AdNU-CEVAS Web Server Setup Documentation

Wilfredo Paulo A. PEREZ III & Ana Nicole P. OLEA

Prerequisites

- Ensure a **linux** machine. A physical or a virtual machine may do it.
- **Kali linux distro** is highly recommended as we delve on the server penetration test.
- Secure administrative access to the resources.
- ***THIS DOCUMENTATION ASSUMES YOU ALREADY HAVE DYNAMIC HTML (via PHP)*** This documentation only covers the **Installation, Configuration,** and **Penetration Testing.**
- Secure a local network for two physical/virtual machines.

Install Necessary Components (LAMP Stack)

- Installing Apache

Unset

```
//sudo: grants temporary admin rights and privileges
//apt: package tool. Manages preferred software dependencies
//systemctl: interact with a service

$ sudo apt update && sudo apt upgrade -y
$ sudo apt install apache2 -y
$ sudo systemctl enable apache2 //Configure auto start on system boot
$ sudo systemctl start apache2
$ sudo systemctl status apache2
```

- Installing PHP

Unset

```
$ sudo apt install php libapache2-mod-php php-mysql
```

- Check the Installed Database Version

Unset

```
$ mysql --version //Necessary to prompt whether you are using MySQL or MariaDB
$ sudo systemctl enable mariadb //Enable auto start on system boot
$ sudo systemctl start mariadb
$ sudo systemctl status mariadb
$ sudo mariadb-secure-installation //Prompt security script to configure root
passwords and remove unwanted defaults
```

- Configure PHP pages

Inside the web server, there are pages configured in PHP logic. Ideally, you can have static HTML pages first to picture out the whole flow of the web you are designing.

All these pages should have a PHP backend enclosed with <?php ?>

Unset

```

//the PHP logic below will include the basic POST request method syntax and
data process

//start with the session. This will allow storing and accessing session
variables for the user.

        session_start();
        $_SESSION['user_id'] = $user['id'];

//connect to the database (to be configured after this)

        require 'db.php';

//you can opt to intialize variables the next line

        $variable = ""

//include basic form requirement to process data submitted by the user
//$_POST['email'] and $_POST['password'] hold the respective values entered by
the user.

        if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $email = $_POST['email'];
        $password = $_POST['password'];
```

```
//$pdo to interact to the database. Defined in db.php.
//$stmt to prevent most sql injections. Uses prepared sql statements
//prepares a SQL query to select all columns from the users table where the
email matches the provided value.

    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");

//execute prepare statement. $email value is passed as an array ([$email]).

    $stmt->execute([$email]);

//fetches result if $user found a match. false if otherwise

    $user = $stmt->fetch();

//if $user exists, password is checked to match the hatched password.

    if ($user && password_verify($password, $user['password'])) {

//secures session for the user. Prevents fixation

    session_regenerate_id(true);

//heads to the preferred page after successful log in

    header('Location: ValidatePHP.php');
```

- Configure database

Unset

```
//crucial during the initial setup of MySQL to prevent unauthorized access.

$ sudo mysql -u root

    SELECT, user, host, plugin FROM mysql.user;
    ALTER USER 'root'@'localhost' IDENTIFIED BY 'root';
    ALTER USER 'root'@'localhost' IDENTIFIED VIA mysql_native_password USING
    PASSWORD('root');
    FLUSH PRIVILEGES; //reload privileges
    EXIT;
```

```
$ sudo mysql -u root -p //You can opt to use: sudo mariadb -u root -p. Use
password you set

//Primary Key: ensures uniqueness.
//Unique Key: ensures that no two users can have the same email address.
//NOT NULL: ensures not empty
//Foreign Key: references the id_number field in the users table.
//ON DELETE CASCADE: If a user is deleted from the users table, all their
certificates will also be deleted.

       CREATE DATABASE adnu_cevas;
       USE adnu_cevas;
       CREATE TABLE users (
              id INT AUTO_INCREMENT PRIMARY KEY,
              full_name VARCHAR(255) NOT NULL,
              email VARCHAR(255) NOT NULL UNIQUE,
              password VARCHAR(255) NOT NULL,
              id_number VARCHAR(50) NOT NULL UNIQUE,
              created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
       );

       CREATE TABLE certificates (
              id INT AUTO_INCREMENT PRIMARY KEY,
              student_id VARCHAR(50) NOT NULL,
              certificate_name VARCHAR(255) NOT NULL,
              date_issued DATE NOT NULL,
              hash VARCHAR(255) NOT NULL,
              FOREIGN KEY (student_id) REFERENCES users(id_number) ON DELETE
              CASCADE
       );
       SHOW TABLES;
       EXIT;
```

- Allow one-way access from PHP files to the database. Disable javascript manipulation **Replace project paths with your own path**

Unset

```
$ sudo cd /home/rhino/Desktop/AdNU_CEVAS
$ nano

//inside file
/*
       <?php
```

```
      $host = "localhost";
      $dbname = "adnu_cevas";
      $username = "root";
      $password = "password_you_set";
                                              //*IMPORTANT*
      ini_set('session.cookie_httponly', 1); //Prevent access to session
      cookies via JavaScript
      ini_set('session.cookie_secure', 1);  //Ensure cookies are sent over
      HTTPS
      ini_set('session.use_strict_mode', 1); //Reject uninitialized session IDs

      try {
            $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username,
            $password);
            $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
      } catch (PDOException $e) {
            die("Database connection failed: " . $e->getMessage());
      }
      ?>
*/
//save file as db.php
```

- Disable Web server URL manipulation ***OPTIONAL*** **Replace project paths with your own path.**

Unset

```
//make .htaccess file

$ sudo cd /home/rhino/Desktop/AdNU_CEVAS
& nano

//inside file
/*
      <FilesMatch "\.(php|inc)$">
            Deny from all
      </FilesMatch>

      <Files "index.php">      //default page of apache
            Allow from all
      </Files>

      <Files "LogoutPHP.php">     //in my project's case. DO NOT COPY
            Allow from all
```

```
       </Files>

       <Files "LoginPHP.php">     //in my project's case. DO NOT COPY
              Allow from all
       </Files>

       <Files "SignupPHP.php">     //in my project's case. DO NOT COPY
              Allow from all
       </Files>
*/
//save as .htaccess
```

- Configure the PHP Application, SymLinks, and .htaccess **Replace project paths with your own path.**

Unset

```
$ sudo rm /var/www/html/index.php   //removes default apache page (to be
changed with preferred default page i.e., homepage or splashpage)

//symlink allows immediate data changes from specific project folder to the
apache data folder

$ sudo ln -s /home/rhino/Desktop/AdNU_CEVAS/* /var/www/html
$ ls /var/www/html

//chown: change ownership
//chmod: change permission
//-R: recursive application
//www-data:www-data: owner and group of the web server
//755: permission levels,
       //7 (Owner): Full permissions (read, write, execute).
       //5 (Group): Read and execute only.
       //5 (Others): Read and execute only.


$ sudo chown -R www-data:www-data /home/rhino/Desktop/AdNU_CEVAS
$ sudo chmod -R 755 /home/rhino/Desktop/AdNU_CEVAS
$ sudo chmod 755 /home/rhino/Desktop
$ sudo chmod 755 /home/rhino
$ nano /etc/apache2/apache2.conf

//inside this file, FIND <Directory /var/www/html>
```

```
//replace "AllowOverride none" to "AllowOverride All"
//replace "Require all denied" to "Replace all granted"

$ sudo a2enmod rewrite //enables .htaccess
$ sudo systemctl restart apache2
```

Penetration Testing

- Determine the IP address of the targeted server

Unset

```
$ ip addr show
```

- Allow Apache through Firewall

Unset

```
//SERVER SIDE
//install ufw: user-friendly front end for managing firewall rules

$ sudo apt install ufw -y
$ sudo ufw allow 'Apache Full'
$ sudo nano /etc/apache2/ports.conf

//add line "Listen 8000"

$ sudo ufw enable
$ sudo ufw status
$ sudo ufw allow 8000

//test via Kali Linux browser, enter http://<server_IP>
```

- Installing Nmap, Using Nmap

Unset

```
//ATTACKER SIDE

$ sudo apt update
$ sudo apt install nmap -y
$ nmap <server_IP> //basic scan
$ nmap -sV --script=vuln <server_IP> //full vulnerability scan
```

- Installing Nessus, Using Nessus

Unset

```
//ATTACKER SIDE
//download Nessus from the Tenable website
//https://www.tenable.com/products/nessus

$ sudo apt install -y libgomp1
$ sudo dpkg -i Nessus-<version>.deb
$ sudo systemctl start nessusd
$ sudo systemctl enable nessusd

//access Nessus at https://<IP>:8834 and configure the scan for your server.
```

- SQL Injection

Unset

```
//use SQL injection payloads to test vulnerabilities in form inputs
//basic tests to be inputted in a form field:
       //' OR 1=1 --
       //'' OR '1'='1' --
       //admin' --
       //   '' UNION SELECT id, full_name, email, password, NULL, id_number FROM users --
       //' AND 1=CONVERT(int, @@version) --
       //' AND 1=1 --  (Valid query)
       //' AND 1=2 --  (Invalid query)

//for advanced testing, SQLMAP is recommended

$ sqlmap -u "http://<IP>/LoginPHP.php" --data="email=test@example.com&password=test" --dump

$ sqlmap -u "http://<IP>/LoginPHP.php" --data="full_name=test&password=test" --dump

***//ideally, common sql injections should not work on this web server because the page scripts use prepared statements ($stmt = $pdo->prepare(...))***
```

Adding Web Server security

- Use **HTTPS**: Install an SSL certificate with **Let’s Encrypt**

```
Unset
//using https instead of http adds a layer of security by encrypting exchange
of data between the client and the server

$ sudo apt install certbot python3-certbot-apache -y
$ sudo certbot --apache
```

- Harden PHP

```
Unset
$ nano /etc/php/<version>/apache2/php.ini
//inside file
/*
       expose_php = Off
       display_errors = Off
*/
```

- Limit directory access

```
Unset
//edit .htaccess file

//add:
/*
       <Directory /var/www/html/images>     //in my case
              Order Allow,Deny
              Deny from all
       </Directory>
*/

$ sudo systemctl restart apache2
```

- Limit Permissions

```
Unset
//set right of (others) user to NONE

$ sudo chmod -R 750 /var/www/html
```

- Consistent use of SQL prepare statements

Unset

```
$stmt = $pdo->prepare("SELECT * FROM users WHERE username = ? AND password = ?");
```

—END—