COMP 4820

Project

POODLE
(Padding Oracle On Downgraded Legacy Encryption)
Demo of the System

Demo Link: https://youtu.be/Ww4JrvyIikE

Ujjwal Singla
MD Sakif Al Mohaimen
Sarwar Ahsan
Tanaya Siddiqui
Arnav Verma

Secure Cookie Management and Encryption in a Web Application

Abstract

This report outlines the design and operation of a secure web application that uses encryption to manage user sessions through cookies. It details the use of the Express.js framework, body parsing middleware, cookie parsing, and the Node.js crypto module for encryption and decryption purposes.

Introduction

The objective of this web application is to demonstrate secure user authentication and session management practices. Utilizing Express.js for server operations and crypto for cryptographic functions, the system provides a basic yet secure platform for handling user login data.

System Architecture

The application is built on Node.js, leveraging Express.js as its core framework. The middleware used includes `body-parser` for parsing incoming request bodies and `cookie-parser` for handling cookies. The encryption is handled by the crypto module, which ensures the confidentiality of user credentials stored within cookies.

Middleware Integration

- Body Parser: Parses the content of incoming requests and makes it accessible through `req.body`.
- Cookie Parser: Parses cookie header and populates `req.cookies` with an object keyed by cookie names.

Encryption Utilities

- Crypto Module: Provides cryptographic functionality that includes a set of wrappers for OpenSSL's hash, HMAC, cipher, decipher, sign, and verify functions.
- AES-256-CBC Mode: Symmetric key algorithm used for encrypting and decrypting user data.

Encryption and Decryption Flow

The system defines a secret key for AES encryption and ensures that it is of the appropriate length (32 bytes) to comply with AES-256 standards. An initialization vector (IV) is generated randomly for each encryption operation to enhance security.

Encryption Process:

1. User enters a username which is passed to the `encrypt` function.

2. The plaintext is encrypted using AES-256-CBC mode with the provided key and IV.
3. The resulting ciphertext is encoded in Base64 and set as a cookie.

Decryption Process:

1. Upon subsequent requests, the server attempts to decrypt the cookie value.
2. If decryption is successful, the username is retrieved, and a welcome message is displayed.
3. If decryption fails, an error is caught, and an appropriate message is returned based on the error type.

User Interaction Flow

- Homepage Route (`GET /`): Displays a form for the user to enter a username if no encrypted cookie is present.
- Submit Route (`POST /submit`): Handles form submission, encrypts the username, sets the cookie, and redirects to the welcome page.
- Welcome Route (`GET /welcome`): Displays the decrypted username from the cookie if available or redirects to the homepage if not.
- Logout Route (`POST /logout`): Clears the cookie and redirects the user to the homepage.

Security Considerations

The system is designed with security in mind; however, it includes deliberate vulnerabilities for educational purposes, such as displaying specific decryption error messages.

Conclusion

The web application represents a fundamental but practical approach to managing secure user sessions in web applications. It illustrates encryption implementation, cookie handling, and secure user management practices.