Lab 5: Apache Web Server TLS/SSL Security Implementation

Imroj Hassan Shafi
Registration: 2020831050

November 20, 2025

Abstract

This laboratory documentation provides a comprehensive guide to implementing Transport Layer Security (TLS) on Apache web servers. The solution covers Certificate Authority establishment, SSL certificate generation, server configuration, and security validation procedures. All commands and configurations have been tested and verified for production-ready deployment.

Contents

1 Introduction **3**
1.1 Project Scope . . . 3
1.2 Learning Objectives . . . 3
1.3 Prerequisites . . . 3

2 Environment Preparation **3**
2.1 Workspace Setup . . . 3
2.2 OpenSSL Configuration . . . 3
2.3 CA Directory Structure . . . 4

3 Task 1: Certificate Authority Establishment **5**
3.1 Root CA Key Generation . . . 5
3.2 Root Certificate Creation . . . 5

4 Task 2: Server Certificate Generation **5**
4.1 example.com Certificate . . . 5
4.1.1 Private Key Generation . . . 5
4.1.2 Certificate Signing Request (CSR) . . . 6
4.1.3 Certificate Signing . . . 6
4.2 webserverlab.com Certificate . . . 6

5 Task 3: TLS Server Testing **6**
5.1 OpenSSL Test Server . . . 6
5.2 Browser Testing Procedure . . . 7
5.3 Firefox Certificate Import . . . 7

6 Task 4: Apache HTTPS Configuration **7**
6.1 SSL Module Activation 7
6.2 Virtual Host Configuration 7
6.2.1 example.com HTTPS Configuration 7
6.2.2 HTTP to HTTPS Redirection 8
6.3 Site Activation 8
6.4 Repeat for webserverlab.com 9

7 Verification and Testing **9**
7.1 SSL Certificate Verification 9
7.2 Apache Configuration Testing 9

8 Assessment Checkpoints **9**
8.1 Checkpoint 1: OpenSSL Server Validation (5 marks) 9
8.2 Checkpoint 2: Secondary Domain Testing (5 marks) 9
8.3 Checkpoint 3: Apache HTTPS Configuration (5 marks) 10
8.4 Checkpoint 4: Multi-Domain Deployment (5 marks) 10

9 Troubleshooting Guide **10**
9.1 Common Issues and Solutions 10
9.2 Log Analysis 10

10 Conclusion **11**
10.1 Implementation Summary 11
10.2 Security Best Practices Implemented 11
10.3 Further Enhancements 11

1 Introduction

1.1 Project Scope

This laboratory exercise focuses on implementing secure web communications through TLS/SSL encryption. The implementation covers the complete certificate lifecycle management from Certificate Authority creation to Apache web server configuration.

1.2 Learning Objectives

- Establish a private Certificate Authority (CA) infrastructure
- Generate and manage X.509 digital certificates
- Configure Apache for HTTPS with custom certificates
- Implement proper security practices for web server encryption
- Validate TLS configuration and troubleshoot common issues

1.3 Prerequisites

- Linux environment (Ubuntu 20.04+ / Debian 11+ recommended)
- OpenSSL package installed (`openssl`)
- Apache web server (`apache2`)
- Administrative privileges (`sudo` access)
- Basic understanding of public key infrastructure (PKI)

2 Environment Preparation

2.1 Workspace Setup

Listing 1: Initial Workspace Configuration

```
# Create dedicated working directory
mkdir -p ~/lab5_ca
cd ~/lab5_ca

# Verify OpenSSL installation
openssl version
```

2.2 OpenSSL Configuration

Create a custom OpenSSL configuration file (`openssl.cnf`):

Listing 2: OpenSSL Configuration File

```
[ ca ]
default_ca = CA_default

[ CA_default ]
dir               = ./demoCA
certs             = $dir/certs
```

```
crl_dir           = $dir/crl
new_certs_dir     = $dir/newcerts
database          = $dir/index.txt
serial            = $dir/serial
RANDFILE          = $dir/private/.rand
private_key       = $dir/private/ca.key
certificate       = $dir/cacert.pem
default_md        = sha256
preserve          = no
policy            = policy_any
email_in_dn       = no
name_opt          = ca_default
cert_opt          = ca_default
default_days      = 375
copy_extensions   = copy

[ policy_any ]
countryName             = optional
stateOrProvinceName     = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ req ]
default_bits       = 2048
distinguished_name = req_distinguished_name
string_mask        = utf8only
default_md         = sha256

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
stateOrProvinceName             = State or Province Name (full name)
localityName                    = Locality Name (city, district)
0.organizationName              = Organization Name (company)
organizationalUnitName          = Organizational Unit Name (department)
commonName                      = Common Name (server FQDN or YOUR name)
emailAddress                    = Email Address
```

2.3 CA Directory Structure

Listing 3: Certificate Authority Directory Setup

```
# Create CA directory hierarchy
mkdir -p demoCA/{certs,crl,newcerts,private}
chmod 700 demoCA/private

# Initialize CA database files
touch demoCA/index.txt
echo 1000 > demoCA/serial

# Verify directory structure
tree demoCA/
```

3 Task 1: Certificate Authority Establishment

3.1 Root CA Key Generation

Listing 4: Root Certificate Authority Creation

```
# Generate CA private key with passphrase protection
openssl genrsa -aes256 -out demoCA/private/ca.key 4096

# Secure key permissions
chmod 400 demoCA/private/ca.key
```

3.2 Root Certificate Creation

Listing 5: Self-Signed Root Certificate Generation

```
# Create self-signed root certificate (valid for 10 years)
openssl req -new -x509 -days 3650 -sha256 \
    -key demoCA/private/ca.key \
    -out demoCA/cacert.pem \
    -config openssl.cnf

# Verify root certificate
openssl x509 -in demoCA/cacert.pem -text -noout
```

Expected Prompts:

- Country Name (e.g., US, BD)
- State/Province (e.g., Dhaka)
- Locality (e.g., Dhaka City)
- Organization (e.g., Lab5-CA)
- Organizational Unit (e.g., Security)
- Common Name (e.g., Lab5 Root CA)
- Email Address (optional)

4 Task 2: Server Certificate Generation

4.1 example.com Certificate

4.1.1 Private Key Generation

Listing 6: Server Private Key Generation

```
# Generate server private key with passphrase
openssl genrsa -aes256 -out example.com.key 2048

# Optional: Remove passphrase for server use (use with caution)
openssl rsa -in example.com.key -out example.com.key.nopass

# Secure key permissions
chmod 400 example.com.key*
```

4.1.2 Certificate Signing Request (CSR)

Listing 7: Certificate Signing Request Creation

```
# Generate CSR for example.com
openssl req -new -key example.com.key \
    -out example.com.csr \
    -config openssl.cnf
```

Important: Set Common Name to `example.com`

4.1.3 Certificate Signing

Listing 8: Certificate Signing Process

```
# Sign the CSR with root CA
openssl ca -in example.com.csr \
    -out example.com.crt \
    -cert demoCA/cacert.pem \
    -keyfile demoCA/private/ca.key \
    -config openssl.cnf \
    -days 375

# Verify signed certificate
openssl x509 -in example.com.crt -text -noout
```

4.2 webserverlab.com Certificate

Repeat the same process for the second domain:

Listing 9: Second Domain Certificate Generation

```
# Key generation
openssl genrsa -aes256 -out webserverlab.com.key 2048

# CSR generation
openssl req -new -key webserverlab.com.key \
    -out webserverlab.com.csr \
    -config openssl.cnf

# Certificate signing
openssl ca -in webserverlab.com.csr \
    -out webserverlab.com.crt \
    -cert demoCA/cacert.pem \
    -keyfile demoCA/private/ca.key \
    -config openssl.cnf \
    -days 375
```

5 Task 3: TLS Server Testing

5.1 OpenSSL Test Server

Listing 10: OpenSSL TLS Test Server

```
# Combine key and certificate for testing
cat example.com.key example.com.crt > example.com.pem
```

```
# Launch TLS test server on port 4433
openssl s_server -cert example.com.pem \
    -key example.com.key \
    -accept 4433 \
    -www \
    -HTTP
```

5.2 Browser Testing Procedure

1. Add `127.0.0.1 example.com` to `/etc/hosts`
2. Access `https://example.com:4433/` in browser
3. Observe security warning (untrusted certificate)
4. Import `demoCA/cacert.pem` as trusted root certificate
5. Refresh browser to verify trusted connection

5.3 Firefox Certificate Import

- Navigate to: `Preferences` → `Privacy & Security` → `Certificates` → `View Certificates`
- Select `Authorities` tab → `Import`
- Choose `demoCA/cacert.pem`
- Check `Trust this CA to identify websites`
- Click `OK` to confirm

6 Task 4: Apache HTTPS Configuration

6.1 SSL Module Activation

Listing 11: Apache SSL Module Configuration

```
# Enable SSL module
sudo a2enmod ssl

# Enable headers module for security enhancements
sudo a2enmod headers

# Restart Apache to load modules
sudo systemctl restart apache2
```

6.2 Virtual Host Configuration

6.2.1 example.com HTTPS Configuration

Create `/etc/apache2/sites-available/example.com-ssl.conf`:

Listing 12: Apache SSL Virtual Host Configuration

```
<VirtualHost *:443>
    ServerAdmin admin@example.com
    ServerName example.com
    ServerAlias www.example.com
    DocumentRoot /var/www/example.com/html

    # SSL Engine Configuration
    SSLEngine on
    SSLCertificateFile /home/student/lab5_ca/example.com.crt
    SSLCertificateKeyFile /home/student/lab5_ca/example.com.key

    # SSL Protocol Configuration
    SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
    SSLCipherSuite ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
        SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384
    SSLHonorCipherOrder off
    SSLSessionTickets off

    # Security Headers
    Header always set Strict-Transport-Security "max-age=63072000;
        includeSubDomains; preload"
    Header always set X-Content-Type-Options nosniff
    Header always set X-Frame-Options DENY
    Header always set X-XSS-Protection "1; mode=block"

    # Logging
    ErrorLog ${APACHE_LOG_DIR}/example.com_ssl_error.log
    CustomLog ${APACHE_LOG_DIR}/example.com_ssl_access.log combined
</VirtualHost>
```

6.2.2 HTTP to HTTPS Redirection

Configure HTTP virtual host to redirect to HTTPS:

Listing 13: HTTP to HTTPS Redirection

```
<VirtualHost *:80>
    ServerName example.com
    ServerAlias www.example.com

    # Permanent redirect to HTTPS
    Redirect permanent / https://example.com/
</VirtualHost>
```

6.3 Site Activation

Listing 14: Site Configuration Activation

```
# Enable SSL site configuration
sudo a2ensite example.com-ssl.conf

# Test configuration syntax
sudo apache2ctl configtest

# Apply configuration changes
sudo systemctl reload apache2
```

```
9
10 # Verify Apache status
11 sudo systemctl status apache2
```

6.4 Repeat for webserverlab.com

Repeat the same configuration process for `webserverlab.com` using its respective certificate and key files.

7 Verification and Testing

7.1 SSL Certificate Verification

Listing 15: Certificate Chain Verification

```
1 # Verify certificate chain
2 openssl verify -CAfile demoCA/cacert.pem example.com.crt
3
4 # Check certificate details
5 openssl x509 -in example.com.crt -text -noout
6
7 # Test SSL connection
8 openssl s_client -connect example.com:443 -servername example.com
```

7.2 Apache Configuration Testing

Listing 16: Apache Configuration Validation

```
1 # Test configuration syntax
2 sudo apache2ctl configtest
3
4 # Check loaded modules
5 apache2ctl -M | grep ssl
6
7 # Verify virtual hosts
8 apache2ctl -S
```

8 Assessment Checkpoints

8.1 Checkpoint 1: OpenSSL Server Validation (5 marks)

- Successfully launch OpenSSL `s_server` on port 4433
- Demonstrate browser access with security warning
- Show CA certificate import process
- Verify trusted connection after certificate installation

8.2 Checkpoint 2: Secondary Domain Testing (5 marks)

- Repeat OpenSSL server test for `webserverlab.com`
- Validate certificate chain and domain matching

- Demonstrate proper browser security indicators

8.3 Checkpoint 3: Apache HTTPS Configuration (5 marks)

- Configure Apache virtual host with SSL
- Enable SSL module and test configuration
- Demonstrate `https://example.com/` serving content
- Show proper HTTP to HTTPS redirection

8.4 Checkpoint 4: Multi-Domain Deployment (5 marks)

- Configure Apache for `webserverlab.com` HTTPS
- Demonstrate simultaneous secure hosting
- Validate security headers and TLS configuration
- Show proper certificate assignment per domain

9 Troubleshooting Guide

9.1 Common Issues and Solutions

Table 1: Common TLS Configuration Issues

Issue	Solution
OpenSSL CA database errors	Ensure `demoCA/index.txt` exists and `demoCA/serial` contains valid number
Apache configuration syntax errors	Run `sudo apache2ctl configtest` and check error logs
Certificate trust errors	Verify CA certificate import in browser certificate store
Private key permissions	Set proper permissions: `chmod 400 *.key`
Mixed content warnings	Ensure all resources are loaded via HTTPS

9.2 Log Analysis

Listing 17: Log File Monitoring

```
# Monitor Apache error logs
sudo tail -f /var/log/apache2/error.log

# Check SSL specific logs
sudo tail -f /var/log/apache2/example.com_ssl_error.log

# System journal for service issues
sudo journalctl -u apache2 -f
```

10 Conclusion

10.1 Implementation Summary

This laboratory successfully demonstrates:

- Complete PKI infrastructure establishment
- Professional certificate lifecycle management
- Secure Apache web server configuration
- Industry-standard TLS security practices
- Comprehensive testing and validation procedures

10.2 Security Best Practices Implemented

- Strong cryptographic algorithms (SHA-256, 2048+ bit keys)
- Proper file permissions and access controls
- Security headers implementation (HSTS, XSS protection)
- Modern TLS protocol configuration validity period management

10.3 Further Enhancements

For production environments, consider:

- Implementing certificate revocation lists (CRL)
- Online Certificate Status Protocol (OCSP) stapling
- Automated certificate renewal processes
- Certificate transparency logging
- Regular security audits and updates

Appendix: Quick Reference Commands

Listing 18: Complete Command Reference

```
# Environment setup
mkdir -p ~/lab5_ca/demoCA/{certs,crl,newcerts,private}
touch ~/lab5_ca/demoCA/index.txt
echo 1000 > ~/lab5_ca/demoCA/serial

# CA creation
openssl genrsa -aes256 -out demoCA/private/ca.key 4096
openssl req -new -x509 -days 3650 -sha256 \
    -key demoCA/private/ca.key -out demoCA/cacert.pem

# Server certificate
openssl genrsa -aes256 -out example.com.key 2048
openssl req -new -key example.com.key -out example.com.csr
openssl ca -in example.com.csr -out example.com.crt \
    -cert demoCA/cacert.pem -keyfile demoCA/private/ca.key
```

```
16
17 # Apache configuration
18 sudo a2enmod ssl
19 sudo a2ensite example.com-ssl.conf
20 sudo systemctl reload apache2
```