNS Service: Authoritative DNS is an DNS service provides an update mechanism that developers use to manage their public DNS names, Authoritative DNS has the final authority over a domain and is responsible for providing answers to recursive DNS servers with the IP address information. Recursive DNS: Clients typically do not make queries directly to authoritative DNS services. Instead, they generally connect to another type of DNS service known a resolver, or a recursive DNS service.

An attacker can take advantage of vulnerabilities in the Domain Name System (DNS) to perform Pharming attacks. One of the most common types of DNS attacks is the cache poisoning attack which happens when there are incorrect IP addresses stored on a DNS cache. For example, instead of leading a user to Google.com, the incorrect DNS cache entry might lead users to a phishing website that looks like the Google website and steel sensitive data of the user. Another form of DNS attacks is when an attacker take control on the victim's machine and modify on local files that are related to local DNS lookup and replace a real IP with a fake one as will be shown with the `/etc/hosts` file.

3 Procedure

3.1 Install and configure the DNS server and client

1. On the server VM the BIND9 DNS server was installed using the command illustrated in Figure 1.
2. The DNS server needs to read the `/etc/bind/named.conf` configuration file to start. This configuration file usually includes an option file called `/etc/bind/named.conf.options`. This file will be edited by adding this line to options: `dump-file "/var/cache/bind/dump.db"`; (see Figure 2)
3. A zone need to be created in the DNS server by adding the following contents to `/etc/bind/named.conf`, as shown in figure 3. The first zone represents the forwarder DNS which passes the DNS query to another DNS server (e.g.ISP server) is case if the

A terminal window titled 'student@server: ~' showing the command 'sudo apt-get install bind9'. The output shows the installation process, including reading package lists, building a dependency tree, and installing additional packages like bind9-utils and python3-ply. It also shows the disk space requirements and the progress of downloading packages from the Ubuntu repository.

```
student@server:~$ sudo apt-get install bind9
[sudo] password for student:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bind9-utils python3-ply
Suggested packages:
  bind-doc resolvconf python-ply-doc
The following NEW packages will be installed:
  bind9 bind9-utils python3-ply
0 upgraded, 3 newly installed, 0 to remove and 3 not upgraded.
Need to get 452 kB of archives.
After this operation, 1,907 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://il.archive.ubuntu.com/ubuntu focal/main amd64 python3-ply all 3.11-
3build1 [46.4 kB]
Get:2 http://il.archive.ubuntu.com/ubuntu focal-updates/main amd64 bind9-utils a
md64 1:9.16.1-0ubuntu2.7 [172 kB]
25% [2 bind9-utils 56.7 kB/172 kB 33%]
```

Figure 1: DNS server installation

A text editor window titled 'named.conf.options /etc/bind' showing the configuration for the DNS server. The file contains options for the directory, dump file, and firewall settings.

```
1 options {
2     directory "/var/cache/bind";
3     dump-file "/var/cache/bind/dump.db"
4     // If there is a firewall between you and nameservers you want
5     // to talk to, you may need to fix the firewall to allow multiple
6     // ports to talk. See http://www.kb.cert.org/vuls/id/800113
7 }
```

Figure 2: Modifying named.conf.options file

server does not have the answer. the second zone represents the resolver DNS which take the query and tries to figure out the answer to that query by recursively querying authoritative DNS servers for that domain.

4. The file name after the file keyword in the created zones is called the zone file. For forwarding DNS its name is **example.com.db**, The actual DNS resolution is done in the zone file. In the **/var/cache/bind/** directory, **example.com.db** file is created and some records is added as figure 4 shows.
5. The DNS reverse lookup file is also need to be setup in the **192.168.0.db** file inside the directory **/var/cache/bind/** as shown in figure 5.
 - The symbol '@' is a special notation meaning the origin from the named.conf. Therefore, '@' here stands for **example.com**. 'IN' means Internet. 'SOA' is short for Start Of Authority. This zone file contains 7 resource records
6. Now the server VM is ready to start the DNS server. by running the command indicated

```
1 // This is the primary configuration file for the BIND DNS server named.
2 //
3 // Please read /usr/share/doc/bind9/README.Debian.gz for information on the
4 // structure of BIND configuration files in Debian, *BEFORE* you customize
5 // this configuration file.
6 //
7 // If you are just adding zones, please do that in /etc/bind/named.conf.local
8
9 include "/etc/bind/named.conf.options";
10 include "/etc/bind/named.conf.local";
11 include "/etc/bind/named.conf.default-zones";
12 zone "example.com" {
13     type master;
14     file "/var/cache/bind/example.com.db"
15 };
16 zone "0.168.192.in-addr.arpa" {
17     type master
18     file "/var/cache/bind/192.168.0.db"
19 };
```

Figure 3: Zone Creation

```
1 $TTL 3D
2 @      IN      SOA      ns.example.com. admin.example.com. (
3       2008111001
4       8H
5       2H
6       4W
7       1D)
8
9 @      IN      NS       ns.example.com.
10 @      IN      MX       10 mail.example.com.
11
12 www    IN      A        192.168.0.101
13 mail   IN      A        192.168.0.102
14 ns     IN      A        192.168.0.254
15 *.example.com. IN      A 192.168.0.100
```

Figure 4: Zone setup

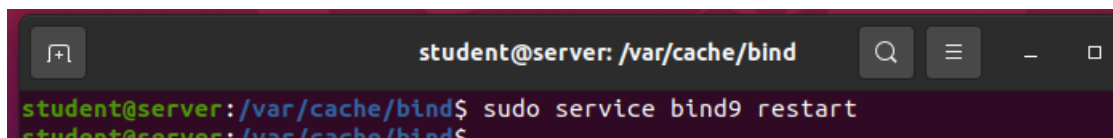
in figure 6 the new configurations will be applied.

7. On the user and attacker machines, the machine 192.168.0.254 should be the default DNS server. this is achieved by changing the DNS setting file `/etc/resolv.conf` of the user machine. (see figure 7)

A screenshot of a text editor window titled '*192.168.0.db' with the path '/var/cache/bind'. The window contains a Reverse DNS lookup file with the following content:

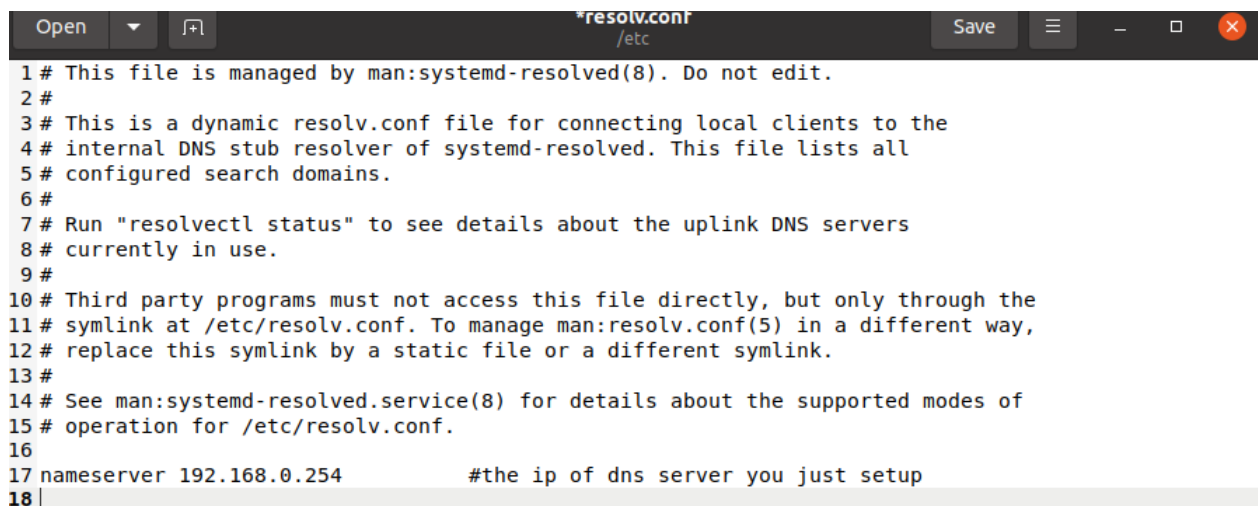
```
1 $TTL 3D
2 @      IN      SOA      ns.example.com. admin.example.com. (
3                               2008111001
4                               8H
5                               2H
6                               4W
7                               1D)
8
9 @      IN      NS       ns.example.com.
10
11
12 101    IN      PTR      www.example.com
13 102    IN      PTR      mail.example.com
14 10     IN      PTR      ns.example.com.
15
16
```

Figure 5: Reverse DNS lookup file .

A screenshot of a terminal window with the prompt 'student@server: /var/cache/bind'. The command 'sudo service bind9 restart' has been entered and executed.

```
student@server:/var/cache/bind$ sudo service bind9 restart
student@server:/var/cache/bind$
```

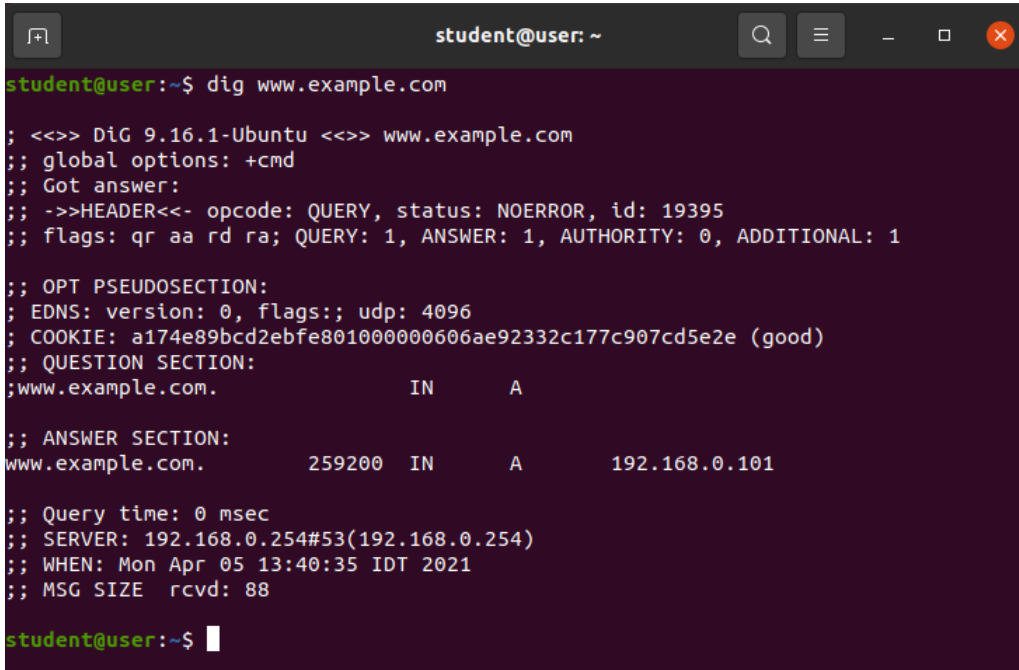
Figure 6: Restarting DNS

A screenshot of a text editor window titled '*resolv.conf' with the path '/etc'. The window contains the following content:

```
1 # This file is managed by man:systemd-resolved(8). Do not edit.
2 #
3 # This is a dynamic resolv.conf file for connecting local clients to the
4 # internal DNS stub resolver of systemd-resolved. This file lists all
5 # configured search domains.
6 #
7 # Run "resolvectl status" to see details about the uplink DNS servers
8 # currently in use.
9 #
10 # Third party programs must not access this file directly, but only through the
11 # symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
12 # replace this symlink by a static file or a different symlink.
13 #
14 # See man:systemd-resolved.service(8) for details about the supported modes of
15 # operation for /etc/resolv.conf.
16
17 nameserver 192.168.0.254          #the ip of dns server you just setup
18
```

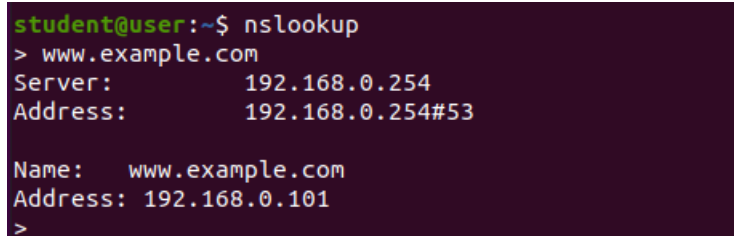
Figure 7: Changing default DNS server

8. Now to test the DNS do the following and notice the DNS server IP. As the figure 8 below shows ,the result of resolving the IP address of `www.example.com` in the answer section is now `192.169.0.101`, which is what we have set up in the DNS server.



```
student@user: ~  
student@user:~$ dig www.example.com  
  
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com  
;; global options: +cmd  
;; Got answer:  
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 19395  
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1  
  
;; OPT PSEUDOSECTION:  
; EDNS: version: 0, flags:; udp: 4096  
; COOKIE: a174e89bcd2ebfe801000000606ae92332c177c907cd5e2e (good)  
;; QUESTION SECTION:  
;www.example.com.                IN      A  
  
;; ANSWER SECTION:  
www.example.com.                259200  IN      A      192.168.0.101  
  
;; Query time: 0 msec  
;; SERVER: 192.168.0.254#53(192.168.0.254)  
;; WHEN: Mon Apr 05 13:40:35 IDT 2021  
;; MSG SIZE rcvd: 88  
  
student@user:~$
```

Figure 8: Output of `dig www.example.com`



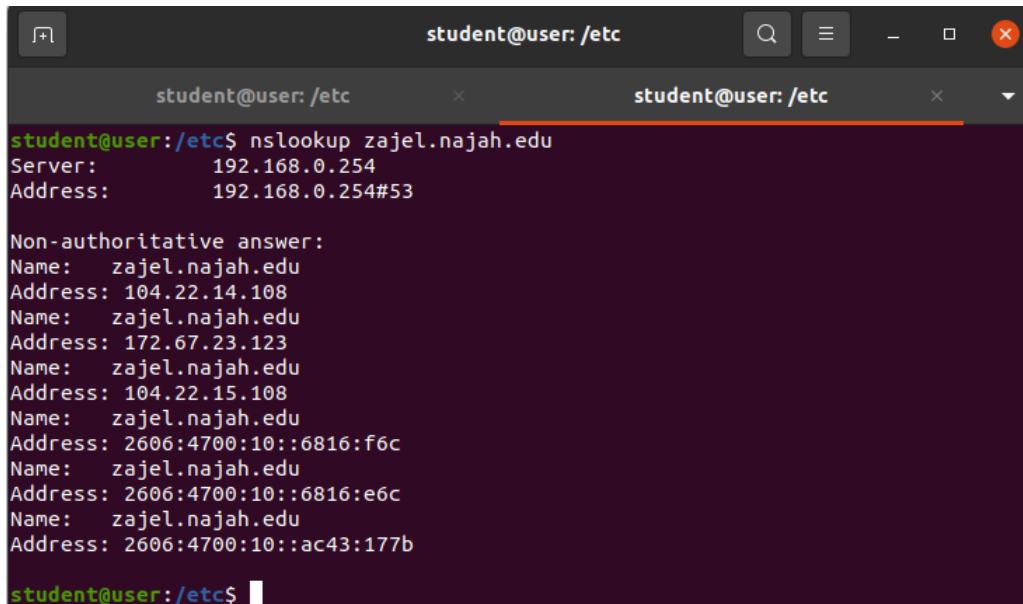
```
student@user:~$ nslookup  
> www.example.com  
Server:          192.168.0.254  
Address:         192.168.0.254#53  
  
Name:   www.example.com  
Address: 192.168.0.101  
>
```

Figure 9: Output of `nslookup` for `www.example.com`

3.2 Modifying HOSTS file by Attacker

- **Note :** we assume that the attacker has access to the user machine either local or remote to modify the Hosts file. The host name and IP address pairs in the HOSTS file (/etc/hosts) are used for local lookup; they take the preference over remote DNS lookups.

1. First, the IP address of zajel.najah.edu was found:

A terminal window titled 'student@user: /etc' with a dark purple background. It shows the command 'nslookup zajel.najah.edu' and its output. The output includes a server address (192.168.0.254) and a list of non-authoritative answers with names and IP addresses for zajel.najah.edu.

```
student@user: /etc$ nslookup zajel.najah.edu
Server:         192.168.0.254
Address:        192.168.0.254#53

Non-authoritative answer:
Name:   zajel.najah.edu
Address: 104.22.14.108
Name:   zajel.najah.edu
Address: 172.67.23.123
Name:   zajel.najah.edu
Address: 104.22.15.108
Name:   zajel.najah.edu
Address: 2606:4700:10::6816:f6c
Name:   zajel.najah.edu
Address: 2606:4700:10::6816:e6c
Name:   zajel.najah.edu
Address: 2606:4700:10::ac43:177b
student@user: /etc$
```

Figure 10: Zajel IP address

2. www.facebook.com was opened to make sure it's working normally.

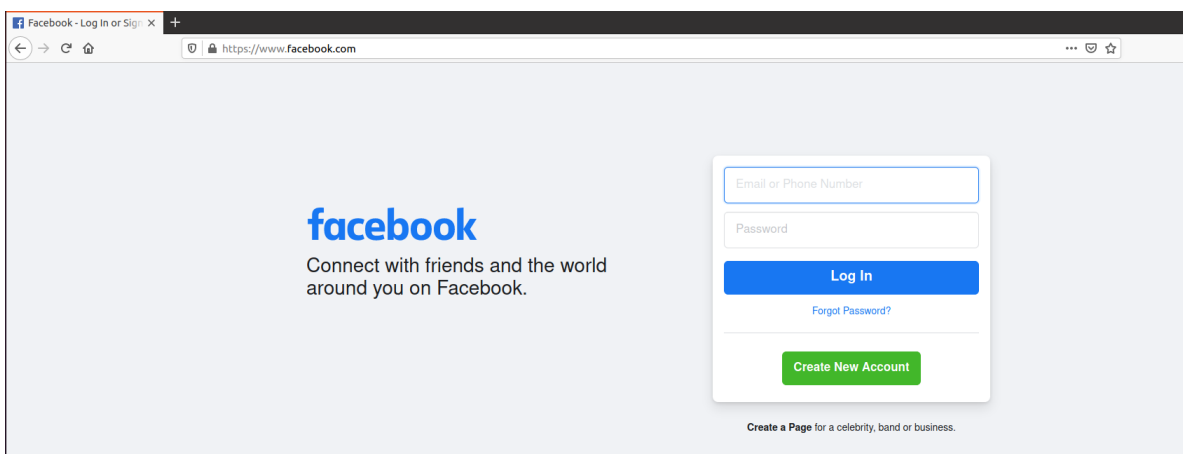
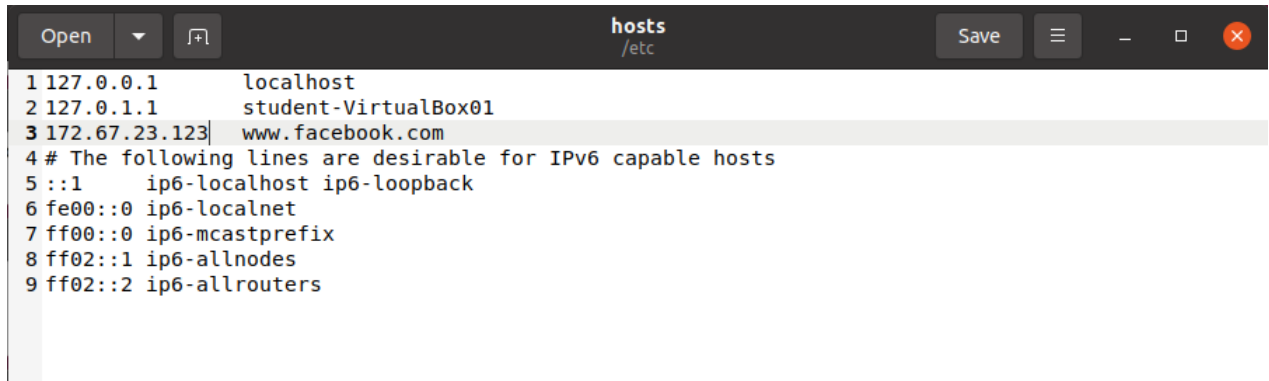


Figure 11: Opening Facebook

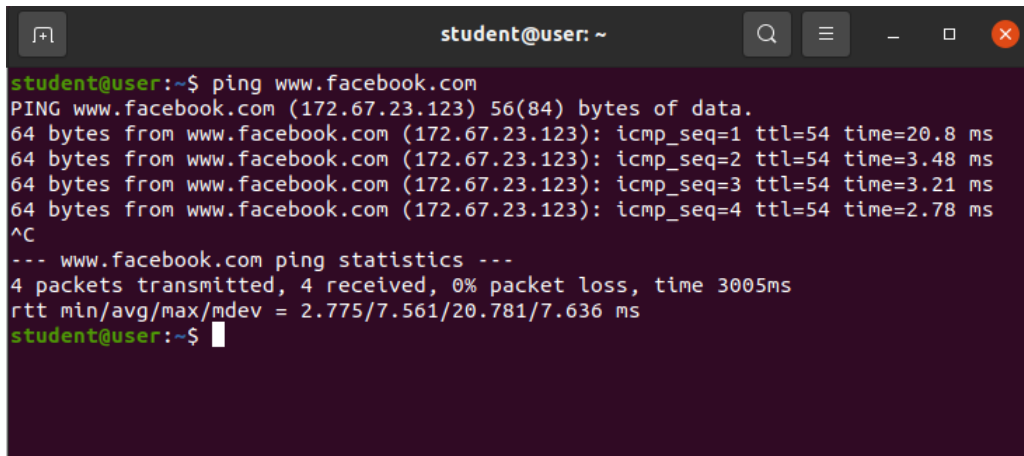
3. After that the hosts file (`/etc/hosts`) was edited to map `www.facebook.com` to Zajel's IP address.

A screenshot of a text editor window titled 'hosts /etc'. The window shows the contents of the /etc/hosts file. The first three lines are: '1 127.0.0.1 localhost', '2 127.0.1.1 student-VirtualBox01', and '3 172.67.23.123 www.facebook.com'. The third line is highlighted. Below these are several lines of IPv6 addresses and their corresponding hostnames, starting with a comment: '# The following lines are desirable for IPv6 capable hosts'.

```
1 127.0.0.1      localhost
2 127.0.1.1      student-VirtualBox01
3 172.67.23.123  www.facebook.com
4 # The following lines are desirable for IPv6 capable hosts
5 ::1           ip6-localhost ip6-loopback
6 fe00::0       ip6-localnet
7 ff00::0       ip6-mcastprefix
8 ff02::1       ip6-allnodes
9 ff02::2       ip6-allrouters
```

Figure 12: Modifying 'hosts' file

4. In order to test connectivity, Facebook was opened again, but the browser security prevented us from opening the site, thus, ping was used to test the attack, and the IP was spoofed successfully as indicated in the figure below.

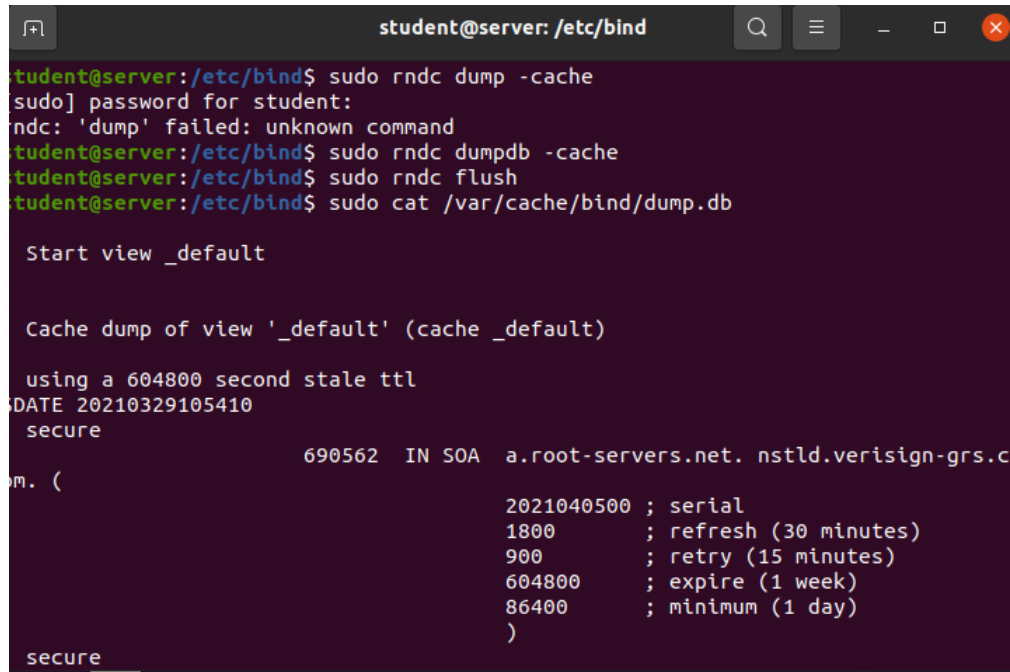
A screenshot of a terminal window titled 'student@user: ~'. The terminal shows the command 'ping www.facebook.com' being executed. The output shows four successful ping requests to the IP address 172.67.23.123, with varying response times. The statistics at the bottom show 4 packets transmitted, 4 received, 0% packet loss, and a time of 3005ms.

```
student@user:~$ ping www.facebook.com
PING www.facebook.com (172.67.23.123) 56(84) bytes of data.
64 bytes from www.facebook.com (172.67.23.123): icmp_seq=1 ttl=54 time=20.8 ms
64 bytes from www.facebook.com (172.67.23.123): icmp_seq=2 ttl=54 time=3.48 ms
64 bytes from www.facebook.com (172.67.23.123): icmp_seq=3 ttl=54 time=3.21 ms
64 bytes from www.facebook.com (172.67.23.123): icmp_seq=4 ttl=54 time=2.78 ms
^C
--- www.facebook.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 2.775/7.561/20.781/7.636 ms
student@user:~$
```

Figure 13: Ping Facebook.

3.3 DNS Server Cache Poisoning

1. First of all, this attack requires to clear and flush the cache to dump and view the DNS server's cache, we run these commands as shown in the figure.

A terminal window titled 'student@server: /etc/bind' with standard window controls. The terminal shows a series of commands and their outputs. The first command 'sudo rndc dump -cache' fails with 'unknown command'. The second 'sudo rndc dumpdb -cache' also fails. The third 'sudo rndc flush' succeeds. The fourth 'sudo cat /var/cache/bind/dump.db' displays the contents of the DNS cache dump, including view information, a stale TTL, a date, and SOA record details for 'a.root-servers.net'.

```
student@server:/etc/bind$ sudo rndc dump -cache
[sudo] password for student:
rndc: 'dump' failed: unknown command
student@server:/etc/bind$ sudo rndc dumpdb -cache
student@server:/etc/bind$ sudo rndc flush
student@server:/etc/bind$ sudo cat /var/cache/bind/dump.db

Start view _default

Cache dump of view '_default' (cache _default)

using a 604800 second stale ttl
DATE 20210329105410
secure
690562 IN SOA a.root-servers.net. nstld.verisign-grs.c
m. (
2021040500 ; serial
1800      ; refresh (30 minutes)
900       ; retry (15 minutes)
604800    ; expire (1 week)
86400     ; minimum (1 day)
)
secure
```

Figure 14: Flushing the DNS cache

2. On the Attacker VM the following attack was conducted by using netwox 105 tool which refers to sniffing and sending DNS answers. (see Figure 15)
 - The response to DNS server is spoofed, in which the filter field was set to 'src host 192.168.0.254', that is the IP address of the DNS server.
 - The TTL field (time-to-live) was used to indicate how long the fake answer is wanted to stay in the DNS server's cache.
3. Now in the user VM when trying to dig or using nslookup command to resolve the www.example.com IP, the answer came from the fake IP which was set by the attacker. (see the figure 16 below)
4. Figure 17 shows the attacker's VM while the attack is running successfully.

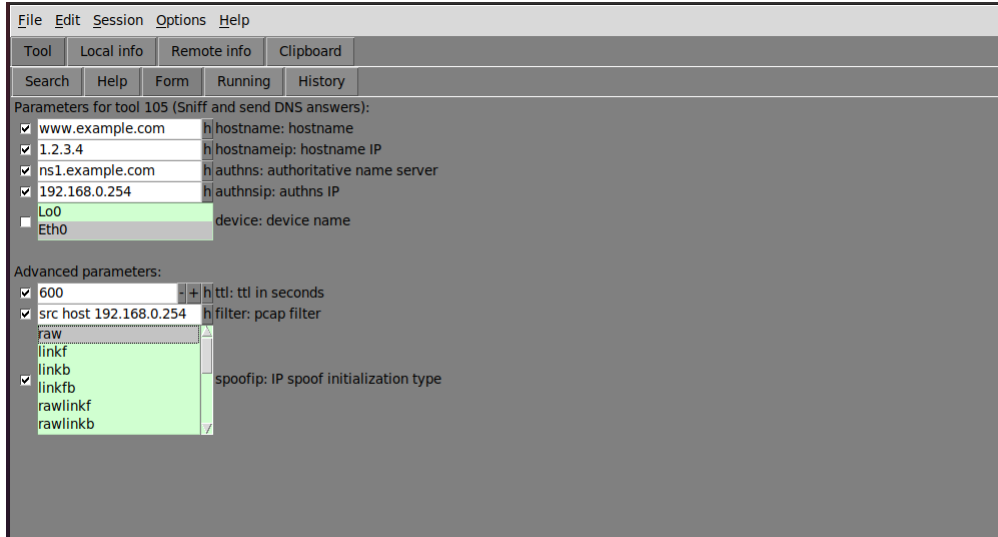


Figure 15: Configuring the attack parameters

```
student@user:~$ dig zajel.najah.edu

; <<>> DiG 9.16.1-Ubuntu <<>> zajel.najah.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27071
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: a4903481bfcfbad901000000606aefb37f54bb957ad868ba (good)
;; QUESTION SECTION:
;zajel.najah.edu.                IN      A

;; ANSWER SECTION:
zajel.najah.edu.                600     IN      A      1.2.3.4

;; Query time: 1571 msec
;; SERVER: 192.168.0.254#53(192.168.0.254)
;; WHEN: Mon Apr 05 14:08:35 IDT 2021
;; MSG SIZE rcvd: 88

student@user:~$ nslookup zajel.najah.edu
Server:                192.168.0.254
Address:               192.168.0.254#53

Non-authoritative answer:
Name:   zajel.najah.edu
Address: 1.2.3.4
** server can't find zajel.najah.edu: SERVFAIL

student@user:~$
```

Figure 16: Result of using dig and nslookup example.com

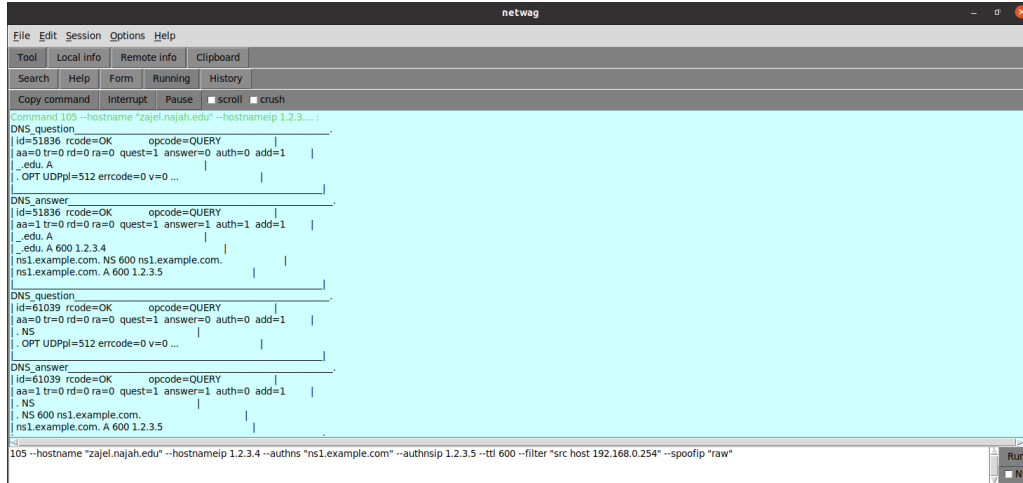


Figure 17: Running the attack

4 Conclusion

DNS attacks could be really danger the network was not well protected, because it may lead to security breaches in terms of availability or confidentiality. DNS lookup procedure can be exploited by a hacker that can modify on some files related to DNS service that will eventually lead to redirecting a website to another, or by spoofing the resolved IP of a query reply from the server that will also deny the access of a legitimate user, thus it's important to make sure that a user is using the correct DNS server.

5 References

<https://cybernews.com/resources/what-is-a-dns-attack/>

<https://serverfault.com/questions/661821/what-s-the-difference-between-recursion-and-forwarding-in-bind>

https://www.cloudns.net/dynamic-dns/?utm_source=microsoftutm_medium=dynamicdnsutm_campaign=dynamicdnskeyword

<https://web.ecs.syr.edu/wedu/Teaching/cis758/netw522/netwox-doc_{html}/tools/105.html>