LAB 2

**INSTALLING AND CONFIGURING BIND9 FOR DVWA DOMAIN NAME RESOLUTION ON KALI LINUX**

**Student ID: 24560025**

**Student Name: Dao Minh Chau**

**Objective**

Students will learn to:

Install and configure BIND9 as a private DNS server on Kali Linux.

Set up DNS resolution for the DVWA website to allow access via a domain name .

Configure forward and reverse DNS zones to map domain name to the web server’s IP .

Test and verify DNS resolution for client access to DVWA.

**Network Configuration**

Web Server & DNS Server: 192.168.241.131 (hosts DVWA and BIND9).

Database Server: 192.168.241.132.

Domain Name: dvwa.local (resolves to 192.168.241.131).

Client Network: 192.168.241.135 (clients query the DNS server and access DVWA).

Note: Explanation is highlighted in yellow.

**Part 1: Web Server Setup (VM1: 192.168.241.131)**

**Step 1: Update the System**

Command:

sudo apt update && sudo apt upgrade -y: update and upgrade kali to the newest files

Function: Ensures VM1 is up-to-date with the latest security patches and package versions, preventing compatibility issues during LAMP stack installation.

**Step 2: Install Apache, PHP, and Dependencies**

Command:

sudo apt install apache2 php php-mysql php-gd git -y

where:

apache2: Apache web server to host DVWA.

php: PHP interpreter for DVWA’s dynamic content.

php-mysql: PHP extension for database connectivity.

php-gd: PHP extension for image processing (used by DVWA for CAPTCHAs).

git: Tool to clone the DVWA repository.

-y: Confirms installation prompts.

Note: MariaDB is not installed on VM1, as the database server is on VM2 (192.168.241.132).

Function: Sets up the web server environment on VM1 to host DVWA, excluding the database component.

**Step 3: Enable and Start Apache**

Commands:

sudo systemctl enable apache2

sudo systemctl start apache2

sudo systemctl status apache2

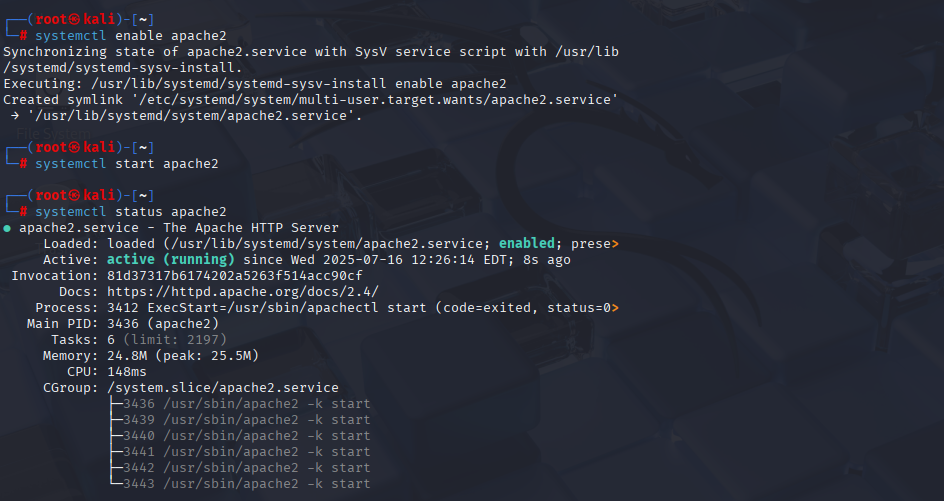
Explanation:

sudo systemctl enable apache2: Configures Apache to start automatically on system boot.

sudo systemctl start apache2: Starts the Apache service immediately.

sudo systemctl status apache2: Verifies Apache is running (Active: active (running)).

Function: Activates the Apache web server on VM1 and ensures it persists across reboots, allowing DVWA to be served.



**Step 4: Clone the DVWA Repository**

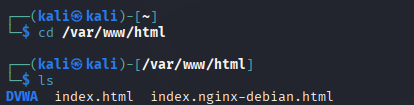
Commands:

cd /var/www/html: change directory to Apache’s default web root directory.

sudo git clone https://github.com/digininja/DVWA.git: Downloads the DVWA source code from GitHub into /var/www/html/DVWA.

sudo chown -R www-data:www-data /var/www/html/DVWA: set ownership to the www-data user and group

sudo chmod -R 755 /var/www/html/DVWA: Grants the owner full access (read, write, execute) and others read/execute access for security and functionality.



**Step 5: Configure PHP**

Commands:

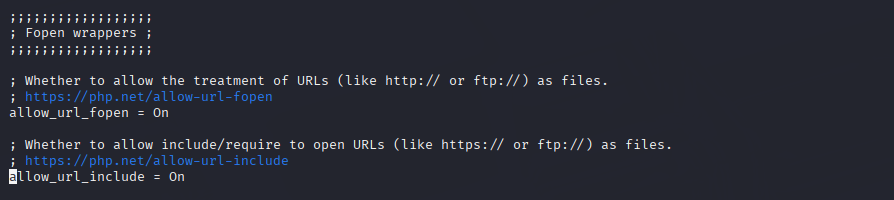
sudo nano /etc/php/\*/apache2/php.ini

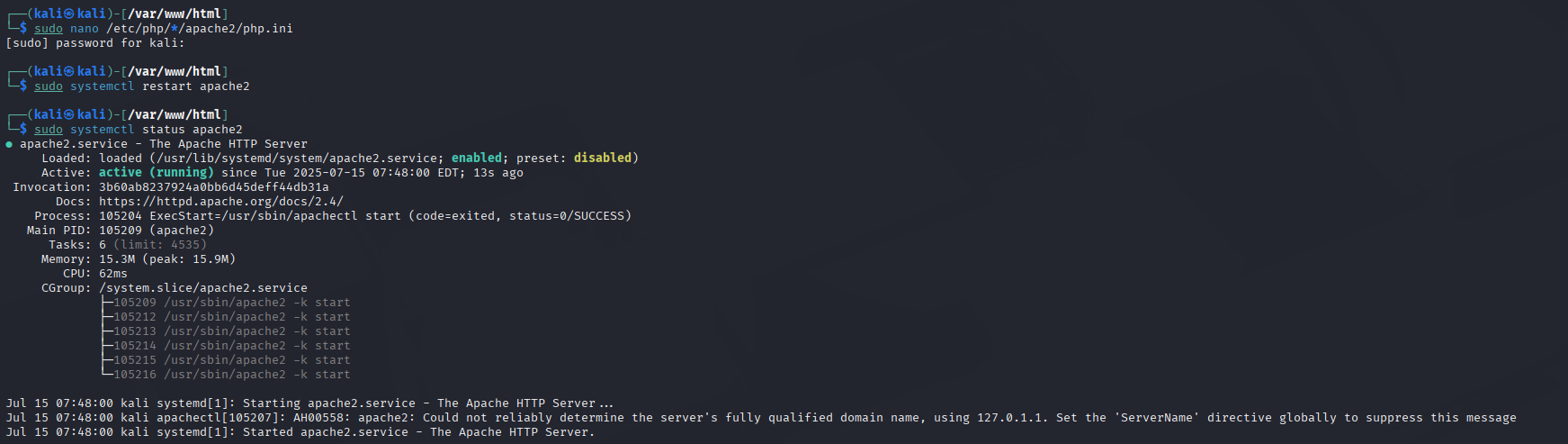
Set:

allow\_url\_include = On: Enables PHP to include files from URLs, required for DVWA’s file inclusion vulnerabilities.

Then:

sudo systemctl restart apache2: restart the apache2 to update the configuration.





**Part 2: Database Server Setup (VM2: 192.168.241.132)**

Step 6: Update the System

Command:

sudo apt update && sudo apt install mariadb-server -y

Explanation:

sudo apt update: Refreshes the package index on VM2.

sudo apt install mariadb-server -y: Installs MariaDB, the database server, on VM2.

-y: Confirms installation prompts.

Function: Ensures VM2 has the latest package information and installs MariaDB to store DVWA’s data.

**Step 7: Enable and Start MariaDB**

Commands:

sudo systemctl enable mariadb: activate the database

sudo systemctl start mariadb: start the database

sudo systemctl status mariadb: check database status.

Function: Activates the MariaDB service on VM2, ensuring the database server is operational.



**Step 8: Configure the DVWA Database**

Commands:

sudo mariadb

Execute:

DROP DATABASE IF EXISTS dvwa;: delete any existed database named dvwa

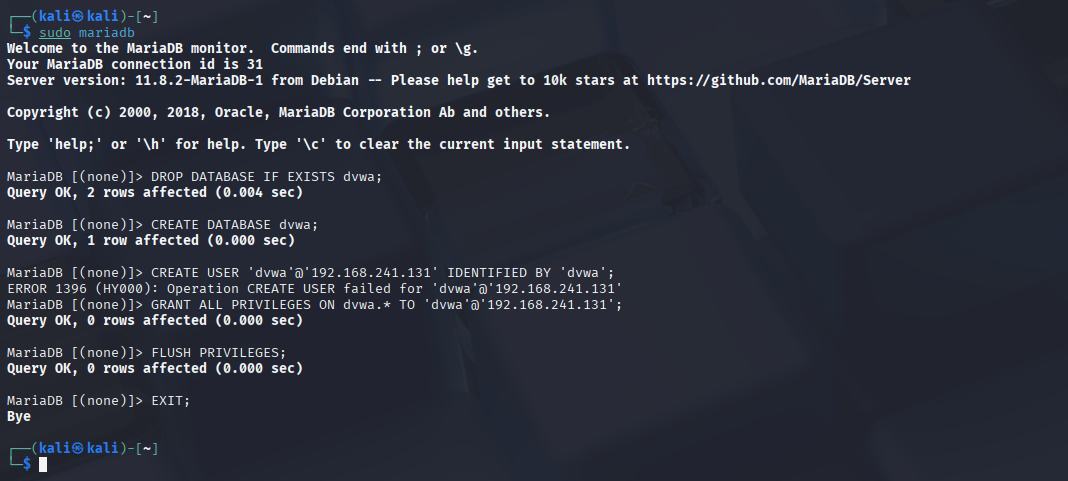
CREATE DATABASE dvwa;: create a new database

CREATE USER 'dvwa'@'192.168.241.131' IDENTIFIED BY 'dvwa';: create a user identified by database dvwa

GRANT ALL PRIVILEGES ON dvwa.\* TO 'dvwa'@'192.168.241.131';: set permission to the user

FLUSH PRIVILEGES;

EXIT;



**Step 9: Allow Remote Database Connections**

Commands:

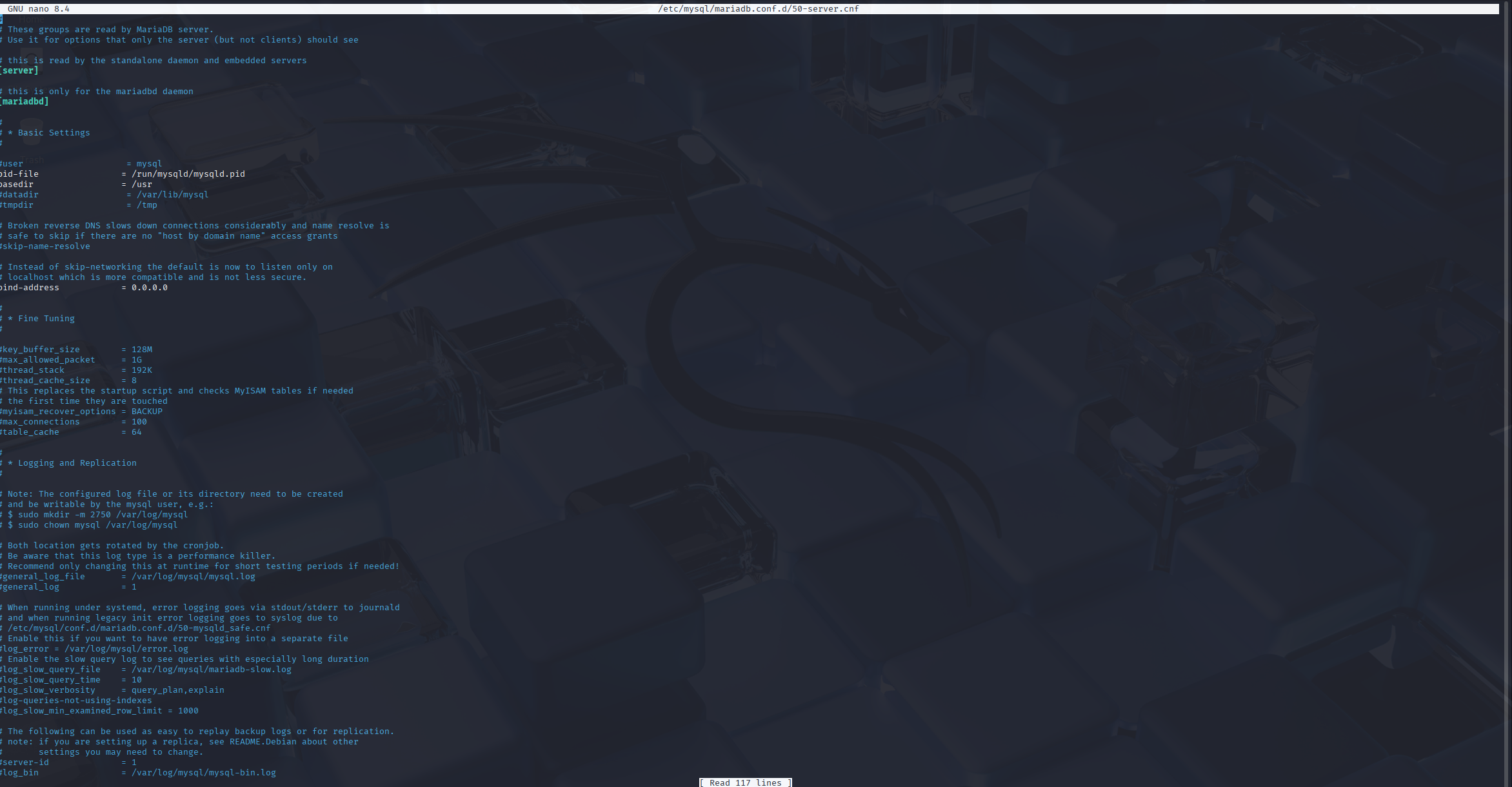
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf : edit the file

Set:

bind-address = 0.0.0.0: Allows MariaDB to accept connections from any IP, enabling VM1 (192.168.241.131) to connect.

Then:

sudo systemctl restart mariadb: Restarts MariaDB to apply the configuration.





**Part 3: Configure DVWA (VM1: 192.168.241.131)**

**Step 10: Configure DVWA**

Commands:

cd /var/www/html/DVWA/config: Navigates to the DVWA configuration directory on VM1.

sudo cp config.inc.php.dist config.inc.php: copy the default configuration file to a working file

sudo nano config.inc.php: edit the file

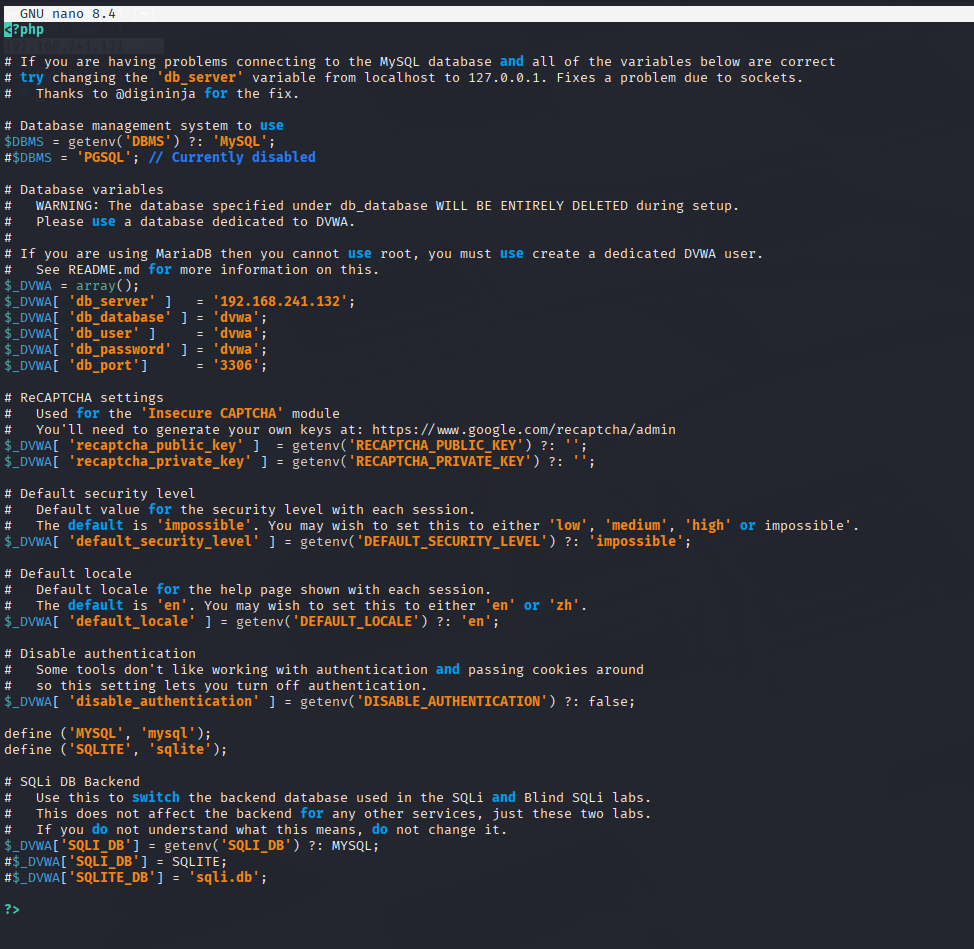
Set:

$\_DVWA['db\_server'] = '192.168.241.132';: set DVWA to the database server on VM2

$\_DVWA['db\_user'] = 'dvwa';: set the database user

$\_DVWA['db\_password'] = 'dvwa';:Set the user’s password

$\_DVWA['db\_database'] = 'dvwa';: names the database



**Part 4: DNS Server Setup (VM1: 192.168.241.131)**

**Step 11: Install BIND9 and Required Tools**

Command:

sudo apt update && sudo apt install bind9 bind9utils dnsutils -y

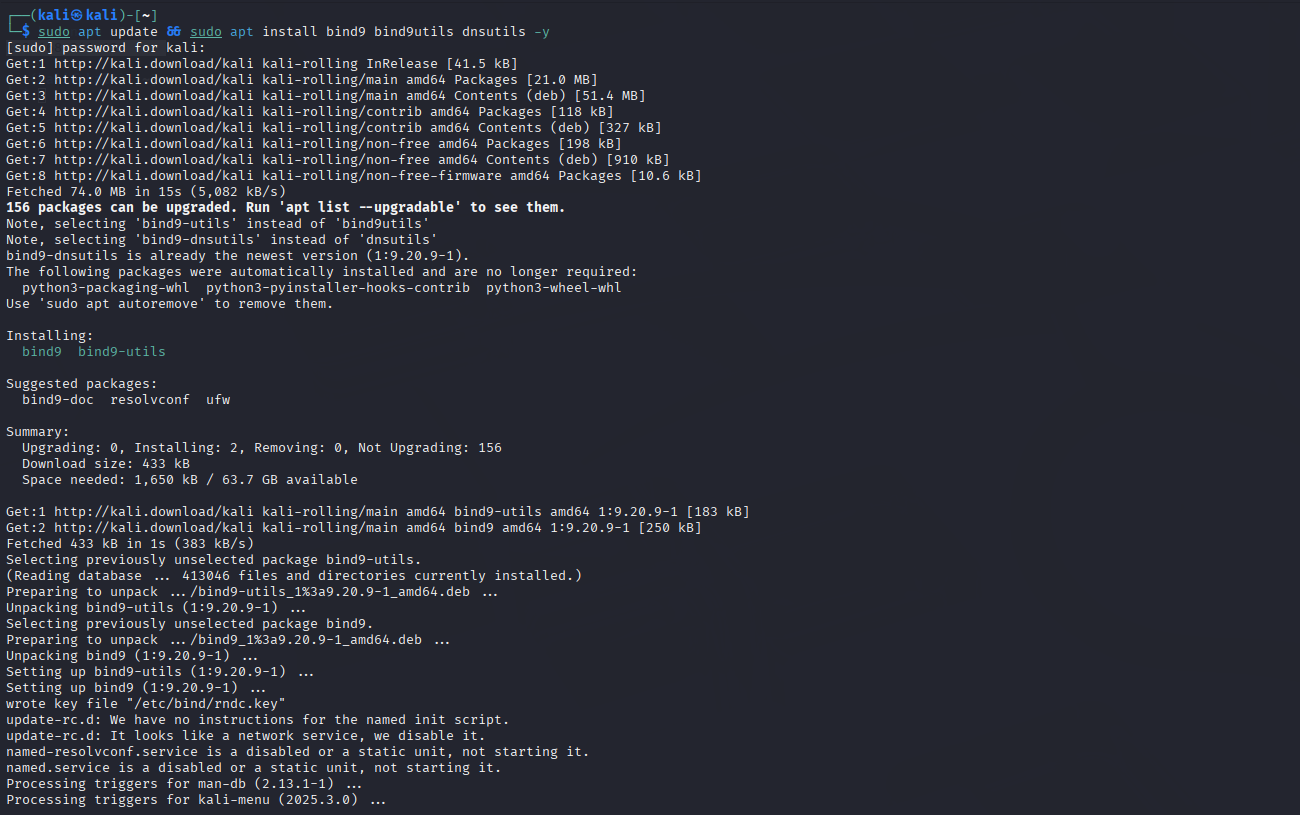
The command above installs:

bind9: DNS server software.

bind9utils: Tools for managing BIND9 (e.g., rndc).

dnsutils: Utilities like dig and nslookup for testing.

Function: Installs BIND9 and tools on VM1 to enable DNS resolution for dvwa.local.



**Step 12: Configure BIND9 Options**

Commands:

sudo nano /etc/bind/named.conf.options: edit the file

Add or modify:

acl trusted {

192.168.241.0/24; // Allow queries from this network

localhost;

};

options {

directory "/var/cache/bind";

allow-query { trusted; };

forwarders {

8.8.8.8;

8.8.4.4;

};

recursion yes;

dnssec-validation auto;

listen-on-v6 { none; }; // Disable IPv6 for simplicity

};

acl trusted: Allows DNS queries from the 192.168.241.0/24 LAN and localhost.

directory "/var/cache/bind": Sets the directory for cache and zone files.

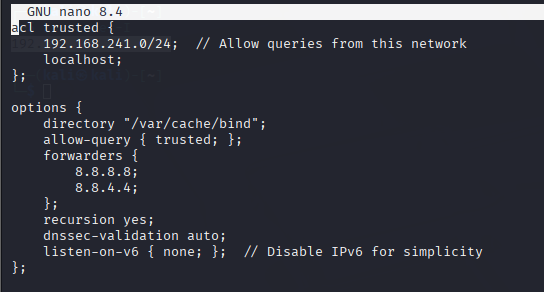
allow-query { trusted; }: Restricts queries to the trusted ACL.

forwarders { 8.8.8.8; 8.8.4.4; }: Uses Google’s DNS for external queries.

recursion yes: Enables recursive queries.

dnssec-validation auto: Enables DNSSEC validation.

listen-on-v6 { none; }: Disables IPv6 to match the LAN’s IPv4 setup.



**Step 13: Define DNS Zones**

Commands:

sudo nano /etc/bind/named.conf.local: edit the file

Add:

zone "dvwa.local" {

type master;

file "/etc/bind/zones/db.dvwa.local";

};

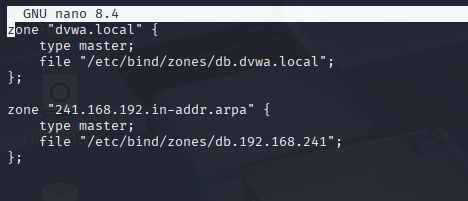
zone "241.168.192.in-addr.arpa" {

type master;

file "/etc/bind/zones/db.192.168.241";

};

The content of the file allows reversed lookup so when user enter the ip address, it automatically trace to the domain name.



**Step 14: Create Zone Files Directory**

Commands:

sudo mkdir -p /etc/bind/zones: Creates a directory for zone files on VM1.

sudo chown bind:bind /etc/bind/zones: Sets ownership to the bind user and group.

sudo chmod 755 /etc/bind/zones: Grants full access to bind and read/execute to others.

**Step 15: Create Forward Zone File**

Commands:

sudo nano /etc/bind/zones/db.dvwa.local

Add:

$TTL 604800

@ IN SOA ns1.dvwa.local. admin.dvwa.local. (

2025071401 ; Serial

3600 ; Refresh

1800 ; Retry

604800 ; Expire

86400 ) ; Negative Cache TTL

;

@ IN NS ns1.dvwa.local.

ns1 IN A 192.168.241.131

dvwa.local. IN A 192.168.241.131

www IN CNAME dvwa.local.

sudo nano /etc/bind/zones/db.dvwa.local: Creates the forward zone file on VM1.

$TTL 604800: Sets the Time-To-Live (7 days).

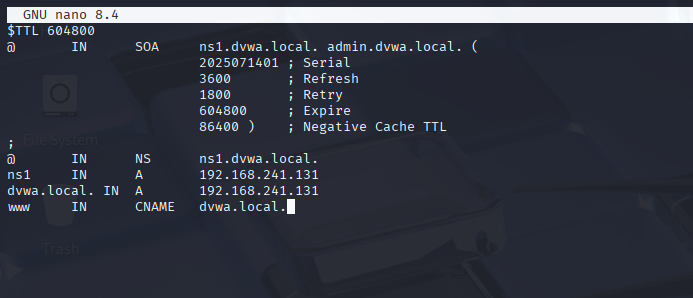
SOA: Defines the nameserver (ns1.dvwa.local) and admin email (admin.dvwa.local).

NS: Specifies ns1.dvwa.local as the nameserver.

A: Maps ns1.dvwa.local and dvwa.local to 192.168.241.131.

CNAME: Aliases www.dvwa.local to dvwa.local.

Function: Configures DNS to resolve dvwa.local and www.dvwa.local to VM1’s IP.



**Step 16: Create Reverse Zone File**

Commands:

sudo nano /etc/bind/zones/db.192.168.241

Add:

$TTL 604800

@ IN SOA ns1.dvwa.local. admin.dvwa.local. (

2025071401 ; Serial

3600 ; Refresh

1800 ; Retry

604800 ; Expire

86400 ) ; Negative Cache TTL

;

@ IN NS ns1.dvwa.local.

131 IN PTR dvwa.local.

sudo nano /etc/bind/zones/db.192.168.241: Creates the reverse zone file on VM1.

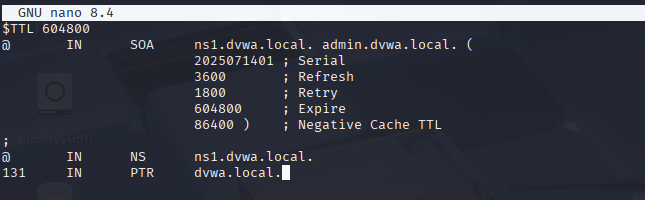
$TTL 604800: Sets the TTL.

SOA: Defines authoritative information.

NS: Specifies the nameserver.

PTR: Maps 192.168.241.131 to dvwa.local.

Function: Enables reverse DNS resolution for 192.168.241.131.



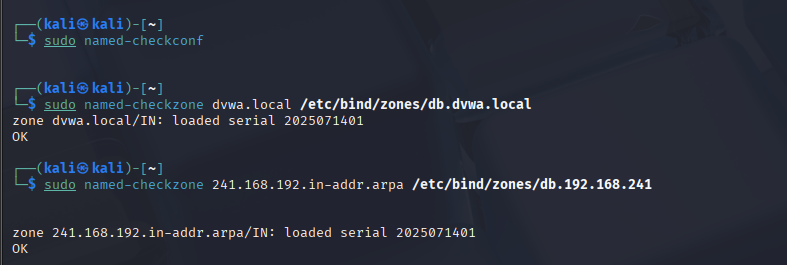
**Step 17: Verify BIND9 Configuration**

Commands:

sudo named-checkconf: Checks BIND9 configuration file syntax on VM1.

sudo named-checkzone dvwa.local ...: Verifies the forward zone file.

sudo named-checkzone 241.168.192.in-addr.arpa ...: Verifies the reverse zone file.



**Step 18: Start and Enable BIND9**

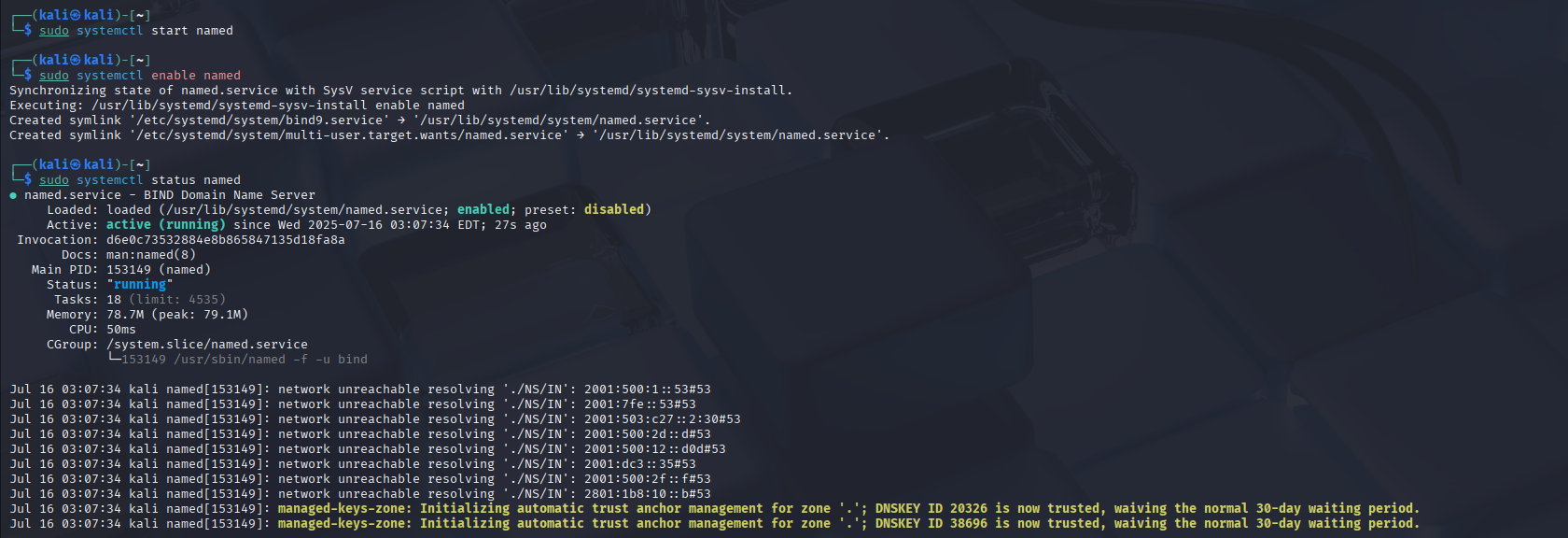
Commands:

sudo systemctl start bind9: Starts BIND9 on VM1.

sudo systemctl enable bind9: Configures BIND9 to start on boot.

sudo systemctl status bind9: Confirms BIND9 is running.

\*NOTE: In some old versions of kali the name of bind9.service will be registered as named.service like mine.



**Part 5: Verify Setup**

**Step 19: Verify Web and Database Setup**

Commands:

On VM1, open a browser or use curl:

curl http://192.168.241.131/DVWA/setup.php

Access in a browser:

http://192.168.241.131/DVWA/setup.php

Click Create / Reset Database.

Log in at:

http://192.168.241.131/DVWA/login.php

Use:

Username: admin

Password: password

Explanation:

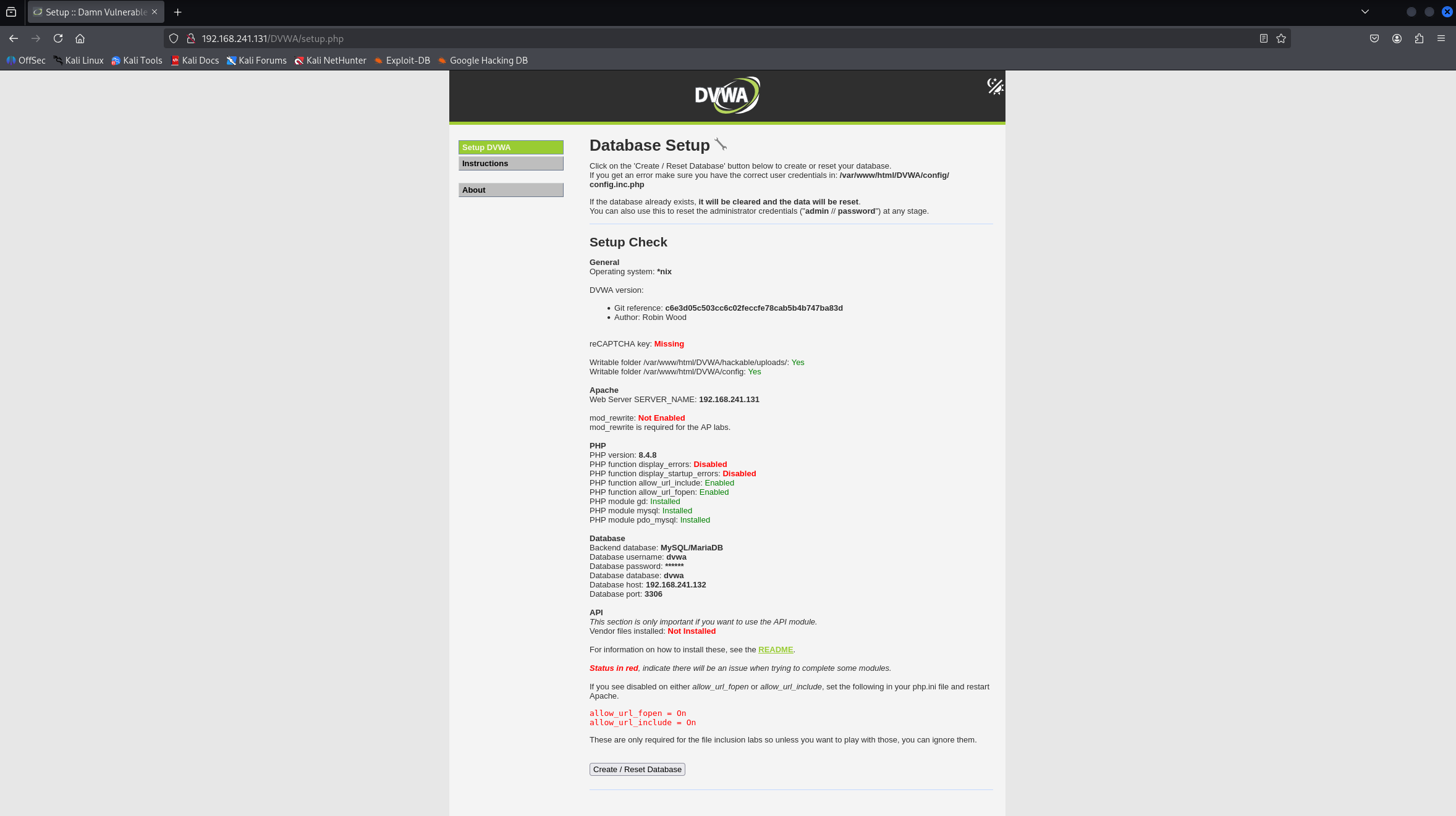
curl http://192.168.241.131/DVWA/setup.php: Tests web server access on VM1.

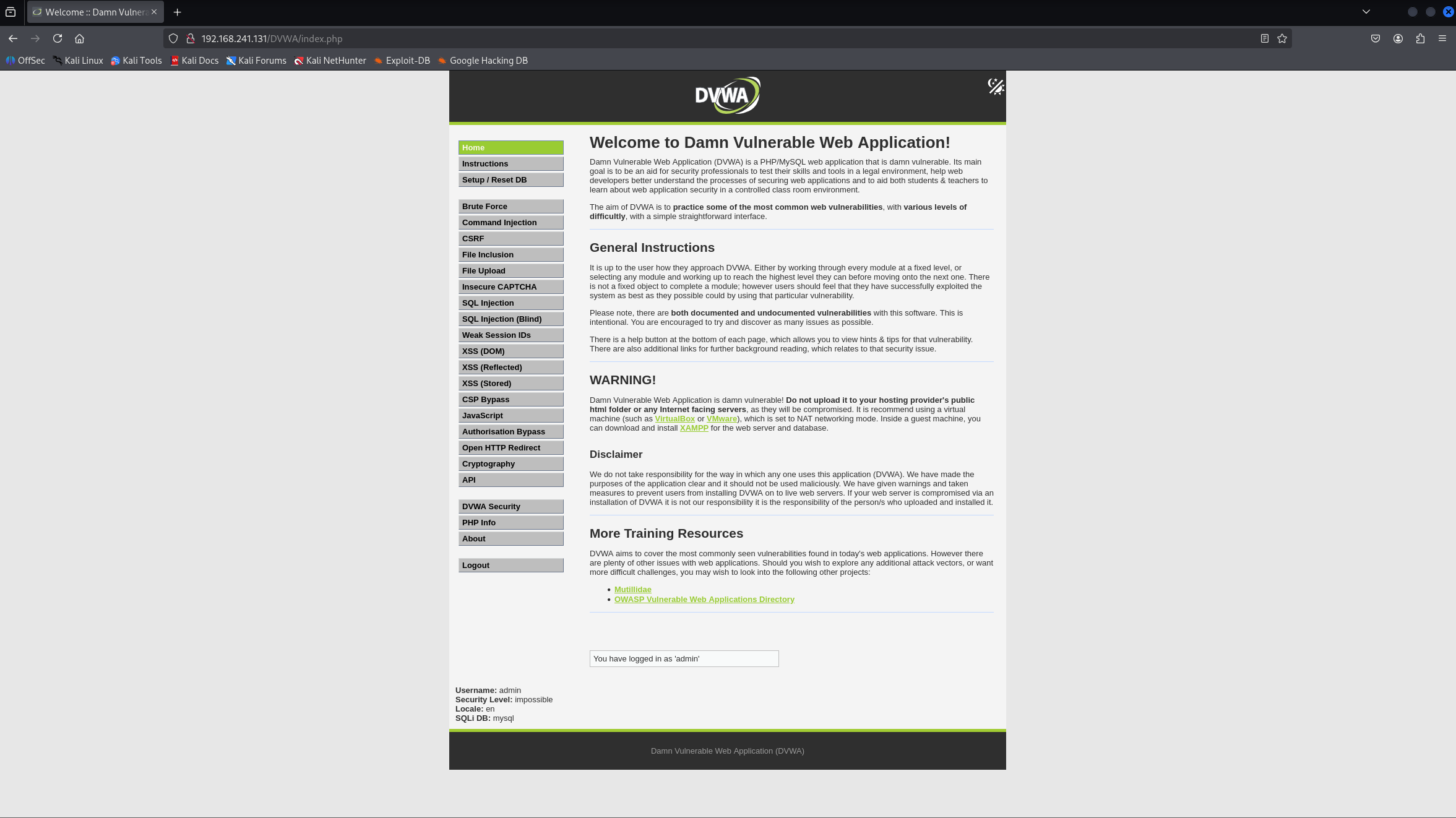
Accessing setup.php: Loads DVWA’s setup page, which connects to VM2’s database.

Clicking Create / Reset Database: Initializes database tables on VM2.

Logging in: Verifies DVWA functionality with default credentials.

Function: Confirms that the web server (VM1) and database server (VM2) are correctly configured and DVWA is operational.





**Step 20: Configure Client DNS Settings**

Commands (on a client, e.g., host machine):

Explanation:

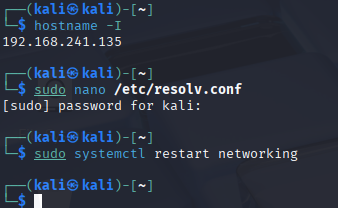
sudo nano /etc/resolv.conf: Configures the client to use VM1’s DNS server.

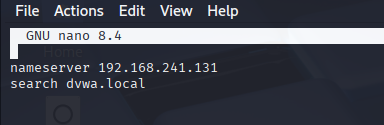
nameserver 192.168.241.131: Directs DNS queries to VM1.

search dvwa.local: Appends dvwa.local to unqualified names.

sudo systemctl restart networking: Applies network changes.

Function: Enables clients on the LAN to resolve dvwa.local using VM1’s DNS server.





**Step 21: Test DNS Resolution**

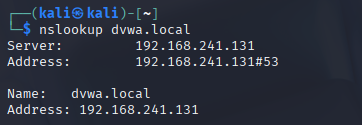
Commands (on a client or VM1):

dig dvwa.local: Queries the A record (should return 192.168.241.131).

dig www.dvwa.local: Queries the CNAME record (resolves to dvwa.local).

dig -x 192.168.241.131: Performs a reverse lookup (returns dvwa.local).

nslookup dvwa.local: Confirms resolution.



**Step 22: Access DVWA via Domain Name**

Commands:

Accessing http://dvwa.local/DVWA/setup.php: Resolves dvwa.local to 192.168.241.131 via VM1’s DNS server.

Clicking Create / Reset Database: Initializes the database on VM2.

Logging in: Confirms DVWA is accessible via the domain name.

Setting security to low: Configures DVWA for vulnerability testing.

Function: Verifies domain-based access to DVWA over the LAN, fulfilling the lab’s objective.

