**Foundational Projects: 1. Cyber Security**

### **Project 1: Configuring a Secure Web Server on a Local Machine**

**Objective:** Set up and secure a web server on your local machine.

**Tasks:**

1. **Install a Web Server**: Install Apache on your local machine.

2. **Configure HTTPS**: Set up SSL/TLS using self-signed certificates.

3. **Implement Firewall Rules**: Use a firewall like UFW (Uncomplicated Firewall) to control traffic.

4. **Secure Configuration**: Apply best practices for securing the web server configuration.

**Tools:** Apache OpenSSL, UFW

### **Project 2: Conducting a Vulnerability Assessment Using Free Tools**

**Objective:** Perform a basic vulnerability assessment of your local network or a local web application. **Tasks:**

1. **Install Vulnerability Scanners**: Install tools like OpenVAS or Nikto on your local machine.

2. **Scan for Vulnerabilities**: Run scans on your local network or a web application.

3. **Analyze Results**: Review the scan results and identify potential vulnerabilities.

4. **Mitigate Vulnerabilities**: Implement mitigation measures for the identified vulnerabilities.

**Tools:** OpenVAS, Nikto

**Project 1: Configuring a Secure Web Server on a Local Machine**

***Introduction***

This lab involves setting up and securing a web server on a local machine. There is increased reliance on web applications and services, and therefore it's crucial to understand the fundamentals of web security.

This lab guides through installing a web server, configuring HTTPS for secure communication, implementing firewall rules to control traffic, and applying best practices to secure the web server configuration.

## **Objectives**

· **Install a Web Server**: Learn how to install Apache or Nginx on your local machine.

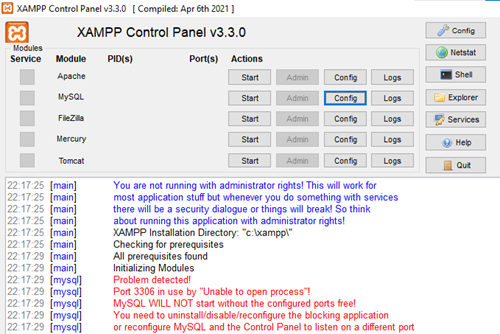
· **Configure HTTPS**: Set up SSL/TLS using self-signed certificates to secure communications.

· **Implement firewall rules**: Use a firewall like UFW (Uncomplicated Firewall) to control inbound and outbound traffic.

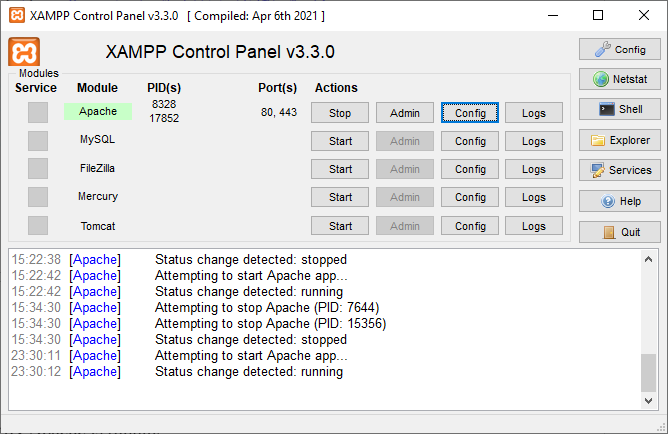
· **Apply Security Best Practices**: Understand and implement best practices to secure the web server configuration

**Step 1**: **Install a Web Server (Apache) for Windows:**

· a. Download and install XAMPP, which includes Apache.



b. Verify Apache is running

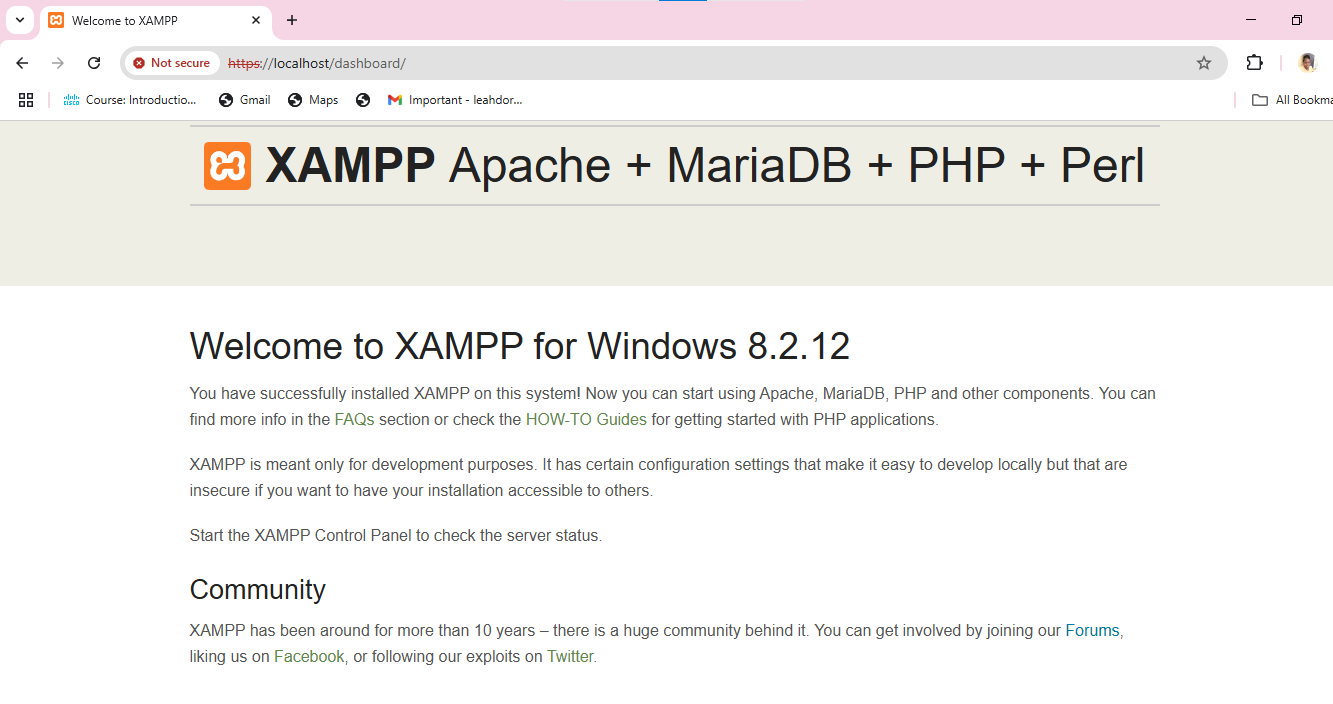


**Apache**: If Apache is running and you can access http://localhost/ (or https://localhost/ if using HTTPS) without errors, then Apache is successfully installed and configured.

**SSL Certificate**: If you can access your site over https://localhost/ and the SSL certificate is in place (even if it's a self-signed certificate), then the SSL setup is also successful.

To confirm:

* Open a web browser and navigate to http://localhost/. You should see the default Apache welcome page.



Apache Installation and Configuration

* **Apache Version**: 2.4.58
* **XAMPP Version**: 8.2.12
* **Operating System**: Windows 10 Enterprise 64-bit
* **Document Root**: C:/xampp/htdocs
* **Modules Enabled**:
  + mod\_rewrite
  + mod\_ssl

**Changes Made**:

* Disabled directory listing
* **SSL Certificate File**: C:/xampp/apache/conf/ssl.crt/server.crt
* **SSL Certificate Key File**: C:/xampp/apache/conf/ssl.key/server.key
* **Server Name**: localhost:443

### 

### **Security Settings**

* **Directory Permissions**:
  + Web root directory (htdocs): Proper permissions set to prevent unauthorized access.

### **Testing and Verification**

* **Test URL**:
  + http://localhost/: Apache default welcome page
  + https://localhost/: Self-signed SSL certificate warning, but accessible

### **Troubleshooting**

* **Issue**: Port 80 conflict
* **Solution**: Changed Apache port to 8080 in httpd.conf

**Summary**

**Installed Apache and XAMPP**.

**Configured SSL** with a self-signed certificate.

**Verified Apache is running** by accessing http://localhost/ and https://localhost/.

**Step 2.**  **Configure HTTPS: Set up SSL/TLS using self-signed certificates.**

Setting up SSL/TLS helps to secure the website using self-signed

certificates, ensuring all communications between the client and server

are encrypted.

### : Generate a Self-Signed Certificate

### **Completed**:

* + Generated a self-signed certificate using OpenSSL.
  + Command used:

openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout C:\xampp\apache\conf\ssl.key\server.key -out C:\xampp\apache\conf\ssl.crt\server.crt

### : Update Apache SSL Configuration

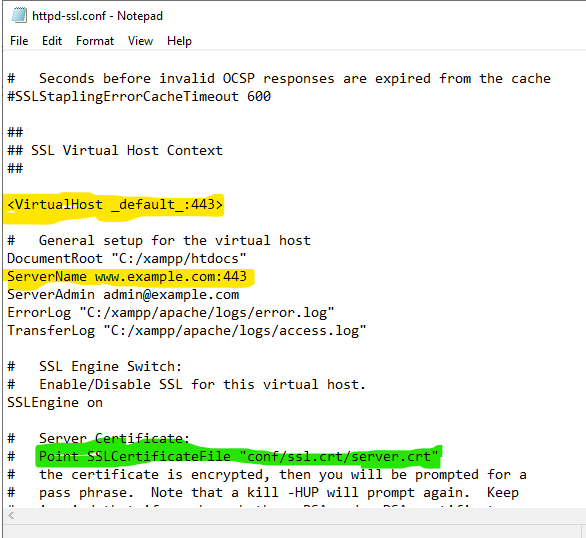
**Open** httpd-ssl.conf:

* + Location: C:\xampp\apache\conf\extra\httpd-ssl.conf
  + Edited the SSL configuration to point to the new certificate and key files:

SSLCertificateFile "conf/ssl.crt/server.crt"

SSLCertificateKeyFile "conf/ssl.key/server.key"

* + Updated the server name to match your domain name (e.g., localhost:443):



### **Summary**

With these steps, the website is now accessible over HTTPS using a self-signed certificate, ensuring secure communication between the client and server.

## Step 3: Implement Firewall Rules

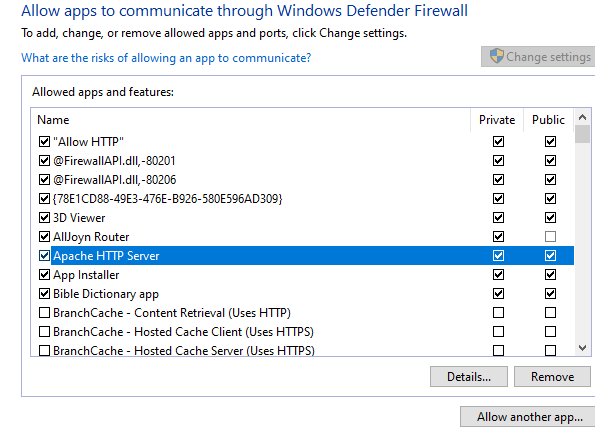
## Set up and configure Firewall to control traffic to and from your server, ensuring that only authorized connections are allowed.

### Open Windows Firewall

* Open Control Panel:
  + Press Win + X and select "Control Panel".
* Select "System and Security".
* Select "Windows Defender Firewall".

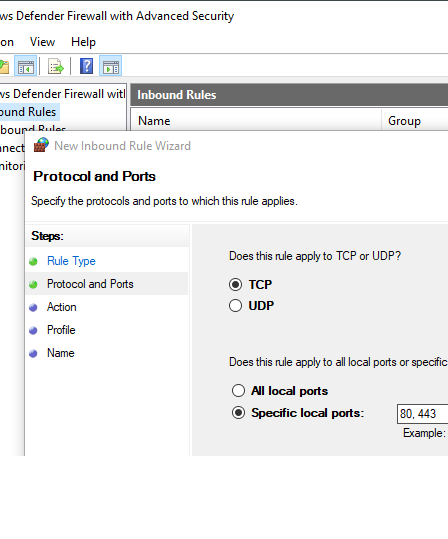
#### Allow an App or Feature Through Windows Defender Firewall

* Click on "Allow an app or feature through Windows Defender Firewall."
* Click on "Change settings" (you might need administrative privileges).
* Find "Apache HTTP Server" in the list and ensure it’s allowed through both private and public networks.
* Click "OK"

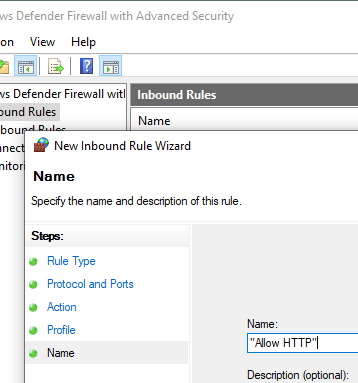


### b. Create New Inbound Rules

* Click on "Advanced settings" on the left side.
* Select "Inbound Rules" from the left panel.
* Click on "New Rule..." on the right panel.
* Select "Port" and click "Next"
* Select "TCP" and specify the port(s) (e.g., 80 for HTTP, 443 for HTTPS).



* Select "Allow the connection" and click "Next".
* Choose when the rule applies (e.g., Domain, Private, Public) and click "Next".
* Name the rule (e.g., "Allow HTTP") and click "Finish".



**Summary**

By configuring the Windows firewall, you enhance the security of your server by controlling traffic to and from your server. It ensures that only authorized connections and services are allowed.

## **Step 4: Secure Configuration**

Apply best practices to secure the configuration of your web server, ensuring it is protected against common vulnerabilities and threats.

### Steps and Configurations

#### Disable Directory Listing

* **Purpose**: Prevent users from seeing the contents of directories on your web server.
* **Configuration**: Open the httpd.conf file and add or modify the following line:

*Options -Indexes*

#### Disable Unnecessary Modules

* Purpose: Reduce the attack surface by disabling modules that are not needed.
* Configuration: Open the httpd.conf file and comment out or remove the lines that load unnecessary modules. e.g by adding #

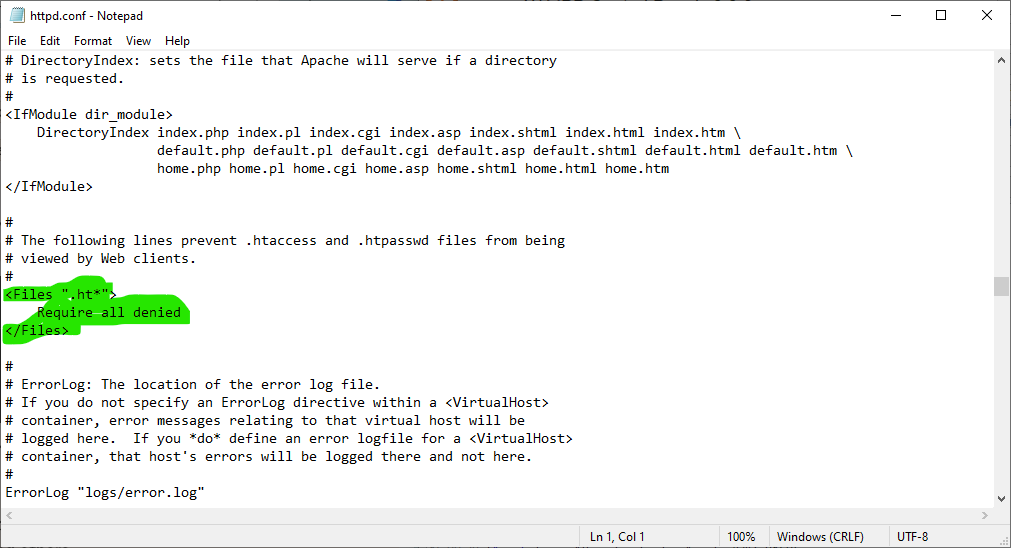
#*LoadModule status\_module modules/mod\_status.so*

*#LoadModule info\_module modules/mod\_info.so*

#### Restrict Access to Sensitive Files

* **Purpose**: Prevent unauthorized access to sensitive files, such as configuration files.

**Configuration**: Add the following directives to your httpd.conf or .htaccess file



#### Configure HTTPS and Enforce SSL

* **Purpose**: Ensure all data transmitted between the client and server is encrypted.
* **Configuration**: Update your httpd-ssl.conf file to include the following:  
  apache  
  SSLEngine on

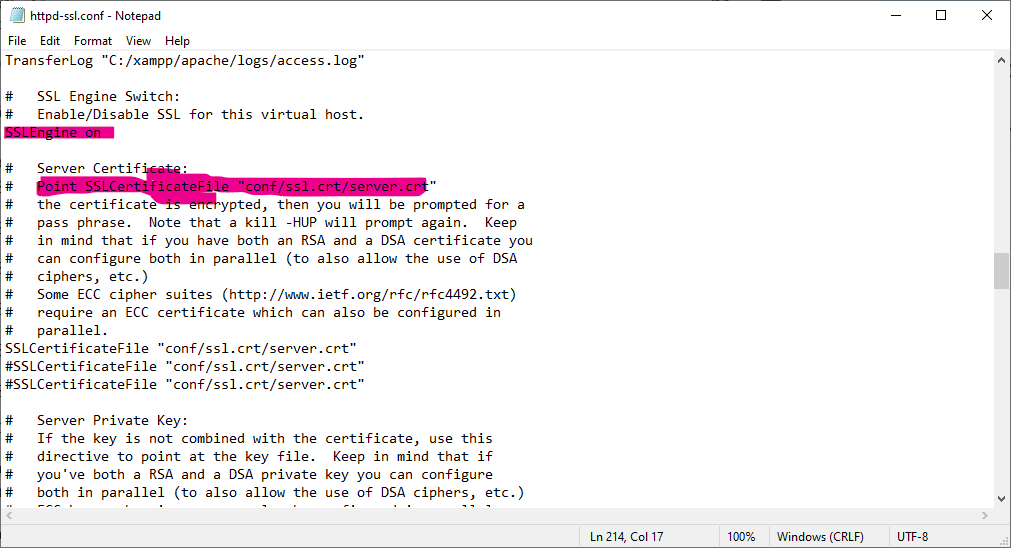
SSLCertificateFile "conf/ssl.crt/server.crt"

SSLCertificateKeyFile "conf/ssl.key/server.key"

SSLProtocol all -SSLv2 -SSLv3

SSLCipherSuite HIGH:!aNULL:!MD5

SSLHonorCipherOrder on



#### Limit Request Size

* **Purpose**: Prevent denial-of-service (DoS) attacks by limiting the size of client requests.

**Configuration**: Add the following directive to your httpd.conf file:  
apache  
LimitRequestBody 10485760

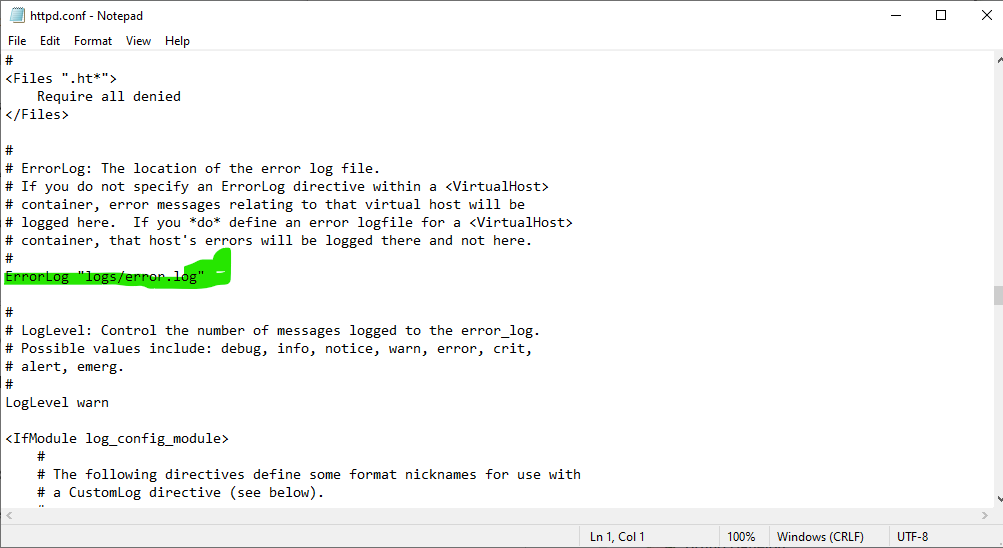
#### Configure Logging and Monitoring

* **Purpose**: Keep track of access and errors for auditing and troubleshooting.

**Configuration**: Ensure logging is enabled in your httpd.conf file:  
apache  
LogLevel warn

CustomLog "logs/access.log" combined

ErrorLog "logs/error.log"



### **Summary**

By applying these best practices for securing your web server configuration, you can significantly enhance the security of your website and protect it against common vulnerabilities and threats.