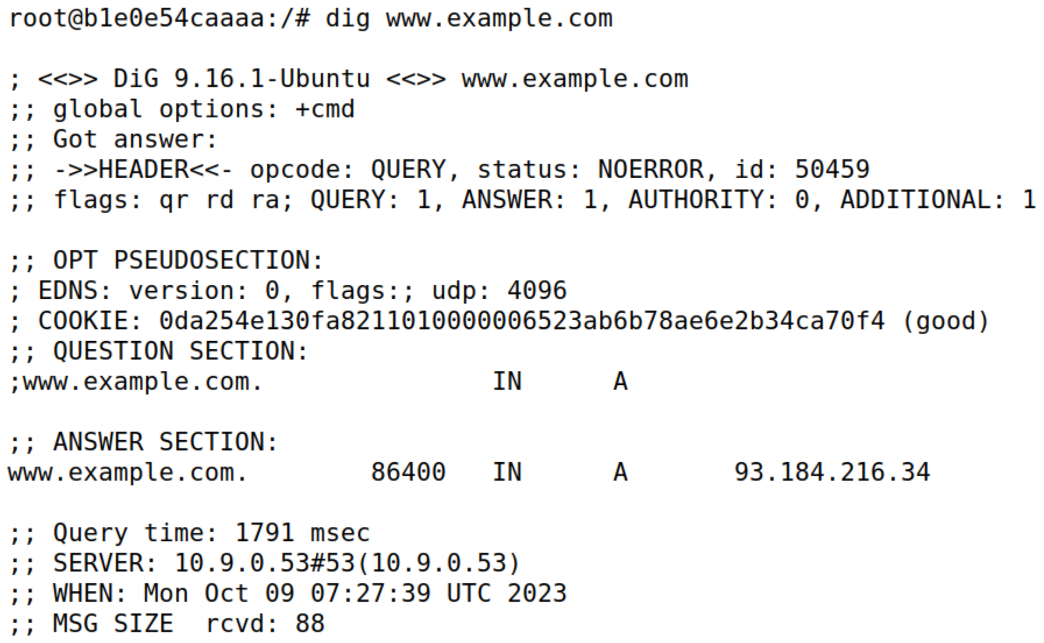
Lab 3

Student ID: 1005288

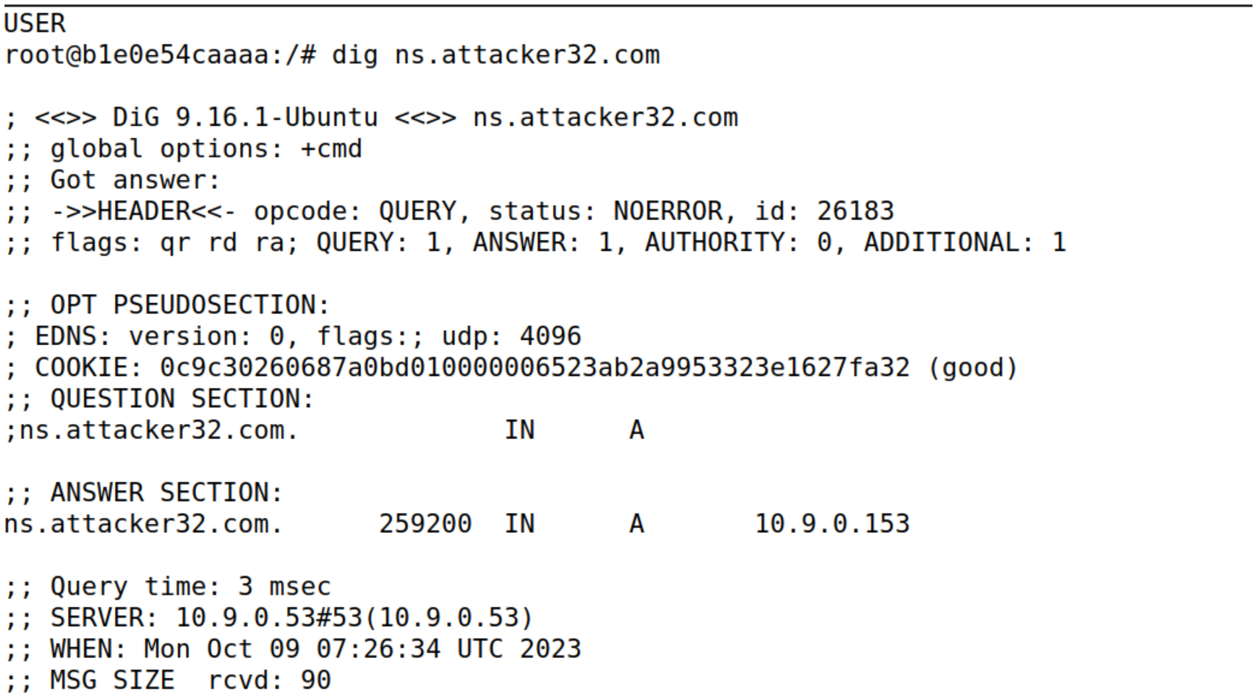
As usual, scripts are named and put under the corresponding task folder

1

This task is to test if the DNS setup was done correctly
dig ns.attacker32.com and dig www.example.com shows that the user got his answer from the local dns server.

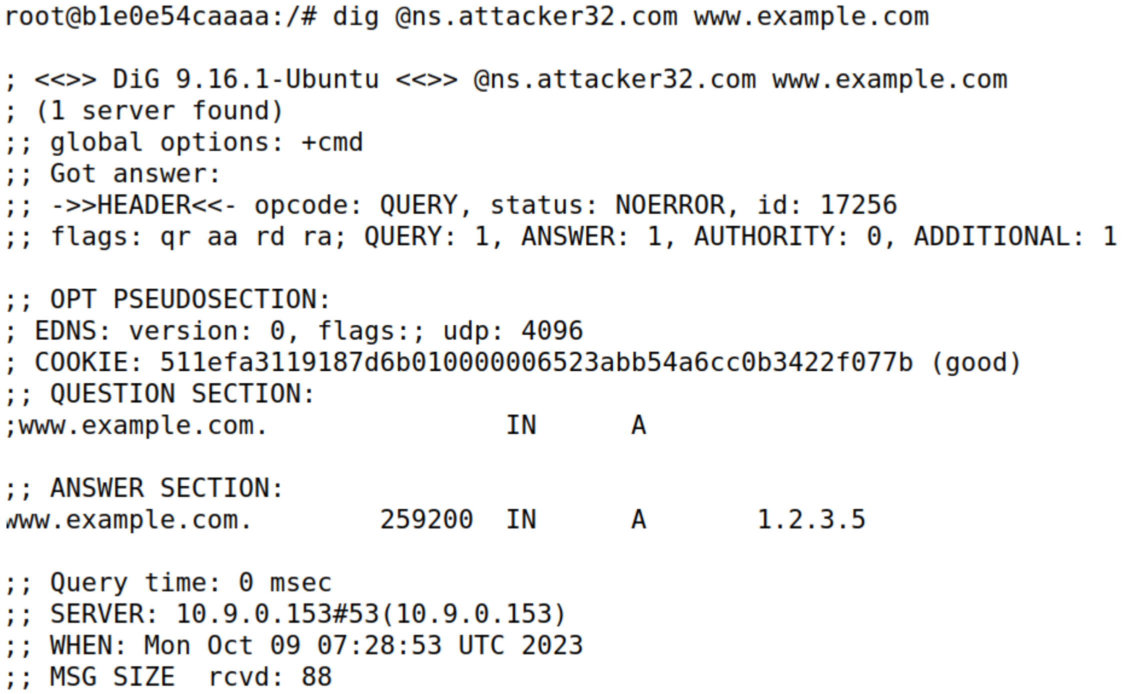


```
root@b1e0e54caaaa:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50459
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0da254e130fa8211010000006523ab6b78ae6e2b34ca70f4 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        86400   IN      A       93.184.216.34

;; Query time: 1791 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Oct 09 07:27:39 UTC 2023
;; MSG SIZE  rcvd: 88
```



```
USER
root@b1e0e54caaaa:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26183
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 0c9c30260687a0bd010000006523ab2a9953323e1627fa32 (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

;; ANSWER SECTION:
ns.attacker32.com.      259200  IN      A       10.9.0.153

;; Query time: 3 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Oct 09 07:26:34 UTC 2023
;; MSG SIZE  rcvd: 90
```

But when we use the @ option, the IP of the dns server becomes 10.9.0.153.

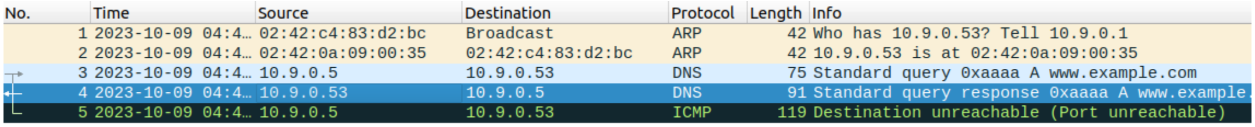


```
root@b1e0e54caaaa:/# dig @ns.attacker32.com www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> @ns.attacker32.com www.example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17256
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 511efa3119187d6b010000006523abb54a6cc0b3422f077b (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.2.3.5

;; Query time: 0 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Mon Oct 09 07:28:53 UTC 2023
;; MSG SIZE  rcvd: 88
```

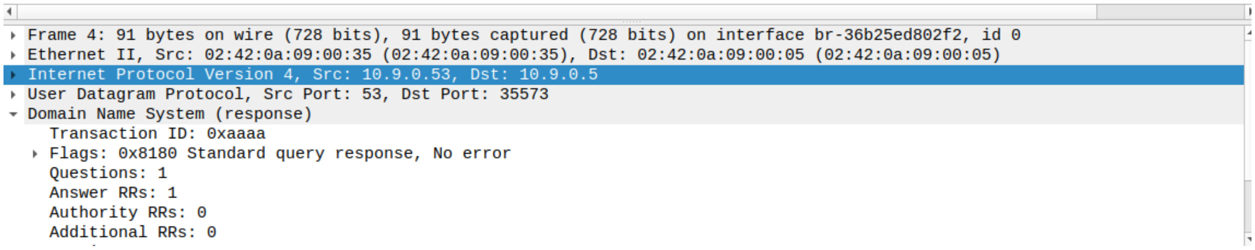
So the purpose of this attack is to route all user queries through 10.9.0.153 instead of 10.9.0.53, in this case, cause the IP in the answer section to become 1.2.3.5 when the user queries www.example.com

2

dst IP = 10.9.0.53 (local DNS server we want to target IP)
src IP = 10.9.0.5 (user's IP)
UDP dport =53 as it is the DNS port
UDP sport = 35573, it was taken from a packet captured on Wireshark, but can also be any random number generated by the user machine



```
No.     Time             Source             Destination        Protocol Length Info
      1 2023-10-09 04:4… 02:42:c4:83:d2:bc  Broadcast          ARP        42 Who has 10.9.0.53? Tell 10.9.0.1
      2 2023-10-09 04:4… 02:42:0a:09:00:35  02:42:c4:83:d2:bc  ARP        42 10.9.0.53 is at 02:42:0a:09:00:35
      3 2023-10-09 04:4… 10.9.0.5           10.9.0.53          DNS        75 Standard query 0xaaaa A www.example.com
      4 2023-10-09 04:4… 10.9.0.53          10.9.0.5           DNS        91 Standard query response 0xaaaa A www.example.
      5 2023-10-09 04:4… 10.9.0.5           10.9.0.53          ICMP      119 Destination unreachable (Port unreachable)
```

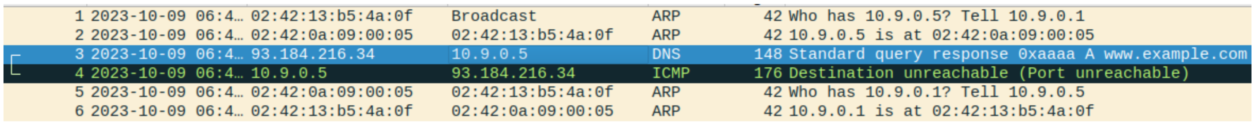


```
▸ Frame 4: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface br-36b25ed802f2, id 0
▸ Ethernet II, Src: 02:42:0a:09:00:35 (02:42:0a:09:00:35), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
▸ Internet Protocol Version 4, Src: 10.9.0.53, Dst: 10.9.0.5
▸ User Datagram Protocol, Src Port: 53, Dst Port: 35573
▾ Domain Name System (response)
    Transaction ID: 0xaaaa
  ▸ Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
```

The packets from wireshark shows that there is no error in our request and it prompted a response.
There is an extra packet that says "port unreachable" as the src machine was not listening for any message on that port, which was the case since the request.was sent by the attacker.

3

name = www.example.com since we want to give fake DNS records from that domain
domain = example.com as implied from the name,
ns = ns.attacker32.com (attacker's nameserver) as we are spoofing a reply
The reply should be the opposite of a DNS request, UDP dport and sport are swapped,
dst IP = 10.9.0.5 (user’s IP), src IP = 93.184.216.34 (www.example.com)



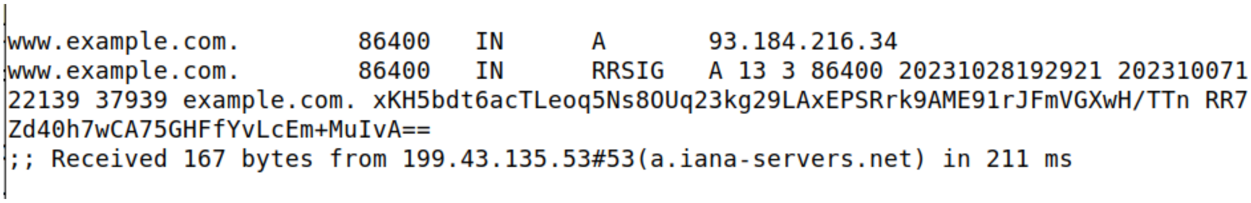
```
1 2023-10-09 06:4… 02:42:13:b5:4a:0f   Broadcast           ARP     42 Who has 10.9.0.5? Tell 10.9.0.1
2 2023-10-09 06:4… 02:42:0a:09:00:05   02:42:13:b5:4a:0f   ARP     42 10.9.0.5 is at 02:42:0a:09:00:05
3 2023-10-09 06:4… 93.184.216.34       10.9.0.5            DNS    148 Standard query response 0xaaaa A www.example.com
4 2023-10-09 06:4… 10.9.0.5            93.184.216.34       ICMP   176 Destination unreachable (Port unreachable)
5 2023-10-09 06:4… 02:42:0a:09:00:05   02:42:13:b5:4a:0f   ARP     42 Who has 10.9.0.1? Tell 10.9.0.5
6 2023-10-09 06:4… 02:42:13:b5:4a:0f   02:42:0a:09:00:05   ARP     42 10.9.0.1 is at 02:42:13:b5:4a:0f
```

After sending the spoofed response, Wireshark was able to capture the packet sent from www.example.com to the user IP
Again, the port is unreachable as the user was not listening on the port since the machine did not make any DNS request

4

generate_dns_request:
src=10.9.0.5, dst=10.9.0.53, to create fake requests coming from 10.9.0.5 to its local-dns-server
sport=355753, dport=53, sport can be any number, so I reused it. dport has to be 53 since DNS uses it

generate_dns_reply:
used dig www.example.com +trace, to see the route taken by the query, found that the nameserver responsible for www.example.com is a.iana-servers.net, IP=199.43.135.53.

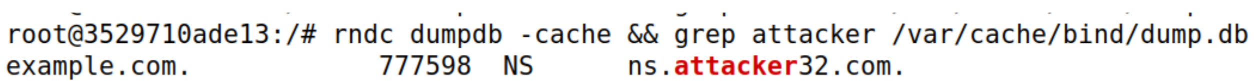


```
www.example.com.        86400   IN      A       93.184.216.34
www.example.com.        86400   IN      RRSIG   A 13 3 86400 20231028192921 202310071
22139 37939 example.com. xKH5bdt6acTLeoq5Ns8OUq23kg29LAxEPSRrk9AME91rJFmVGXwH/TTn RR7
Zd40h7wCA75GHFfYvLcEm+MuIvA==
;; Received 167 bytes from 199.43.135.53#53(a.iana-servers.net) in 211 ms
```

Constructing a response packet is the same as in task 3, we have to swap the src and dst port number as well as their IP in the request packet
So, src=199.43.135.53 (IP of nameserver), dst=10.9.0.53, to create fake dns response from the TLD to the local dns server
sport=53 (same reason as above), dport=33333 as given in the configuration file

To run the attack script attack.c, the python codes to generate dns request and response have to create binary files with the specific query details. After which, compile attack.c to an output file attack, which will take in the binary files and start flooding the user's local-dns-server.

To increase the success rate, we have to send more packets with different transaction ids. I implemented the attack to send all possible tid (16 bits = 65535) for the same response packet. Base code was taken from the guidelines section in the lab instructions.



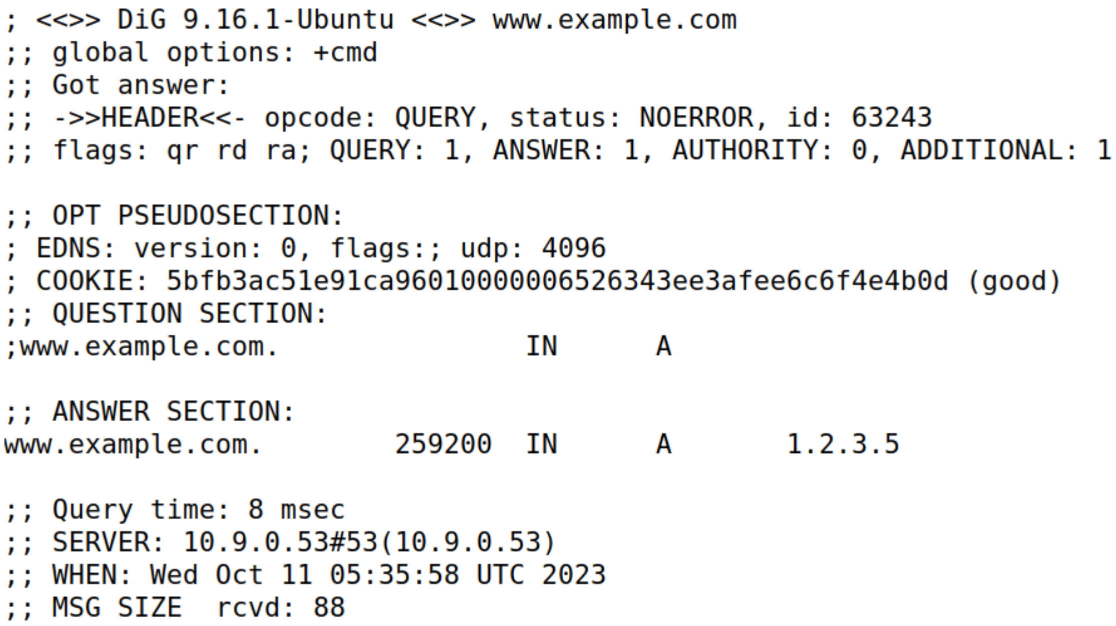
```
root@3529710ade13:/# rndc dumpdb -cache && grep attacker /var/cache/bind/dump.db
example.com.            777598  NS      ns.attacker32.com.
```

By running rndc dumpdb -cache && grep attacker /var/cache/bin/dump.db, we dumped the cache into dump.db file and extracted entries with the attacker substring. This output shows that our attack is successful.

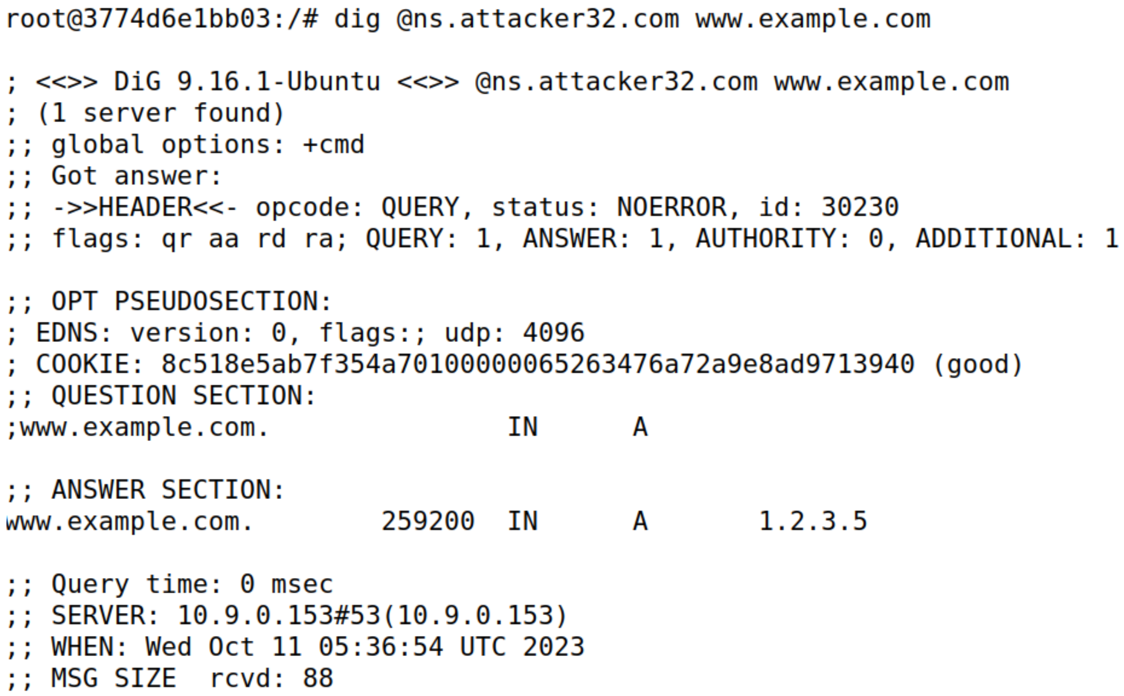
However, this attack will only work for records that are not in the DNS cache of the user's machine. Otherwise, we will have to wait for the records to expire after their TTL becomes 0.

5

The results of dig www.example.com and dig @ns.attacker32.com www.example.com are the same, which implies the attack was successful as the queries for example.com will go through the attacker's nameserver, and return a 1.2.3.5 IP in the answer section.



```
; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63243
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 5bfb3ac51e91ca96010000006526343ee3afee6c6f4e4b0d (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.2.3.5

;; Query time: 8 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Wed Oct 11 05:35:58 UTC 2023
;; MSG SIZE  rcvd: 88
```



```
root@3774d6e1bb03:/# dig @ns.attacker32.com www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> @ns.attacker32.com www.example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30230
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 8c518e5ab7f354a70100000065263476a72a9e8ad9713940 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       1.2.3.5

;; Query time: 0 msec
;; SERVER: 10.9.0.153#53(10.9.0.153)
;; WHEN: Wed Oct 11 05:36:54 UTC 2023
;; MSG SIZE  rcvd: 88
```