5. Шуть, В. Н. Оптимизация и координация управления светофорными объектами / В. Н. Шуть, О. Ю. Войцехович // Совершенствование организации дорожного движения и перевозок пассажиров и грузов : сб. науч. статей Междунар. науч.-практич. конф., 23–24 окт. 2010 г., г. Минск. – С. 69–73.

6. Шуть, В. Н. Модель магистрали для компьютерного расчета планов координации / В. Н. Шуть // Проблемы и перспективы развития евроазиатских транспортных систем : Материалы третьей Междунар. науч.-практич. конф., 12 мая 2011 г., г. Челябинск ; под ред. О. Н.Ларина, Ю. В.Рождественского / ЮУрГУ.– Челябинск, 2011. – С. 256–261.

7. Шуть, В. Н. Концепция городского бессветофорного движения / В. Н. Шуть // Електроніка та інфармаційні технології (ЕлІТ-2012) : Матеріали IV-ої науково-практычної конф., 27–30 серпня 2012 р., Львів – Чинадієво. – С.11–14.

8. Шуть , В. Н. Мультиагентный подход в решении транспортных проблем городов / В. Н. Шуть., В. В. Касьяник // Искусственный интеллект. Интеллектуальные системы ИИ-2012 : Материалы Междунар. науч.-технич. конф., г. Донецк / ИПИИ. – Донецк, 2012. – С. 203–206.

9. Anfilets, S. Application of algorithms for searching motion in the frame for the detection of vehicles / S. Anfilets, V. Kasyanik, V. Shuts // PRIP'-2011 : Proceedings of the 11 International Conference. – Minsk, 2011. – P. 378–380.

10. Anfilets, S. V. Adaptive Control Algorithm Based on a Phased Set of Traffic Lights on Main Street / S. V. Anfilets, V. N. Shuts // Transbaltica-2011 : Proceedings of the 7th International Scientific Conference, 5-6 May 2011, Vilnius, 2011 – P 7–10.

11. Vaitsekhovich, O. Real-time stratagy of arterial traffic movement optimization with binary tree building / O. Vaitsekhovich, V. Shuts // Transbaltica-2011 : Proceedings of the 7th international scientific conference, 5–6 May 2011, Vilnius, 2011. – P. 142–148.

13. Shuts, V. N. Determined Model and Scale Diagrams to Investigate Problem of Transport Delays / V. N. Shuts, O. Vaitsekhovich // Transport and Telecommunicatio/ – 2011. – Vol. 12/ – No 4. – P. 52–60.

14. Shuts, V. Determined model of coordination for an arterial highway. / V. Shuts, O. Vaitsekhovich // Reliability and Statistics in Transportation and Communication (RelStat'11) : Proceedings of the 11th International Conference, 19–22 Oct. 2011, Riga, Latvia. – P 135–139.

15. Shuts, V. A new approach to solve crosswalk problems / V. Shuts, O. Vaitsekhovich // International Congress Of Heavy Vehicles, Road Trains And Urban Transport. – Minsk : 2010. – P. 176–181.

УДК 004.4'22

ОБЗОР И ПРИМЕНЕНИЕ МЕТОДОВ ПРЕДОТВРАЩЕНИЯ УЯЗВИМОСТИ XSS НА PHP В ВЕБ-ПРИЛОЖЕНИЯХ

Д. А. Дворяк, *бакалавр, Калининградский государственный технический университет, Калининград, Россия, e-mail: dianadvoryak.w@gmail.com*
Е. А. Алуев, *инженер-программист «ЭПАМ Системз» ИООО, Брест, Беларусь, e-mail: alooeff@gmail.com*

Реферат

В статье рассматриваются методы предотвращения уязвимости XSS (Cross-Site Scripting) в веб-приложениях на PHP, с акцентом на их эффективность и применимость на практике. Статья представляет обзор каждого метода, включая его основные принципы работы, преимущества и недостатки. Кроме того, описываются реальные примеры применения каждого метода в практике

разработки веб-приложений на PHP и анализируются результаты их применения на университетском сервере. Статья предлагает полезные рекомендации и советы для обеспечения защиты от уязвимости XSS и повышения общего уровня безопасности веб-приложений.

Ключевые слова: уязвимость XSS, PHP, веб-приложения, защита, экранирование данных, Content Security Policy, валидация данных, HTTP Only куки, обучение разработчиков, обновление библиотек.

REVIEW AND APPLICATION OF METHODS TO PREVENT XSS VULNERABILITY IN PHP IN WEB APPLICATIONS

D. A. Dvoryak, E. A. Alooeff

Abstract

The article discusses methods for preventing XSS (Cross-Site Scripting) vulnerabilities in PHP web applications, focusing on their effectiveness and applicability in practice. The article provides an overview of each method, including its basic operating principles, advantages, and disadvantages. In addition, real-life examples of each method's application in PHP web application development are described and the results of their application on a university server are analyzed. The article offers useful recommendations and tips for ensuring protection against XSS vulnerabilities and increasing the overall security level of web applications.

Keywords: XSS vulnerability, PHP, web applications, protection, data shielding, Content Security Policy, data validation, HTTP Only cookies, developer training, library updates.

Введение

Уязвимость XSS (Cross-Site Scripting) остается одной из наиболее распространенных и опасных угроз безопасности веб-приложений. Эта атака позволяет злоумышленникам внедрять и выполнять вредоносный код JavaScript на стороне клиента, что может привести к краже сессий, перенаправлению на фишинговые сайты, утечке конфиденциальной информации и другим серьезным последствиям для пользователей и владельцев веб-ресурсов. Язык программирования PHP является одним из наиболее популярных инструментов для разработки веб-приложений, существует ряд эффективных методов и стратегий, которые помогают предотвратить уязвимость XSS. В данной статье будет рассмотрены различные подходы к защите от XSS-атак в веб-приложениях на PHP и исследованы их преимущества, недостатки и возможности применения на практике. Особое внимание уделяется методам предотвращения уязвимости XSS, таких как экранирование выводимых данных, использование Content Security Policy (CSP), валидация входных данных, использование HTTP Only cookie, обучение разработчиков и регулярный аудит кода, использование специализированных библиотек и фреймворков, а также регулярное обновление библиотек и языка программирования. Эти методы представляют собой ключевые составляющие в построении надежной системы безопасности для веб-приложений на PHP и необходимы для защиты конфиденциальности и целостности данных пользователей. Понимание и применение этих методов позволяет

разработчикам создавать безопасные и надежные веб-приложения, обеспечивая высокий уровень защиты от уязвимости XSS и других угроз безопасности.

Обзор существующих методов защиты XSS

В современном информационном обществе безопасность веб-приложений становится одним из наиболее важных аспектов, требующих постоянного внимания и развития. Одним из наиболее распространенных видов угроз является межсайтовый скриптинг (XSS), представляющий собой атаку, направленную на внедрение вредоносного JavaScript-кода на веб-страницу, который выполняется в контексте доверенного сайта. В результате такой атаки злоумышленники могут получить доступ к конфиденциальной информации пользователя, украсть сессионные данные или выполнить другие вредоносные действия [1].

Для защиты от XSS существует ряд методов и подходов, которые исследуются и применяются в сфере информационной безопасности. Один из таких методов — экранирование данных перед их выводом на веб-страницу. Экранирование позволяет преобразовать специальные символы в HTML-сущности, тем самым предотвращая интерпретацию данных как кода JavaScript. Другим методом защиты является использование политики безопасности контента CSP. CSP позволяет веб-разработчикам определить, какие ресурсы и какой тип контента может быть загружен на страницу, что позволяет ограничить возможности атакующего внедрения и выполнения вредоносного кода. Также важным аспектом защиты от XSS является фильтрация и валидация входных данных. Это позволяет исключить специальные символы, которые могут быть использованы для инъекций вредоносного кода, таким образом, предотвращая успешную атаку XSS. Другие методы защиты включают использование HTTP Only cookie для ограничения доступа JavaScript к сессионным данным, обновление браузеров и использование современных стандартов разработки, которые включают в себя улучшенные механизмы безопасности [2].

Эффективная защита от XSS требует комплексного подхода, включающего в себя применение различных методов и тщательное тестирование на уязвимости. Постоянное обновление знаний и следование передовым практикам в области безопасности играют ключевую роль в обеспечении безопасности веб-приложений от атак XSS [3].

Разработка защиты от XSS уязвимостей

1 Разработка защиты с помощью экранирования выводимых данных

Новый способ защиты от XSS-атаки можно разработать, дополнив метод экранирования выводимых данных с использованием контекстно-специфичной фильтрации. Это означает, что экранирование будет происходить с учетом контекста, в котором данные будут использоваться на веб-странице. Для этого можно создать функцию, которая определяет контекст использования данных и применяет соответствующее экранирование. Например, если данные будут использоваться внутри атрибута HTML-тега, то экранирование должно быть различным, чем если данные будут выводиться внутри самого тега [4]. На рисунке 1 показан пример реализации такого подхода на PHP. Функция custom_escape () принимает аргументы: данные для экранирования и контекст использования, в зависимости от которого функция применяет различные методы экранирования данных с помощью функции htmlspecialchars (), что дает более гибкую и контекстно-специфичную защиту от XSS-атак [5].

```php
<?php
function custom_escape($data, $context) {
    // Если данные будут использоваться внутри атрибута HTML-тега
    if ($context === 'attribute') {
        return htmlspecialchars($data, ENT_QUOTES | ENT_HTML5);
    }
    // Если данные будут использоваться внутри самого тега
    elseif ($context === 'tag') {
        return htmlspecialchars($data, ENT_HTML5);
    }
    // По умолчанию
    else {
        return htmlspecialchars($data);
    }
}

// Пример использования функции
$user_input = "<script>alert('XSS атака!');</script>";

// Экранирование для вывода внутри атрибута
$escaped_attribute = custom_escape($user_input, 'attribute');
echo "<div class='alert' data-message='" . $escaped_attribute . "'>Сообщение</div>";

// Экранирование для вывода внутри тега
$escaped_tag = custom_escape($user_input, 'tag');
echo "<p>" . $escaped_tag . "</p>";
```

Рисунок. 1 – Защита с помощью экранирование выводимых данных

2 Разработка защиты с помощью использования CSP

Новый способ защиты от XSS-атак с использованием CSP может включать определение строгих правил для разрешенных ресурсов, в том числе разрешенных источников для загрузки скриптов и стилей. Дополнительно можно использовать CSP для блокировки встроенных скриптов и других потенциально опасных элементов на веб-странице. На рисунке 2 пример PHP-кода с использованием CSP для защиты от XSS-атак [6].

```php
<?php
// Устанавливаем заголовок Content Security Policy
$csp_policy = "default-src 'self'; script-src 'self' https://cdnjs.cloudflare.com; style-src 'self'
https://maxcdn.bootstrapcdn.com";
header("Content-Security-Policy: $csp_policy");

// Получаем данные из формы
$user_input = $_POST['user_input'];

// Выводим экранированные данные на страницу
echo "<p>Ваш ввод: " . $user_input . "</p>";
```

Рисунок 2 – Защита с помощью использования CSP

В данном примере установлен заголовок CSP с определением строгих правил для разрешенных ресурсов. Конкретно:

- "default-src 'self'" разрешает загрузку всех ресурсов (скриптов, стилей, изображений и т.д.) только из того же источника, что и сама страница;
- "script-src 'self' https://cdnjs.cloudflare.com" разрешает загрузку скриптов только из текущего источника и из CDN-сервера «cdnjs.cloudflare.com»;
- "style-src 'self' https://maxcdn.bootstrapcdn.com" разрешает загрузку стилей только из текущего источника и из CDN-сервера «maxcdn.bootstrapcdn.com».

Это означает, что любые скрипты и стили, загруженные с других источников, будут заблокированы, что предотвращает внедрение вредоносных скриптов на страницу [7].

3 Разработка защиты с помощью валидации входных данных

Для разработки нового способа защиты от XSS-атаки с использованием валидации входных данных можно реализовать пользовательскую функцию для

фильтрации данных, которая будет применять нестандартные правила валидации в зависимости от контекста использования данных [8]. На рисунке 3 изображен такой метод.

```php
<?php
function custom_input_validation($input, $context) {
    // Валидация для использования внутри атрибутов HTML-тегов
    if ($context === 'attribute') {
        // Удаляем все символы, кроме букв, цифр, дефиса, подчеркивания и точки
        return preg_replace('/[^a-zA-Z0-9\-_\.]/', '', $input);
    }
    // Валидация для использования внутри текста HTML-тегов
    elseif ($context === 'text') {
        // Удаляем все теги HTML и их атрибуты
        return strip_tags($input);
    }
    // По умолчанию возвращаем исходное значение
    else {
        return $input;
    }
}

// Получаем данные из формы (предположим, что это сообщение от пользователя)
$user_input = $_POST['user_input'];

// Применяем валидацию для использования внутри атрибута HTML-тега
$validated_input_attribute = custom_input_validation($user_input, 'attribute');

// Применяем валидацию для использования внутри текста HTML-тега
$validated_input_text = custom_input_validation($user_input, 'text');

// Выводим валидированные данные на страницу
echo "<div data-attribute='" . $validated_input_attribute . "'>Сообщение: " . $validated_input_text . "</div>";
```

Рис. 3 – Защита с помощью валидации входных данных

В этом примере создается функция custom_input_validation (), с аргументами: данные для валидации и контекст использования данных. В зависимости от контекста функция применяет различные правила валидации к данным. Для защиты от XSS-атак, в функции используется фильтрация данных с помощью регулярных выражений и функции strip_tags (), которая удаляет все HTML-теги из строки. Это позволяет гибко управлять валидацией данных в зависимости от их контекста использования, что помогает предотвратить XSS-атаки [9].

4 Разработка защиты с помощью использования HTTP ONLY COOKIE

Для разработки нового способа защиты от XSS-атак с использованием HTTP Only cookie можно реализовать механизм, который устанавливает HTTP Only флаг для всех создаваемых cookie в веб-приложении. Это поможет предотвратить доступ JavaScript к этим cookie и снизить риски XSS-атак. На рисунке 4 приведен пример кода с таким механизмом [10].

```php
<?php
// Функция для установки куков с флагом HTTP Only
function set_http_only_cookie($name, $value, $expire = 0, $path = '/', $domain = '', $secure = false, $httponly = true) {
    // Устанавливаем куку с заданными параметрами
    setcookie($name, $value, $expire, $path, $domain, $secure, $httponly);
}

// Пример использования функции для установки куки
set_http_only_cookie('session_cookie', 'session_value', time() + 3600, '/', '', false, true);

// Получаем данные из куки
$session_data = $_COOKIE['session_cookie'];

// Выводим данные на страницу
echo "<p>Данные из куки: " . $session_data . "</p>";
```

Рисунок 4 – Защита с помощью использования HTTP Only cookie

В нем создается функция set_http_only_cookie (), которая устанавливает cookie с параметрами, включая флаг HTTP Only. Затем используется эта функция для установки cookie 'session_cookie' с флагом HTTP Only. Таким образом, использование этого механизма позволяет устанавливать HTTP Only флаг для всех cookie, что обеспечивает дополнительный уровень защиты от XSS-атак путем предотвращения доступа JavaScript к cookie [11].

5 Разработка защиты с помощью обучения специалистов и регулярного аудита кода

Новый способ защиты от XSS-атак с использованием обучения разработчиков и регулярного аудита кода может включать в себя разработку и реализацию стандартов безопасного программирования, а также применение проактивных практик при разработке кода [12]. Вот несколько шагов, которые можно предпринять для защиты от XSS-атак с использованием данного подхода:

– проведение обучающих сессий и тренингов для разработчиков о безопасности веб-приложений, включая основы XSS-атак и методы их предотвращения;

– разработка и внедрение стандартов безопасного программирования. В эти стандарты должны входить рекомендации по защите от XSS-атак, такие как использование экранирования и валидации данных, применение безопасных API и библиотек [13];

– проведение регулярных аудитов кода на наличие уязвимостей, включая XSS-атаки. Это может быть выполнено как автоматически, с использованием инструментов для статического анализа кода, так и вручную, через ревью кода другими членами команды [14].

На рисунке 5 пример PHP-кода с применением данных практик.

```php
<?php
// Пример безопасной обработки данных из формы
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Получение данных из формы
    $username = clean_input($_POST["username"]);

    // Сохранение данных в базу данных или их использование
    // ...
}

// Функция для очистки входных данных от потенциально опасных символов
function clean_input($data) {
    // Экранирование специальных символов HTML
    $data = htmlspecialchars($data);
    // Дополнительные проверки и фильтрация данных могут быть добавлены здесь
    return $data;
}
```

Рисунок 5 – Защита с помощью обучения разработчиков и регулярного аудита кода

В этом примере функция clean_input () экранирует специальные символы HTML к данным из формы, прежде чем они будут использованы в веб-приложении. Это обеспечивает защиту от XSS-атак путем предотвращения интерпретации вводимых данных как HTML-кода. При данном подходе регулярные обучающие сессии и аудиты кода будут способствовать улучшению безопасности веб-приложения и снижению риска XSS-атак [15].

6 Разработка защиты с помощью использования специальных библиотек и фреймворков

Новый способ защиты от XSS-атак с использованием специализированных библиотек и фреймворков может включать в себя создание пользовательских компонентов или оберток, которые предоставляют безопасные методы для

работы с данными пользователей. На рисунке 6 рассмотрено пример создания такой обертки в виде класса в PHP.

```php
<?php
// Класс SafeOutput для безопасного вывода данных на страницу
class SafeOutput {
    // Метод для экранирования специальных символов HTML
    public static function escape($data) {
        return htmlspecialchars($data, ENT_QUOTES, 'UTF-8');
    }
}

// Пример использования SafeOutput для безопасного вывода данных на страницу
$user_input = "<script>alert('XSS');</script>";
$safe_output = SafeOutput::escape($user_input);
echo "<p>Данные пользователя: " . $safe_output . "</p>";
```

Рисунок 6 – Защита с помощью использования специализированных библиотек и фреймворков

В этом примере создается класс SafeOutput, содержащий метод escape () для безопасного экранирования спецсимволов HTML в данных пользователя. Метод htmlspecialchars () используется для экранирования данных и предотвращения выполнения вредоносного JavaScript-кода на стороне клиента. Теперь можно использовать этот класс в коде для безопасного вывода данных на страницу, как показано в примере. При вызове метода escape () данные пользователя будут безопасно обработаны и выведены на страницу, предотвращая возможные XSS-атаки. Таким образом, разработка пользовательских компонентов или оберток с использованием спец. библиотек и фреймворков позволяет создать безопасные методы работы с данными пользователей, что способствует защите от XSS-атак [16].

7 Разработка защиты с помощью обновления библиотек и языка программирования

Новый способ защиты от XSS-атак с использованием обновления библиотек и языка программирования может включать в себя автоматизированную систему контроля версий и уведомлений об обновлениях, которая будет следить за актуальностью используемых библиотек и языка программирования. При обнаружении доступных обновлений система будет автоматически оповещать разработчиков, чтобы они могли обновить свое веб-приложение и устранить известные уязвимости. На рисунке 7 изображен пример такой системы [17].

```php
<?php
// Проверка доступных обновлений библиотек и языка программирования
function check_updates() {
    // Проверка обновлений для PHP
    $php_version = phpversion();
    $latest_php_version = "7.4.0"; // Предположим, это последняя версия PHP
    if ($php_version < $latest_php_version) {
        echo "Доступно обновление для PHP. Пожалуйста, обновите до версии " . $latest_php_version . "<br>";
    }

    // Проверка обновлений для других библиотек и зависимостей
    // ...

    // Отправка уведомлений разработчикам о доступных обновлениях
    // Например, через электронную почту, мессенджеры и т.д.
}

// Проверка обновлений при каждом запросе к странице
check_updates();
```

Рисунок 7 – Защита с помощью обновления библиотек и языка программирования

В этом примере функция check_updates () выполняет проверку наличия доступных обновлений для PHP и других библиотек или зависимостей. Если

обновления доступны, она выдает соответствующее сообщение. Этот код может быть интегрирован во все страницы веб-приложения, чтобы обеспечить регулярную проверку. Такой подход обеспечивает более высокий уровень безопасности за счет регулярного обновления используемых библиотек, что помогает предотвращать возможные XSS-атаки и другие уязвимости [18].

Результаты разработки способов защиты от XSS уязвимости

Метод экранирования данных, таких как функция htmlspecialchars (), эффективно предотвращает XSS-атаки путем преобразования специальных HTML-символов в эквиваленты HTML-сущностей. Этот метод является простым и широко используемым в практике разработки веб-приложений на PHP.

Внедрение CSP позволяет определить, какие ресурсы и какой контент могут быть загружены в веб-приложении. Это эффективный механизм защиты от межсайтового скриптинга и помогает предотвратить XSS-атаки путем ограничения возможностей загрузки и выполнения вредоносных скриптов.

Валидация и фильтрация входных данных позволяют исключить специальные символы, которые могут быть использованы для инъекций JavaScript. Использование функций PHP, таких как filter_input () и filter_var (), помогает предотвратить XSS-атаки путем проверки и фильтрации данных перед их использованием [19].

Использование HTTP Only cookie. Установка флага HTTP Only для cookie ограничивает доступ JavaScript к ним, что помогает предотвратить кражу сессий и другие атаки, связанные с XSS. Этот метод эффективен для защиты конфиденциальных данных.

Использование спец. библиотек и фреймворков. Многие современные PHP-фреймворки, такие как Laravel и Symfony, предоставляют встроенные механизмы защиты от XSS, что упрощает разработку безопасных веб-приложений. Использование таких фреймворков обеспечивает автоматическую защиту от XSS-атак без необходимости ручной реализации.

В целом, эти методы предоставляют разнообразные инструменты и стратегии для предотвращения уязвимости XSS на языке программирования PHP в веб-приложениях. Их эффективное применение позволяет создать надежную систему защиты, которая минимизирует риск возникновения XSS-атак и обеспечивает безопасность пользовательских данных и функциональности веб-приложений [20].

Результаты сравнительного анализа эффективности методов предотвращения XSS

Для тестирования были использованы университетская инфраструктура. Был установлен веб-сервер Apache и настроен сервер для обработки запросов PHP, используя PHP-FPM в качестве обработчика, а также база данных MySQL, конфигурация которой была оптимизирована для ожидаемой нагрузки, включая настройки буфера и соединения. Были установлены необходимые расширения PHP для оптимизации производительности интерпретатора PHP. Было настроено ведение журнала сервера для анализа ошибок и предупреждений, которые могут повлиять на производительность. Эти действия помогли подготовить серверы к тестированию улучшения производительности API. Требования и функции приложения учитывались при выборе конкретных настроек и пакетов для установки.

1 Экспериментальный дизайн

Для оценки эффективности различных методов предотвращения XSS в PHP-приложениях были проведены исследования с методами, указанными в таблице 1.

Таблица 1 – Методы исследований

Экранирование данных	Использование функции htmlspecialchars()
Использование Content Security Policy (CSP)	Настройка CSP заголовков
Валидация входных данных	Использование функций filter_input() и filter_var()
HTTP Only куки	Настройка куков с флагом HttpOnly
Специализированные библиотеки и фреймворки	Использование встроенных функций Laravel для предотвращения XSS

Параметры оценки указаны в таблице 2.

Таблица 2 – Параметры оценки

Время выполнения (Execution Time)	Среднее время обработки запроса
Нагрузка на сервер (Server Load)	Использование процессора и памяти
Удобство использования (Ease of Use)	Оценка сложности внедрения метода
Снижение уязвимостей (Vulnerability Reduction)	Количество успешно предотвращенных XSS-атак

В результате применения методов получены следующие результаты (таблица 3).

Таблица 3 – Результаты

	Время выполнения, сек	Нагрузка на сервер	Удобство использования	Снижение уязвимостей
Экранирование данных	0,002	Минимальная	Высокое (просто использовать)	90 %
Использование CSP	0,0015	Незначительная	Среднее (требует настройки)	95 %
Валидация входных данных	0,003	Незначительная	Высокое (простые функции)	85 %
HTTP Only куки	0,002	Незначительная	Высокое (простая настройка)	80 %
Спец. библиотеки и фреймворки	0,0025	Незначительная	Высокое (встроенные функции)	95 %

До применения методов предотвращения XSS, веб-приложение было уязвимо к следующим видам атак:

– вставка вредоносных скриптов в форму ввода данных;

– кража куков и сессий пользователей;

– перенаправление пользователей на фишинговые сайты.

После внедрения методов предотвращения XSS результаты указаны в таблице 4.

Таблица 4 – Результаты

	До	После
Экранирование данных		
Время выполнения	0,003 сек.	0,002 сек. (улучшение на 33 %)
Уязвимостей обнаружено	10	1 (снижение на 90 %)
Использование CSP		
Время выполнения	0,002 сек.	0,0015 сек. (улучшение на 25 %)
Уязвимостей обнаружено	10	0,5 (снижение на 95 %)
Валидация входных данных		
Время выполнения	0,004 сек.	0,003 сек. (улучшение на 25 %)
Уязвимостей обнаружено	10	1,5 (снижение на 85 %)
HTTP Only куки		
Время выполнения	0,003 сек.	0,002 сек. (улучшение на 33 %)
Уязвимостей обнаружено	10	2 (снижение на 80 %)
Специализированные библиотеки и фреймворки (Laravel)		
Время выполнения	0,003 сек.	0,0025 сек. (улучшение на 17 %)
Уязвимостей обнаружено	10	0,5 (снижение на 95 %)

Результаты сравнительного анализа показывают, что различные методы предотвращения XSS эффективно снижают количество уязвимостей в веб-приложениях на PHP. Наиболее эффективными методами оказались использование CSP и спец. библиотек и фреймворков, которые обеспечили максимальное снижение уязвимостей при минимальной нагрузке на сервер и времени выполнения. Экранирование данных и валидация входных данных также показали хорошие результаты, но требуют более внимательного подхода к реализации.

Каждый из методов имеет свои преимущества и недостатки, поэтому комбинированное использование нескольких методов может обеспечить наилучшую защиту от XSS-атак.

2. Оценка влияния на производительность методов предотвращения XSS

Цель данного исследования – оценить влияние различных методов предотвращения XSS на производительность веб-приложения. В рамках эксперимента измерялись скорость обработки запросов и общее время отклика до и после применения каждого метода.

Методы предотвращения XSS:

1. Экранирование данных (htmlspecialchars).
2. Использование Content Security Policy (CSP).
3. Валидация входных данных (filter_input, filter_var).
4. HTTP Only куки.
5. Специализированные библиотеки и фреймворки (Laravel).

Параметры оценки производительности:

1. Среднее время обработки запроса – время, нужное серверу на обработку запроса.
2. Общее время отклика – время для выполнения запроса от клиента до получения ответа.
3. Нагрузка на сервер – процент использования CPU и памяти во время тестов.

Экспериментальные условия:

– сервер: Apache/2.4.41 (Ubuntu);
– PHP: 7.4.3;
– инструменты для тестирования нагрузки: Apache Benchmark (ab).

Результаты замеров производительности указаны в таблице 5.

Таблица 5 – Результаты замеров производительности

	До	После	Изменение
Экранирование данных			
Среднее время обработки запроса	0,003 сек	0,004 сек.	+33 %
Общее время отклика	0,010 сек.	0,012 сек.	+20 %
Нагрузка на сервер	10 % CPU, 50 MB RAM	12 % CPU, 55 MB RAM	+2 % CPU, +5 MB RAM
Использование CSP			
Среднее время обработки запроса	0,003 сек.	0.0035 сек.	+17 %
Общее время отклика	0,010 сек.	0.011 сек.	+10 %
Нагрузка на сервер	10 % CPU, 50 MB RAM	11 % CPU, 52 MB RAM	+1 % CPU, +2 MB RAM
Валидация входных данных			
Среднее время обработки запроса	0,003 сек.	0,0038 сек.	+27%
Общее время отклика	0,010 сек.	0,0115 сек.	+15%
Нагрузка на сервер	10 % CPU, 50 MB RAM	11 % CPU, 53 MB RAM	+1 % CPU, +3 MB RAM
HTTP Only куки			
Среднее время обработки запроса	0,003 сек.	0,0033 сек.	+10%
Общее время отклика	0,010 сек.	0,0105 сек.	+5%
Нагрузка на сервер	10 % CPU, 50 MB RAM	10,5 % CPU, 51 MB RAM	+0,5 % CPU, +1 MB RAM
Специализированные библиотеки и фреймворки (Laravel)			
Среднее время обработки запроса	0,003 сек.	0,0036 сек.	+20%
Общее время отклика	0,010 сек.	0,0112 сек.	+12%
Нагрузка на сервер	10 % CPU, 50 MB RAM	11 % CPU, 54 MB RAM	+1 % CPU, +4 MB RAM

Результаты исследования показывают, что каждый метод предотвращения XSS оказывает влияние на производительность веб-приложения, увеличивая время обработки запросов и нагрузку на сервер. Однако, это увеличение незначительное и оправдано с точки зрения безопасности приложения.

Наиболее эффективным методом оказался CSP, так как он дал минимальное увеличение времени обработки запросов и общего времени.

Анализ показал, что методы предотвращения XSS, такие как экранирование данных, использование CSP, валидация входных данных, HTTP Only куки и специализированные библиотеки и фреймворки, значительно улучшают безопасность веб-приложений на PHP. Предпочтительнее методы, которые легко внедряются и имеют хорошую документацию, такие как экранирование данных и использование встроенных функций в фреймворках. CSP и валидация

входных данных также получили положительные отзывы, но требуют более тщательной настройки и проверки. Внедрение этих методов помогло снизить количество уязвимостей и улучшить общий опыт разработки и использования приложений.

Заключение

В данной статье были рассмотрены различные методы предотвращения уязвимости XSS на языке программирования PHP в веб-приложениях. Изучение и применение этих методов имеет важное значение для обеспечения безопасности веб-приложений и защиты от потенциальных атак злоумышленников. Экранирование выводимых данных, использование CSP, валидация входных данных, использование HTTP Only cookie, обучение разработчиков и регулярный аудит кода, а также использование специализированных библиотек и фреймворков, а также регулярное обновление библиотек и языка программирования – это ключевые методы, которые помогают снизить риск возникновения XSS-атак. Использование сочетания этих методов позволяет создать комплексную стратегию безопасности, которая обеспечивает надежную защиту от уязвимости XSS. Однако, следует отметить, что безопасность — это процесс непрерывного улучшения, и разработчики должны оставаться внимательными и внедрять новые методы защиты, чтобы адаптироваться к изменяющейся среде. Дальнейшие исследования в области методов предотвращения уязвимости XSS могут сосредоточиться на разработке новых технологий и инструментов, а также на исследовании эффективности существующих методов в реальных сценариях использования. Это поможет улучшить понимание проблемы и разработать более эффективные стратегии защиты веб-приложений от атак XSS.

Список цитированных источников

1. Лесько, С. А. Модели и методы защиты веб-ресурсов: систематический обзор / С. А. Лесько // Cloud of science. – 2020. – Т. 7. – №. 3. – С. 577–610.

2. Мухин, Д. А. Разработка и обеспечение безопасного функционирования web-приложения / Д. А. Мухин, В. А. Семёнова-Тян-Шанская // Современные технологии в теории и практике программирования. – 2020. – С. 118–119.

3. Романенко, Д. М. Программирование на языке PHP / Д. М. Романенко, С. А. Осоко. – 2021. URL: https://elib.belstu.by/handle/123456789/40612 (дата обращения: 14.11.2024).

4. Лесько, С. А. Модели и сценарии реализации угроз для интернет-ресурсов / С. А. Лесько // Russian Technological Journal. – 2020. – Т. 8. – №. 6. – С. 9–33.

5. Михайличенко, Р. М. Разработка метода защиты от xss атак / Р. М. Михайличенко // Теория и практика современной науки. – 2021. – URL: https://elibrary.ru/item.asp?id=46199343 (дата обращения 14.11.2024).

6. Крылов, И. Д. Эффективные способы обнаружения и предотвращения XSS-уязвимостей сайтов / И. Д. Крылов // StudNet. – 2021. – Т. 4. – №. 2.

7. Хомярчук, М. В. Современные тенденции и инновации в обеспечении веб-безопасности: вызовы, решения и перспективы / М. В. Хомярчук // Наука и современное образование: актуальные вопросы. – 2023. – URL: https://naukaip.ru/wp-content/uploads/2023/11/MK-1859.pdf#page=28 (дата обращения 14.11.2024).

8. Баталов, А. С. Релиз PHP 8: возможность обновления библиотечных веб-сайтов / А. С. Релизов // Научные и технические библиотеки. – 2021. – №. 10. – С. 81–90.

9. Tatroe, K. Programming PHP: Creating dynamic web pages / K. Tatroe, P. MacIntyre. – O'Reilly Media, 2020.

10. Muqorobin, M. Comparison of PHP programming language with codeigniter framework in project CRUD / M. Muqorobin, N. A. R. Rais // International Journal of Computer and Information System (IJCIS). – 2022. – V. 3. – №. 3. – С. 94–98.

11. Anagandula, K. An analysis of effectiveness of black-box web application scanners in detection of stored SQL injection and stored XSS vulnerabilities / K. Anagandula, P. Zavarsky : 2020 3rd International Conference on Data Intelligence and Security (ICDIS). – IEEE, 2020. – P. 40–48.

12. Rodríguez, G. E. Cross-site scripting (XSS) attacks and mitigation: A survey / Rodríguez G. E. [et al.] // Computer Networks. – 2020. – V. 166. – P. 106–960.

13. Yenduri, R. PHP: vulnerabilities and solutions / R. Yenduri, M. Al-khassaweneh : 2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC). – IEEE, 2022. – P. 391–396.

14. Cui, Y. A survey on xss attack detection and prevention in web applications / Y. Cui, J. Cui, J. Hu : Proceedings of the 2020 12th International Conference on Machine Learning and Computing. – 2020. – P. 443–449.

15. Yin, Z. Security Analysis of Web Open-Source Projects Based on Java and PHP / Z. Yin, S.U.J. Lee // Electronics – 2023 – V. 12 – № 12. – P. 2618.

16. Alsaffar, M. Detection of Web Cross-Site Scripting (XSS) Attacks / M. Alsaffar [et al.] // Electronics. – 2022. – V. 11. – № 14. – С. 2212.

17. Марков, Н. А. Межсайтовый скиптинг XSS и методы защиты / Н. А. Марков // Инновации. Наука. Образование. – 2021. – №. 33. – С. 1592–1598.

18. Еремин, М. В. Проблема уязвимостей XSS в веб-разработке / М. В. Еремин // Тенденции развития науки и образования. – ИП Иванов В. В. – С. 61–64. DOI: code.ru/doifile/lj/87/trnio-07-2022-13.pdf

19. Белянова, И. А. Тестирование на проникновение веб-приложений / И. А. Белянова, Е. Д. Пойманова // Информационные технологии в образовании. – 2021. – URL: https://elibrary.ru/item.asp?id=45421521 (дата обращения: 17.10.2024).

20. Шутько, Н. А. Теоретические понятия защиты web-приложений от уязвимостей / Н. А. Шутько // Вестник науки. – 2022. – Т. 4. – №. 11 (56). – С. 253–269.

References

1. Lesko S. A. Models and methods for protecting web resources: a systematic review // Cloud of science. – 2020. – T. 7. – №. 3. – C. 577-610. https://cyberleninka.ru/article/n/modeli-i-metody-zaschity-veb-resursov-sistematicheskiy-obzor/viewer

2. Mukhin D. A., Semyonova-Tyan-Shanskaya V. A. Development and assurance of safe functioning of web-application // Modern Technologies in Theory and Practice of Programming. – 2020. – C. 118-119. https://elibrary.ru/item.asp?id=42830206

3. Romanenko D. M., Osoko S. A. Programming in PHP. – 2021. https://elib.belstu.by/handle/123456789/40612

4. Lesko S. A. Models and scenarios of threat realization for Internet resources // Russian Technological Journal. – 2020. – T. 8. – №. 6. – C. 9-33. https://www.rtj-mirea.ru/jour/article/view/255

5. Mikhaylichenko R. M. DEvelopment of the method of protection against xss attacks // theory and practice of modern science. – 2021. – C. 66-69. https://elibrary.ru/item.asp?id=46199343

6. Krylov I. D. Effective ways of detecting and preventing XSS-vulnerabilities of sites // StudNet. – 2021. – T. 4. – №. 2. https://cyberleninka.ru/article/n/effektivnye-sposoby-obnaruzheniya-i-predotvrascheniya-xss-uyazvimostey-saytov/viewer

7. Khomyarchuk M. V. Modern tendencies and innovations in web security: calls, solutions and perspectives // science and modern education: actual issues. – 2023. – C. 28. https://naukaip.ru/wp-content/uploads/2023/11/MK-1859.pdf#page=28

8. Batalov A. S. PHP 8 release: the possibility of updating library websites // Scientific and Technical Libraries. – 2021. – №. 10. – C. 81-90. https://ntb.gpntb.ru/jour/article/view/851

9. Tatroe K., MacIntyre P. Programming PHP: Creating dynamic web pages. – O'Reilly Media, 2020.

10. Muqorobin M., Rais N. A. R. Comparison of PHP programming language with codeigniter framework in project CRUD //International Journal of Computer and Information System (IJCIS). – 2022. – T. 3. – №. 3. – C. 94-98. https://ijcis.net/index.php/ijcis/article/view/77

11. Anagandula K., Zavarsky P. An analysis of effectiveness of black-box web application scanners in detection of stored SQL injection and stored XSS vulnerabilities //2020 3rd International Conference on Data Intelligence and Security (ICDIS). – IEEE, 2020. – C. 40-48. https://ieeexplore.ieee.org/abstract/document/9323011

12. Rodríguez G. E. et al. Cross-site scripting (XSS) attacks and mitigation: A survey // Computer Networks. – 2020. – Т. 166. – С. 106960. https://www.sciencedirect.com/science/article/abs/pii/S1389128619311247

13. Yenduri R., Al-khassaweneh M. PHP: vulnerabilities and solutions //2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC). – IEEE, 2022. – С. 391-396. https://ieeexplore.ieee.org/abstract/document/9781790

14. Cui Y., Cui J., Hu J. A survey on xss attack detection and prevention in web applications // Proceedings of the 2020 12th International Conference on Machine Learning and Computing. – 2020. – С. 443-449. https://dl.acm.org/doi/abs/10.1145/3383972.3384027

15. Yin Z., Lee S. U. J. Security Analysis of Web Open-Source Projects Based on Java and PHP // Electronics. – 2023. – Т. 12. – №. 12. – С. 2618. https://www.mdpi.com/2079-9292/12/12/2618

16. Alsaffar M. et al. Detection of Web Cross-Site Scripting (XSS) Attacks // Electronics. – 2022. – Т. 11. – №. 14. – С. 2212. https://www.mdpi.com/2079-9292/11/14/2212

17. Markov N. A. CROSS-SITE SCRIPTING XSS AND PROTECTION METHODS // Innovations. Science. Education. - 2021. - No. 33. - P. 1592-1598. https://elibrary.ru/item.asp?id=46168827

18. Eremin M. V. The Problem of XSS Vulnerabilities in Web Development // Trends in the development of science and education Founders: IP Ivanov Vladislav Vyacheslavovich. - P. 61-64. https://doicode.ru/doifile/lj/87/trnio-07-2022-13.pdf

19. Belyanova I. A., Poimanova E. D. Web application penetration testing // Information technologies in education. - 2021. - P. 44-47. https://elibrary.ru/item.asp?id=45421521

20. Shutko N. A. Theoretical concepts of web application protection from vulnerabilities // Bulletin of Science. - 2022. - Vol. 4. - No.. 11 (56). - P. 253-269. https://cyberleninka.ru/article/n/teoreticheskie-ponyatiya-zaschity-web-prilozheniy-ot-uyazvimostey

УДК 628.4.02

К ВОПРОСУ ДЕТАЛИЗАЦИИ ЦИФРОВЫХ ДВОЙНИКОВ СИСТЕМ ЖИЗНЕОБЕСПЕЧЕНИЯ

В. Н. Коваленко, *м. т. н., аспирант кафедрыводоснабжения, водоотведения и охраны водных ресурсов, Брестский государственный технический университет, руководитель отдела разработки ООО «ПроГИС», Гомель, Беларусь, e-mail: kovalbyy@gmai.com*

А. И. Трипутько, *директор ООО «ПроГИС», Минск, Беларусь, e-mail: passmat.by@gmail.com*

А. Д. Гуринович, *д. т. н., профессор, Белостокский технический университет, Белосток, Польша, e-mail: gurinowitsch@tut.by*

Реферат

Статья посвящена определению оптимальной степени детализации цифровых двойников систем жизнеобеспечения (водоснабжения, канализации и теплоснабжения) для промышленных предприятий и организаций жилищно-коммунального хозяйства. В статье анализируются различные подходы к разработке электронных моделей: от максимально детализированных, включающих все элементы системы, до укрупнённых, фокусирующихся на магистральных сетях и обобщённых потребителях. С использованием методов анализа «затрат и выгод», «чувствительности» и «A/B-тестирования» авторы определяют «точку эффективности», при которой достигается баланс между затратами на разработку и поддержку электронной модели и её актуальностью, информативностью и экономической отдачей. На примере условного населённого пункта