# CS 5153/5053 Network Security, Spring 2023

## Project 4: Local DNS Cache Poisoning

## Report

### Student: Austin Tyler Conn

Contents

[Link to Source Code 3](#_Toc131190281)

[Host Environment Used 3](#_Toc131190282)

[Docker Information 3](#_Toc131190283)

[Assumptions 3](#_Toc131190284)

[How do you setup the User machine and Server machine? 4](#_Toc131190285)

[User Machine (Task 1) 4](#_Toc131190286)

[Server Machine (Task 2) 5](#_Toc131190287)

[Disclaimer: 5](#_Toc131190288)

[How do you perform the attack in your VM? 11](#_Toc131190289)

[Screenshots of each step 18](#_Toc131190290)

[Was the attack successful? 19](#_Toc131190291)

Link to Source Code <https://github.com/austinc3030/dns_m11809075>

## Host Environment Used

Operating System: Ubuntu 20.04 LTS

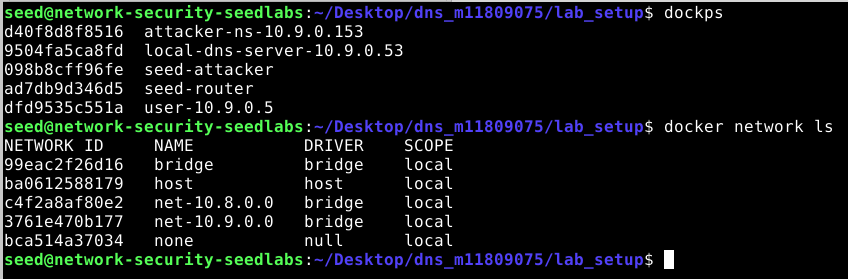


Hardware: Google Cloud E2 Instance

Links Used for Environment Setup:

* [seed-labs/seedvm-cloud.md at master · seed-labs/seed-labs (github.com)](https://github.com/seed-labs/seed-labs/blob/master/manuals/cloud/seedvm-cloud.md)
* [seed-labs/create\_vm\_gcp.md at master · seed-labs/seed-labs (github.com)](https://github.com/seed-labs/seed-labs/blob/master/manuals/cloud/create_vm_gcp.md)

## Docker Information



## Assumptions

* 1. User Machine (10.0.2.18) = user-10.9.0.5 (10.9.0.5)

Graphical user interface, text, application

Description automatically generated

* 1. Attacker (10.0.2.17) = seed-attacker (10.9.0.1)

A screenshot of a computer

Description automatically generated

* 1. Local DNS Server (10.0.2.16) = local-dns-server-10.9.0.53 (10.9.0.53)Graphical user interface, text, application

     Description automatically generated

## How do you setup the User machine and Server machine?

### User Machine (Task 1)

1. Look at the contents of */etc/resolv.conf*.

Graphical user interface, text, application

Description automatically generated

1. It appears that the SEED Labs Docker Image is already configured to use the local-dns-server-10.9.0.53 as the DNS server. Note that *resolvconf* is not installed in this image. Verify using the *dig* command that 10.9.0.53 is the DNS server in use.

Text

Description automatically generated

***Note:*** *as seen above, the server used by dig is 10.9.0.53, the IP address of local-dns-server-10.9.0.53*.

### Server Machine (Task 2)

Disclaimer: *Due to using SEED Labs Docker Containers, the server DOES need to have its /etc/resolv.conf file updated to use a DNS server OTHER than docker’s internal DNS server. Without doing this, the local-dns-server-10.9.0.53 will not reach out for queries, thus inhibiting the attacker being able to spoof the reply to local-dns-server-10.9.0.53.*

1. As mentioned in the disclaimer above, change the address in */etc/resolv.conf* to something other than docker’s internal DNS server.

Graphical user interface, text, application

Description automatically generated

1. Look at the contents of */etc/bind/named.conf*.

Text

Description automatically generated

1. Look at the contents of */etc/bind/named.conf.options*.

Text

Description automatically generated

1. Dump the DNS cache to the file specified by the *dump-file* line in step 2 (*/var/cache/bind/dump.db*).

Graphical user interface, text, application

Description automatically generated

1. Flush the DNS cache.

Graphical user interface, text, application

Description automatically generated

1. Turn off DNSSEC by modifying the */etc/bind/named.conf.options* file to comment out *dnssec-validation auto;* and add *dnssec-enable no;*.

Text

Description automatically generated

***Note:*** *this appears to already be configured in the SEED Labs Docker Images provided.*

1. From the seed-user, test that DNS is working properly using *dig*.

Text

Description automatically generated

1. Use Wireshark to verify a DNS query is made when running *dig* on seed-user.

Graphical user interface, application

Description automatically generated

## How do you perform the attack in your VM?

1. Start Wireshark monitoring traffic between seed-attacker, seed-user, and local-dns-server-10.9.0.53.

Graphical user interface, text, application, email

Description automatically generated

1. Flush DNS cache on local-dns-server-10.9.0.53.

Graphical user interface, text

Description automatically generated with medium confidence

1. Run *dig* for [*www.example.net*](http://www.example.net) on seed-user and note the output.

Text

Description automatically generated

1. Review Wireshark to see the conversation between seed-user and local-dns-server-10.9.0.53.

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. Flush DNS cache on local-dns-server-10.9.0.53.

Graphical user interface, text, application

Description automatically generated

1. Run *dns\_spoof.py* on seed-attacker.

Graphical user interface, text, application, website

Description automatically generated

1. Run *dig* for [www.example.net](http://www.example.net) on seed-user and note the output.

Text

Description automatically generated

1. Compare the outputs from steps 3 and 7. The output of step 7 should indicate that the IP addressed returned to seed-user is now the malicious IP.

The IP address returned prior to the attack was 93.184.216.34, the IP address returned after the attack is 10.9.0.1, the malicious IP of seed-attacker.

Graphical user interface, text

Description automatically generated

1. Review Wireshark to see the conversation between seed-user and local-dns-server-10.9.0.53. Note how seed-attacker is the sender of the reply to local-dns-server-10.9.0.53, which local-dns-server-10.9.0.53 then sends back to seed-user as the IP address for [www.example.net](http://www.example.net).

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

1. Stop the attack *dns\_spoof.py*.

Graphical user interface, text

Description automatically generated

1. Run *dig* on seed-user again for [www.example.net](http://www.example.net)

Text

Description automatically generated

**Note:** even without the attack running, our malicious IP address is still returned when seed-user requests a lookup for [www.example.net](http://www.example.net).

## Screenshots of each step

See screenshots shown with steps in “How do you setup the User machine and Server machine?” and “How do you perform the attack in your VM?”

## Was the attack successful?

### Include screenshots to show the attack is successful and can render an incorrect IP on both the User machine and Server machine

Yes, the attack was successful. When the attack is running and local-dns-server receives a DNS Query for [www.example.net](http://www.example.net), local-dns-server sends a query to it’s upstream DNS server. The attack sees this and replies to local-dns-server acting as the upstream DNS server and provides a result for the query. The result contains our malicious IP address of seed-attacker which local-dns-server then stores in it’s cache. It then sends this response to seed-user as it was the one who requested the lookup. The attack can then be stopped and can still be seen that the malicious IP is returned when a lookup is requested.

See screenshots in steps 8, 9, and 11 of “How do you perform the attack in your VM?”