МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики

Кафедра информационных технологий

ЗАДАНИЕ № 7
по дисциплине
«АППАРАТНО-ПРОГРАММНЫЕ СРЕДСТВА WEB»

Выполнила студентка группы 25/2 _____ А.Е. Головкин

Направление подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем

Курс 2

Краснодар
2023 г.

Задание: Проведите аудит безопасности вашего приложения и исправьте уязвимости. В нем должны быть разделы, посвященные уязвимостям XSS, SQL Injection, CSRF, Include, Upload. В отчете укажите по каждой уязвимости примененные методы защиты с примерами вашего кода.

1. XSS уязвимости

При XSS уязвимостях произвольный JavaScript выполняется в контексте сессии пользователя.

Проведём простейшую проверку и попробуем вставить

<script>alert(123)</script> во все допустимые поля.

Отметим, что данные не были записаны в базу. Этого удалось добиться за счёт того, что данные проходят валидацию:

```php
if (empty($data['name'])) {
    setcookie('fio_error'.$row, '1', time() + 24 * 60 * 60);
    $errors = true;
} else {
    if (!preg_match('/^([а-яА-ЯЁёa-zA-Z0-9_,.\s-]+)$/u', $data['name'])) {
        setcookie('fio_error'.$row, '1', time() + 24 * 60 * 60);
        $errors = true;
    }
    setcookie('fio_value', $data['name'], time() + 365 * 24 * 60 * 60);
}

if (empty($data['email'])) {
    setcookie('email_error'.$row, '1', time() + 24 * 60 * 60);
    $errors = true;
} else {
    if (!filter_var($data['email'], FILTER_VALIDATE_EMAIL)) {
        setcookie('email_error'.$row, '1', time() + 24 * 60 * 60);
        $errors = true;
    }
    setcookie('email_value', $data['email'], time() + 365 * 24 * 60 * 60);
}
```

2. SQL Injection

Изменение SQL запроса параметрами, поступившими от пользователя.

Приложение защищено от SQL уязвимостей, так как используются подготовленные запросы:

```php
$stmt = $db->prepare("SELECT * FROM app_ability2 WHERE id_app = ?");
$stmt -> execute([$id]);
```

3. CRSF уязвимости

Выполнение запроса на атакуемый сайт в контексте сессии пользователя при открытии им страницы с вредоносным кодом, на стороннем сайте. Несанкционированное использование POST и GET.

Для защиты от этого типа уязвимости используется токен.

И в проверяемом приложении токен не использовался, значит оно CRSF уязвимо.

Но вот как можно исправить данную уязвимость:

При запросе от клиента на стороне сервера генерируется токен:

```php
if (empty($_SESSION['token'])) {
    $_SESSION['token'] = bin2hex(random_bytes(32));
}
$token = $_SESSION['token'];
```

Проверка токена:

```php
if (!empty($_POST['token'])) {
    if (!hash_equals($_SESSION['token'], $_POST['token'])) {
        header($_SERVER['SERVER_PROTOCOL'] . ' 405 Method Not Allowed');
        exit;
    }
}
else {
    header($_SERVER['SERVER_PROTOCOL'] . ' 405 Method Not Allowed');
    exit;
}
```

4. Include и upload уязвимости

Include уязвимости возникают, когда код позволяет использовать данные, переданные сценарию в качестве параметров функции include.

Так как в проверяемом приложении функция include используется лишь следующим образом include('module.php'), подобной уязвимости нет.

Upload уязвимости возникают из-за неправильной обработки загружаемого файла.

Данной уязвимости нет, так как в приложении нет функций для загрузки файлов.