МИНОБРНАУКИ РОССИИ
ФГБОУ ВО «СГУ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

ОТЧЕТ

КУРСОВАЯ РАБОТА

студента четвертый курса 431 группы
специальности 090301 — Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Иванова Ксения Владиславовна

Научный руководитель
доцент _____ А. А. Лобов

Заведующий кафедрой
д. ф.-м. н., доцент _____ М. Б. Абросимов

Саратов 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В последнее время компании все чаще предпочитают легкие, быстрые и универсальные web-приложения для своих услуг, взмен тяжеловесным дестктопным. А из этого вытекает появление большого количества фремворков и библиотек для разработки и поддержки web-приложений, которые позволяют реализовать все, что необходимо для функционирования современного web-приложения. Поэтому одной из сложностей является выбор подходящего для целей приложения фреймворка, а для правильного выбора нужно понимать достоинства и недостатки каждого. В этой работе я рассмотрю популярные среди разработчиков фреймворки с точки зрения безопасности, расстмотрю их функционал и оценю его работу.

1 Web-приложение

1.1 Структура и работа

Web-приложение — это клиент-серверное приложение, в котором осуществляется взаимодействие клиента с сервером, за счет браузера, а сервером является веб-сервер, который принимает HTTP-запросы от клиента (браузера), и в свою очередь выдает HTTP-ответы вместе с HTML-страницей, изображением или какими-нибудь другими данными. Клиенты могут получить доступ к веб-серверу по URL адресу необходимой им страницы.

Для веб-приложений на стороне сервера можно применять различные технологии и любые языки программирования. Для клиента-браузера же не важно на какой ОС оно работает, в этом заключается один из главных плюсов – кросс-платформенность (Рис.1).

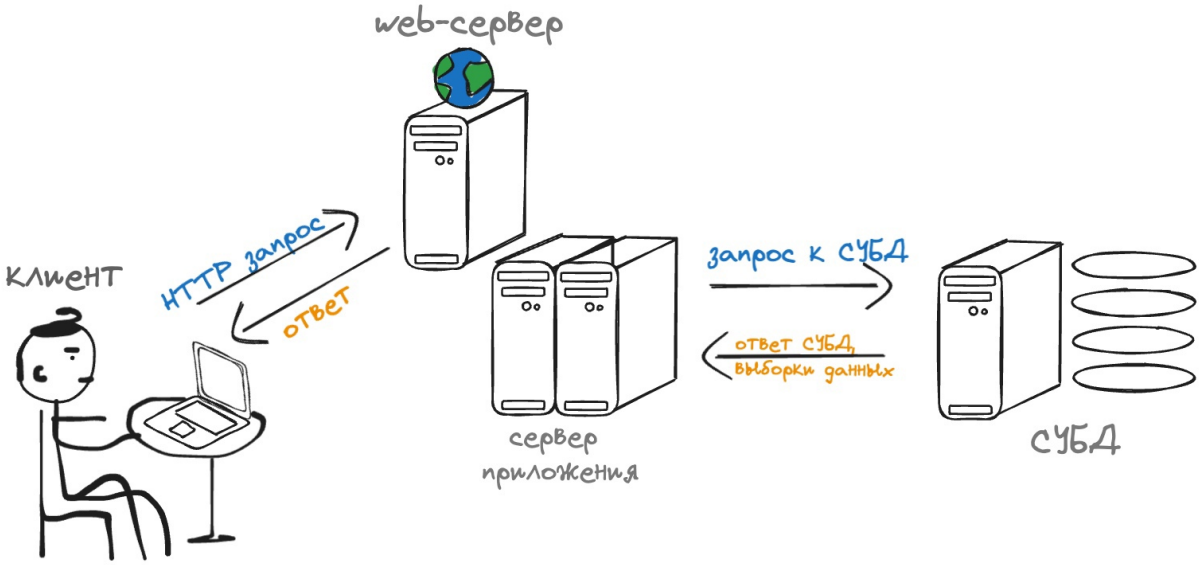

Рисунок 1 – принцип обработки запросов приложением

Существует четыре общих уровня веб-приложений:

– Представления (PL) имеет компоненты пользовательского интерфейса, которые показывают данные для пользователей, также компоненты пользовательского процесса, которые задают взаимодействие с пользователем. PL предоставляет всю необходимую информацию клиентской стороне. Основная цель уровня представления - получить входные данные, обработать запросы пользователей, отправить их в службу данных и показать результаты.

– Слой бизнес-логики BLL несет ответственность за надлежащий обмен данными между уровнем представления PL и уровнем обмена данными DAL, определяет логику бизнес-операций и правил.
– Службы данных DSL передает данные, обработанные уровнем бизнес-логики, на уровень представления. Этот уровень гарантирует безопасность данных, изолируя бизнес-логику со стороны клиента.
– Доступа к данным DAL предлагает упрощенный доступ к данным, хранящимся в постоянных хранилищах (например XML). Уровень доступа к данным также управляет операциями CRUD - создание (C), чтение (R), обновление (U), удаление (D).

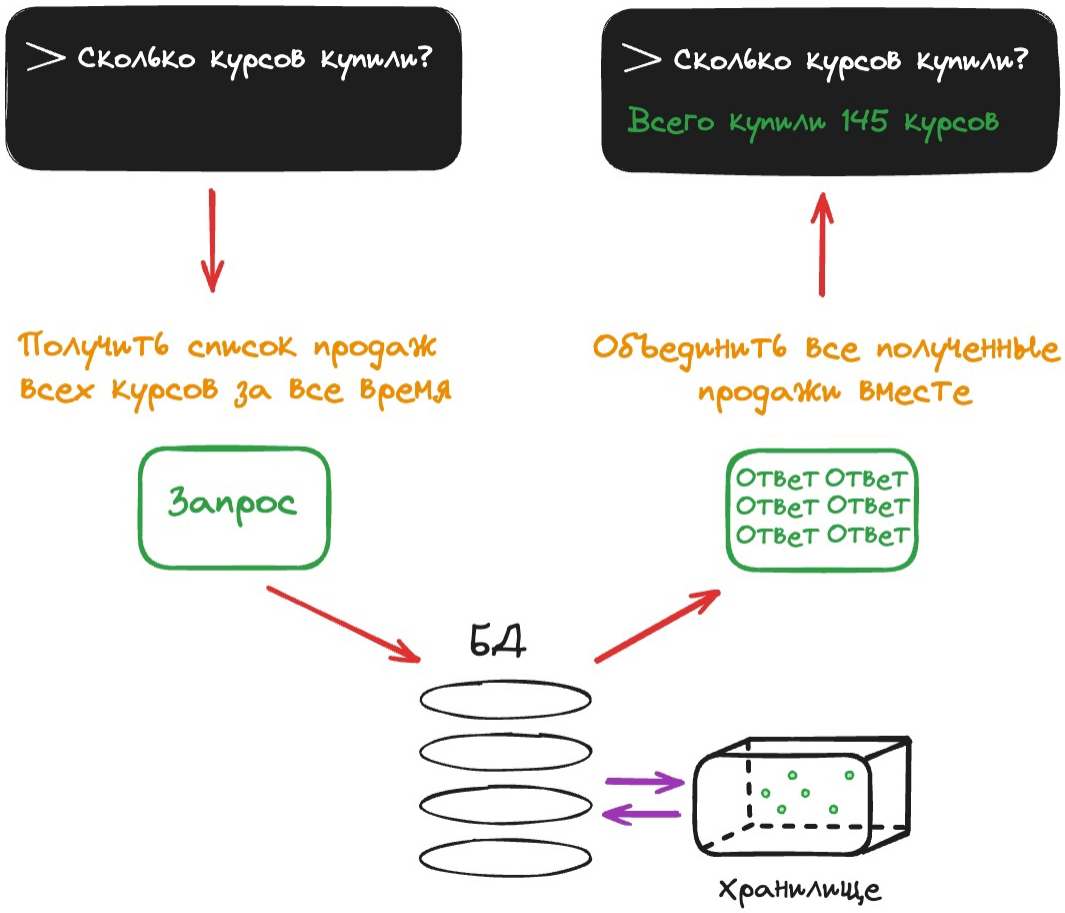

Рисунок 2 – принцип взаимодействия уровней приложения

Чтобы приложение корректно и безопасно функционировало, необходимо, чтобы на каждом уровне была обеспечена безопасность передаваемых

данных и минимизирована возможность использования особенностей каждого уровня, в качестве канала получения информации.

1.2 Уязвимости

При работе с фреймворком разработчик используя функционал чаще всего ограничивается знанием смысла работы какой-либо функции и ее параметров, не заглядывая внутрь нее, не зная всех деталей ее работы. Это большой плюс для разработчика, потому что с помощью данной «абстракции», он может сконцентрироваться на решение своей задачи, не углубляясь в ненужные подробности.

Большинство разработчиков не думает о безопасности работы приложения и данных, которыми оно оперирует, и тем более не считают это основной задачей, поэтому, если рассматривать такое качество фреймворка как безопасность, то тем он будет лучше, чем больше будет предусматривать и закрывать возможные дыры в безопасности – уязвимости.

На момент 2024 года такая организация как OWASP (Открытый проект обеспечения безопасности web-приложений) выделила следующие популярные уязвимости:

- нарушение контроля доступа;
- недочёты криптографии;
- инъекции;
- небезопасный дизайн;
- небезопасная конфигурация;
- использование уязвимых или устаревших компонентов;
- ошибки идентификации и аутентификации;

Нарушение контроля целостности

Это набор уязвимостей, при которых система плохо контролирует уровни доступа к информации или к своей функциональности. Из-за этого злоумышленники могут пользоваться функциями, к которым не должны иметь доступа. Если веб-приложение, где каждая учётная запись имеет разные права доступа слабо защищено, злоумышленник может модифицировать запросы или параметры URL, чтобы получить доступ к данным, на которые у него нет права.

Недочеты криптографии

Это уязвимости, связанные с неправильной настройкой и использованием криптографических методов для защиты данных. Это может быть недостаточная длина ключей, ненадёжные условия их хранения, использование устаревших алгоритмов и другие ошибки в криптографической реализации.

Инъекции

Данный вид уязвимостей – пользовательский ввод с вредоносным кодом. Инъекции позволяют злоумышленникам внедрять свой вредоносный код на сервер и выполнять его. Результат – потеря данных, кража данных или повреждение системы.

Небезопасный дизайн

Широкая категория уязвимостей, впервые появившаяся в последней версии OWASP Top Ten. Уязвимости этой категории возникают потому, что сама логика работы приложения может позволять использовать существующие функции в качестве уязвимостей.

Небезопасная конфигурация

Это случаи, когда настройки приложения, сервера, базы данных или других компонентов системы не являются безопасными. К этому можно отнести ненадёжные или отсутствующие настройки аутентификации, авторизации и доступа, например отсутствие защиты от перебора пароля.

Использование уязвимых компонентов

К этому типу уязвимостей относят случаи, когда веб-приложение использует сторонние фреймворки, библиотеки, плагины или другие компоненты, которые имеют выявленные дефекты безопасности. У злоумышленников даже есть автоматизированные инструменты, которые помогают находить неправильно сконфигурированные системы

Ошибки идентификации и аутентификации

Слабые пароли, недостаточная проверка подлинности, неэффективные системы учёта сеансов, все ошибки, которые могут возникнуть из-за недостаточно безопасной реализации идентификации и аутентификации пользователя в системе.

Можно ли предотвратить реализацию большей части этих уязвимостей, если при разработке использовать определенные фреймворки, а может вообще что-то может быть обработано на уровне языка. В последующей своей работе, я рассмотрю самые популярные фреймворки для разработки web-приложений и попробую проанализировать, какие базовые критерии безопасности они предусматривают.

2 Фреймворки для создания Web

Фреймворк – это динамически пополняемая библиотека языка программирования, в которой собраны его базовые модули. Фреймворки создаются для упрощения процессов разработки приложений, сайтов, сервисов. Чтобы не писать модуль в приложении с нуля, гораздо проще обратиться к готовым шаблонам фреймворков, которые и формируют рабочую среду разработчика.

Архитектура почти всех фреймворков основана на декомпозиции нескольких отдельных слоев (приложения, модули и т. д.) проекта. Это означает, что можно расширять функциональность приложения исходя из потребностей и использовать измененную версию вместе с кодом фреймворка или задействовать сторонние приложения. Такая гибкость является одним из одним из ключевых преимуществ использования фреймворков.

Рассмотрим в общем несколько популярных фреймворков и их особенности.

2.1 ASP.NET (.NET)

Учитывая популярность в свое время технологий .NET, стоит поговорить про такой фреймворк, как ASP.NET – это набор технологий в составе .NET Framework, которые позволяют создавать web-приложения и сервисы на основе Microsoft.NET с использованием любых поддерживаемых ей языков. В отличие от web-страниц, которые представляют собой сочетание статического HTML и сценариев, платформа использует скомпилированные страницы, которые управляются событиями. Но в отличие от десктопных приложений, эти скомпилированные страницы создают информацию, отправляемую клиентам с использованием языков разметки наподобие HTML и XML. Это позволяет разработчикам создавать приложения, защищая при этом интерфейс пользователя под управлением разных операционных систем.

Основная концепция безопасности, которая есть в .NET – прежде всего это ролевая модель безопасности. Она подразумевает два основных режима работы. Первый это создание пользователей и ролей, которые не зависят от ролей ОС Windows. Такая модель удобна, когда все разграничение прав внутри приложения ведется именно с помощью ролей. Все это никоим образом не связано с учетными записями в ОС. Например, это доступность каких-либо компонентов веб приложения в зависимости от заданной роли пользователя.

Второй режим – жесткая привязка ролей в приложении к учетным записям в Windows. Обычно подобную модель безопасности можно встретить в веб-приложениях, работающих во внутренней ё 2среде и тесно связанных с инфраструктурой Active Directory.

Аутентификация в ASP.NET приложениях обычно реализуется или с помощью аутентификации Windows или с помощью форм. Первый вариант построен на использовании штатных средств операционной системы . В каждом случае пользователь предъявляет некий аналог “удостоверения” – в первом случае это SID (Security Identifier), а во втором случае формируется так называемый билет, который затем сохраняется в cookie.

Для работы с базой данных используется объектно-ориентированная технология доступа к данным – Entity Framework (EF), которая является object-relational mapping (ORM) решением для .NET Framework от Microsoft. Изначально с самой первой версии Entity Framework поддерживал подход Database First, который позволял по готовой базе данных сгенерировать edmx модель данных файла. Затем эта модель использовалась для подключения к базе данных. Позже был добавлен подход Model First. Он позволял создать вручную с помощью визуального редактора модель, и по ней создать базу данных. Предпочтительным подходом стал Code First, в котором сначала пишется код модели, а затем по нему генерируется база данных.

Положительные стороны:

- Возможность разработки крупных проектов со сложной архитектурой;
- Совместимость с несколькими операционными системами;
- Разработка на разных языках;

Отрицательные стороны:

- Проблемы с одновременным доступом большого числа пользователей;
- Используется компиляция, при незначительных нагрузках сервисы работают медленнее;
- При разработке приложений используются лицензированные инструменты, что приводит к удорожанию продукта.

2.2 Laravel (Php)

На конец 2023 года, по статистике GitHub, Php занимает 7 место по популярности среди всех существующих языков разработки[6]. Так или иначе этот язык присутствует в 79,2 % от общего числа всех вебсайтов в интернете,

не удивительно, ведь он стоял у истоков интернета. Один из популярнейших фреймворков Php – Laravel считается довольно простым для входа среди всех Php-фреймворков, т.к делает процесс разработки веб-сайтов проще и быстрее из-за простой обработки кода, и множества встроенных модулей. Что важно фреймворк содержит пакет безопасности, который позволяет повысить защищенность приложения.

Внутри фреймворка Laravel есть свой ORM – Eloquent. В библиотеке ORM помимо стандартных CRUD-операций можно выделить наличие методов доступа, мутаторов, безопасное удаление, направления областей запросов, построения отношений, построение взаимодействия на основе событий. Eloquent организует работу с базой через представление таблиц в виде «моделей», через которы и осуществляется доступ и манипуляции с данныими.

Положительные стороны:

- простая интеграция платежных шлюзов;
- встроенные пакеты шифрования, основанные на OpenSSL в системе алгоритмов AES-256 и AES-128;
- наличие встроенных шаблонизаторов Blad;
- частая обновляемость и поддержка.

Отрицательные стороны:

- большой объем файлов и зависимостей, что может навредить производительности;
- несовместимость обновлений фреймворка, что может привести к конфликтам внутри проекта.

Некоторые пункты, которые реализует фреймворк в плане безопасности.

Токены CSRF

Laravel автоматически генерирует «токен» CSRF для каждой активной пользовательской сессии, управляемой приложением. Он используется для проверки, что аутентифицированный пользователь не является злоумышленником.

Защита от XSS

Laravel автоматически экранирует выводимые данные, чтобы предотвратить XSS-атаки. Чтобы безопасно выводить данные, используется Blade-синтаксис (шаблонизатор).

Защита от SQL

Существует несколько различных способов защиты от SQL-инъекций.

2.3 Django (Python)

Высокоуровневый фреймворк, который является не только быстрым решением в веб-разработке, включающим все необходимое для качественного кода и прозрачного написания, но и также удобным для разработчиков. В Django реализован принцип DRY — Don't Repeat Yourself (не повторяйся). То есть при использовании Django не нужно несколько раз переписывать один и тот же код. Фреймворк позволяет создавать сайт из компонентов. Благодаря этому сокращается время создания сайтов.

Фреймворк справляется с большим количеством задач и повышенными нагрузками, также подходит для создания алгоритмических генераторов, платформ для электронных рассылок, систем верификации, платформ для анализа данных и сложных вычислений, машинного обучения. У фреймворка есть своя ORM, которая автоматически передает данные из БД в объекты, которые используются в коде приложения, включает механизмы предотвращения распространенных атак вроде SQL-инъекций и подделки межсайтовых запросов.

Положительные стороны:

- масса различных библиотек;
- подробная документация и очень развитое сообщество;
- возможность масштабирования по мере необходимости.

Отрицательные стороны:

- нет поддержки WebSockets;
- Django ORM сегодня значительно уступает последней SQLAlchemy.

Некоторые пункты, которые реализует фреймворк в плане безопасности.

Внедрение SQL

Это предотвращается благодаря API QuerySet, который не делает никаких действий с базой, пока набор запросов к ней не будет обработан.

Параметризация

Он параметризует запросы и абстрагируется от разработчиков. Готовые шаблоны сами по себе защищают от атак с использованием межсайтовых сценариев.

Защита паролей

Используется функция защиты паролей PBKDF2.o функция получения ключа, разработанная RSA Laboratories, используемая для получения стойких ключей на основе хеша. Она работает путем применения псевдослучайной хеш-

функции к паролю и повторение этого процесса большое количество раз.

Защита от CSRF

По умолчанию промежуточное ПО CSRF активировано в MIDDLEWARE настройках (CSRF защита теперь является частью ядра Django), а также можно использовать его методы для определенных представлений.

3 Практическая часть

3.1 Критерии оценки безопасности фреймворков

Может получится много ситуаций, когда приложение, его компоненты или язык на котором это все написано, ведут себя непредсказуемо, это может быть связано с неправильной обработкой символов, отсутствием проверки каких-либо данных, передачи информации без надлежащей защиты ее или самого сеанса, поэтому попробую сформулировать некоторые критерии для фремворка web-приложения, которые помогут оценить на сколько в плане безопасности продуман функционал.

3.2 ASP.NET (.NET)

3.3 Laravel (Php)

3.4 Django (Python)

3.5 Сравнение защищенности

Фреймворки Критерии	ASP.NET	Laravel	Django
Правильная обработка символов Юникода	+	+	+
Влияние нулевых байтов	+	+	+
Поддержка параметризованных запросов SQL	+	+	+
Наличие способа разделения данных и HTML	+	+	+
Безопасность реализации сеанса, безопасный доступ к ОРМ	+	+	+
Безопасность механизма аутентификации	+	+	+

Фреймворки Критерии	ASP.NET	Laravel	Django
Нормализация пути	+	+	+
Безопасные варианты хранения	+	+	+
Защита от инъекций строки для записи безопасных журналовSQL	+	+	+
Функция применения белого списка на входах шаблонизаторы, опасные теги	+	+	+

ЗАКЛЮЧЕНИЕ

спать надо

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Андриянов В. В., Зефиров С. Л., “ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ БИЗНЕСА”, г. Москва, Издательство Альпина Паблишер, 2011 г., Яз. Рус.

2 Груздев С., Сабанов А., “Вопросы идентификации и аутентификации при разработке мультиаппликационных карт”, [Электронный ресурс] / URL: https://www.aladdin-rd.ru/company/pressroom/articles/voprosy_identifikacii_i_autentifikacii_pri_razrabotke_multiapplikacionnyh_kart, дата обращения 01.07.2020, Яз. Рус.

3 Коржов В., “Пароль на минуту”, [Электронный ресурс] / URL: https://www.osp.ru/cw/2005/01/84855/, дата обращения 01.07.2020, Яз. Рус.

4 Сабанов А., “О технологиях идентификации и аутентификации”, [Электронный ресурс] / URL: https://www.aladdin-rd.ru/company/pressroom/articles/o_tehnologiah_identifikacii_i_autentifikacii, дата обращения 01.07.2020, Яз. Рус.

5 Фисенко Л., “Новое лицо киберпреступности”, [Электронный ресурс] / URL: https://www.itweek.ru/infrastructure/article/detail.php?ID=71779, дата обращения 01.07.2020, Яз. Рус.

6 PHP Framework List: An Ultimate Guide to 102 PHP Frameworks for Web Developers [Электронный ресурс]. URL: https://www.temok.com/blog/php-framework-list/ (дата обращения: 25.12.2022).