Memo DNS MITM Attacks

Vinci Nicolò

05 November 2021

1 Part 1

This section contains answers for question 1-6.

1.1 Q1: What is the IP address that server?

From dig output *SERVER: 10.1.1.3#53(10.1.1.3)*, so the IP address is *10.1.1.3*.

1.2 Q2: What was the status of the response?

From dig output *->>HEADER <<- opcode: QUERY, status: NOERROR, id: 16479*, so the status is *NOERROR*.

1.3 Q3: What is the reported IP address of www.google.com?

From dig output *www.google.com. 10 IN A 10.1.2.155*, so the IP address is *10.1.2.155*.

1.4 Q4: How long will the www.google.com IPv4 address be cached?

10 second from *Network Information* in *Assignment Instructions*.

1.5 Q5: What is the authoritative name server for google.com?

From dig output *google.com. 604800 IN NS ns.google.com.*, so the authoritative name server is *ns.google.com.*.

1.6 Q6: For each such server what is its IPv4 address?

From dig output *ns.google.com 10 IN A 10.1.2.3*, so the IP address is *10.1.2.3*.

2 Part 2

This section contains answers for question 1-4.

2.1 Q1: State the source MAC and IP addresses as well as destination MAC and IP addresses for a packet going from the client to the cache.

Client IP: 10.1.1.2
Client MAC: 00:11:43:d5:f4:c2
Cache IP: 10.1.1.3
Cache MAC: 00:04:23:ae:cc:7c

2.2 Q2: Does the packet travel through the attacker box?

No.

2.3 Q3: State the source MAC and IP addresses as well as destination MAC and IP addresses for a packet going from the cache to the authoritative server

Cache IP: 10.1.2.2
Cache MAC: 00:11:43:d5:f4:e9
Auth server IP: 10.1.2.3
Auth server MAC: 00:0e:0c:68:a7:11

2.4 Q4: Does the packet travel through the attacker box?

No.

3 Part 3

This section contains answers for question 1-7.

3.1 Q1: The command you used.

```
$ ettercap --text --iface eth0 --nosslmitm --nopromisc \
--only-mitm --mitm arp /10.1.2.2/// /10.1.2.3///
```

3.2 Q2: What each option in the command means.

--text : text interface.
--iface: interface.
--nosslmitm: disable SSL certificates forgery. Used to intercept https traffic.
--only-mitm: do not sniff, only perform MitM attack.
--nopromisc: disable sniff of all traffic in iface.
--mitm: which MitM attack to employ.

3.3 Q3: State the source MAC and IP addresses as well as destination MAC and IP addresses for a packet going from the cache to the authoritative server.

Cache IP: 10.1.2.2
Cache MAC: 00:04:23:ae:cc:32
Auth server IP: 10.1.2.3
Auth server MAC: 00:04:23:ae:cc:32

3.4 Q4: Does the packet travel through the attacker box?

Yes.

3.5 Q5: Does the packet travel through the attacker box?

They differ, because the attacker is poisoing both arp tables of cache and auth. The attacker is impersonating auth towards cache and cache towards auth. So, the attacker is in the middle between cache and auth.

3.6 Q6: The complete command (or steps in the GUI) used to have ettercap forge a DNS message and any necessary configuration files.

At the bottom of /etc/ettercap/etter.dns adds: *www.google.com A 10.1.2.4*. The command:

```
$ ettercap --text --iface eth0 --nosslmitm --nopromisc \
--plugin dns_spoof --mitm arp /10.1.2.2/// /10.1.2.3///
```

3.7 Q7: What malicious things could an attacker do by changing the IP address in a DNS response going to the client?

The attacker is able to redirect the client to an arbitrary IP address. For instance, every time the client would go to *www.google.com*, instead he will be redirected to an IP address chosen by the attacker. This IP address may be the attacker himself (10.1.2.4).

4 Part 4

This section contains answers for question 1-2.

4.1 Q1: The signed response obtained on the client machine.

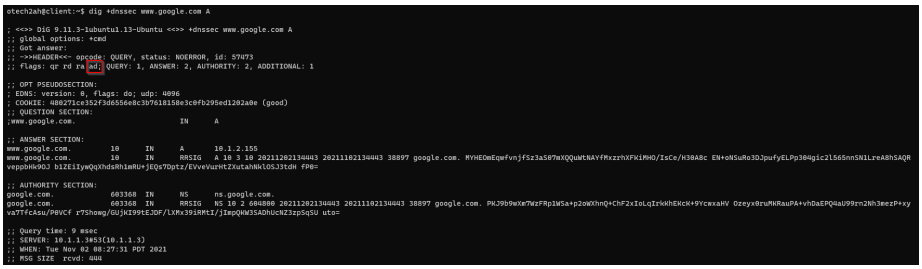

Figure 1: Dig on client machine

4.2 Q2: Detailed description of all the steps you took to implement DNSSEC. Make sure to list all commands you typed and all configuration changes you made.

All below commands need superuser privileges. On **auth** machine open */etc/bind/named.conf.options* file and add:

```
dnssec−enable yes;
dnssec−validation yes;
```

Also, modify *directory* in:

```
directory "/etc/bind";
```

In */etc/bind* folder generate the Zone Signing Key:

```
$ dnssec−keygen −a RSASHA512 −b 1024 −n ZONE google.com
```

And the Key Signing Key:

```
dnssec−keygen −f KSK −a RSASHA512 −b 2048 \
−n ZONE google.com
```

At the end of */etc/bind/google.com* file include ZSK and KSK:

```
$include Kgoogle.com.∗.key
$include Kgoogle.com.∗.key
```

In */etc/bind* folder sign the *google.com* file:

```
$ dnssec−signzone \
−3 $(head −c 1000 /dev/urandom | sha1sum | cut −b 1−16) \
google.com
```

In the */etc/bind/named.conf.local* modify the *file* in:

```
file: "/etc/bind/google.com.signed"
```

Restart bind:

```
service bind9 restart
```

On **cache** machine open /etc/bind/named.conf.options file and add:

```
dnssec−enable yes;
dnssec−validation yes;
```

Also, modify *directory* in:

```
directory "/etc/bind";
```

Create a new file called *nico.keys* in the folder */etc/bind* with the previous ZSK and KSK:

```
managed−keys {
    google.com. initial−key 257 3 10 "KSK";
    google.com. initial−key 256 3 10 "ZSK";
};
```

Modify the file */etc/bind/named.conf* adding:

```
include "/etc/bind/nico.keys";
```

Restart bind:

```
service bind9 restart
```