Колледж Автономной некоммерческой образовательной организации высшего образования

«Научно-технологический университет «Сириус»

**Реферат по предмету введение в специальность**

**«Обеспечение безопасности веб-сайтов»**

Работу подготовил:

Студент группы К0709-24/1

Василевский О. В.

Преподаватель:

Яковлева С. В.

Оглавление

**Элементы оглавления не найдены.**

**Введение**

Представьте, что у вас есть свой магазин. Вы продаете что-то важное: товары, информацию, услуги. Вы бы хотели, чтобы все покупатели чувствовали себя в безопасности, правда? Чтобы никто не украл товары, не обманул других посетителей и не сломал кассу. Вот и с веб-сайтами так же! Они, как наши онлайн-магазины, и нуждаются в защите.

Раньше, когда интернет был совсем новым, мало кто думал о безопасности. Но со временем, появились хакеры. Они начали взламывать сайты, чтобы воровать данные, распространять вирусы или просто хулиганить. Один известный случай: в 2000 году подросток взломал Yahoo!, один из самых популярных сайтов того времени, и остановил его работу на несколько часов! Это показало всем, насколько уязвимы веб-сайты и как важно заниматься их безопасностью.

Сегодня, когда мы делаем покупки онлайн, читаем новости или общаемся в социальных сетях, мы доверяем веб-сайтам свою личную информацию, данные банковских карт и многое другое. Поэтому, обеспечение безопасности веб-сайтов – это не просто “хорошо бы”, это обязательно!

В этом докладе мы разберемся, какие опасности подстерегают веб-сайты и как программирование помогает нам создавать надежную защиту. Мы узнаем, как “строить стены” и “ставить замки”, чтобы защитить наши онлайн-магазины от “воров” и “хулиганов”. Понимание этих основ безопасности необходимо каждому, кто делает или поддерживает веб-сайты, чтобы защитить данные пользователей и сохранить их доверие.

**Основные угрозы безопасности веб-сайта**

Как и в реальном мире, в интернете есть свои “плохие парни”, которые хотят навредить веб-сайтам. Давайте разберем, какие “инструменты” они используют:

1. **SQL Injection (SQLi):** По сути, это когда кто-то может подменить заказ в ресторане и получить бесплатно еду! Но это не просто мелкое хулиганство. Хакеры используют SQLi, чтобы получить доступ к самой ценной информации на сайте – базе данных. Они вставляют специальные команды в поля ввода, например, в форму логина, поиска или в комментариях, обманывая сайт и заставляя его выполнять вредоносные действия. В результате, хакер может увидеть чужие пароли, номера кредитных карт, адреса и телефоны пользователей, а также изменить или удалить любую информацию в базе данных.
2. **XSS (Cross-Site Scripting) или межсайтовый скриптинг:** Представьте, будто кто-то пишет гадость на стене вашего магазина, и все посетители это видят. Но XSS – это гораздо больше, чем просто вандализм. Хакеры добавляют вредоносный код, чаще всего JavaScript, на страницу сайта. Этот код может выглядеть как безобидная картинка или кнопка, но на самом деле он крадет данные пользователей. Важно понимать, что XSS – это не одна атака, а целое семейство. **Отраженный XSS** – самый простой вид, где вредоносный код отправляется в запросе (например, в URL) и немедленно отображается на странице. Вы переходите по ссылке, а там внезапно появляется сообщение, требующее ввести ваш пароль, чтобы “проверить безопасность аккаунта”. **Хранимый XSS** гораздо опаснее: вредоносный код сохраняется на сервере (например, в комментариях, в профиле пользователя или в сообщениях на форуме) и выполняется каждый раз, когда страница загружается. Представьте, что вы заходите на форум, и ваш компьютер заражается вирусом просто из-за того, что кто-то оставил там вредоносный комментарий. И, наконец, **DOM-based XSS** – самый сложный вид, где вредоносный код изменяет структуру страницы (DOM) в браузере пользователя. Это как если бы хакер мог переписать часть кода вашего магазина прямо в вашем браузере.
3. **Broken Authentication:** Подумайте об этом, как если бы кто-то подобрал ключи к вашему магазину. Или еще хуже – кто-то просто оставил дверь открытой! Если на сайте используются слабые пароли (например, “123456” или “password”), если нет защиты от перебора паролей (хакер может пытаться угадывать пароль, и никто его не остановит).
4. **Cross-Site Request Forgery (CSRF):** В переводе на русский – это подделка межсайтовых запросов. Представьте себе ситуацию, когда кто-то, притворившись вами, отправляет письма от вашего имени. Но в интернете это может быть намного опаснее. Хакеры используют CSRF, чтобы заставить вас совершить действия на сайте, даже если вы об этом не знаете. Например Хакер отправляет вам ‘заражённую’ ссылку. Атака работает, потому что сайт доверяет вашему браузеру (вы же авторизованы!), и не может отличить запрос от вас от запроса, отправленного хакером. Защита от CSRF требует использования специальных токенов и других методов.
5. **Security Misconfiguration:** Здесь можно провести параллель с ситуацией, когда вы оставили дверь в магазин открытой или забыли установить сигнализацию. Security Misconfiguration – это когда веб-сервер и приложения настроены неправильно, что создает уязвимости, которыми могут воспользоваться хакеры. Это может включать использование устаревших версий программного обеспечения с известными дырами в безопасности, включенные отладочные режимы, которые позволяют видеть код и конфигурацию сайта, неправильные права доступа к файлам и папкам, оставленные открытыми порты, которые используются для тестирования, а затем забываются закрыть. Правильная настройка безопасности – это критически важный шаг для защиты веб-сайта.

**Роль валидации входных данных**

Валидация входных данных — это процесс анализа вводимых данных и исключения тех из них, которые считаются непригодными.

**Почему это так важно?**

1. Целостность и точность данных. Проверка вводимых данных гарантирует, что данные, поступающие в систему, являются точными, достоверными и согласованными. Это очень важно для поддержания целостности баз данных и предотвращения непреднамеренных ошибок.
2. Предотвращение атак с внедрением кода. Без надлежащей проверки ввода злоумышленники могут внедрить вредоносный код в систему, что может привести к разрушительным последствиям. К ним относятся SQL-инъекции, межсайтовый скриптинг (XSS) и другие атаки на основе кода.
3. Валидация вводимых пользователем данных помогает выявить и устранить уязвимости безопасности до того, как они могут быть использованы. Это является эффективной опережающей мерой против потенциальных нарушений.
4. Защита от переполнения буфера. Проверка длины и формата вводимых данных позволяет предотвратить переполнение буфера - распространенную атаку, при которой злоумышленник использует память программы, переполняя буфер избыточными данными.

**Защита от SQL injection**

1. Параметризация данных. Обычный запрос работает так, что пользователь вводит в строку поиска «лапки» и нажимает искать. Далее браузер запрашивает страницу поиска и передаёт на сервер строку «лапки». Работающее на сервере веб-приложение формирует запрос к базе данных. Для этого в приложение программистом заложена команда на SQL, задающая условия поиска. В неё и подставляется полученное от пользователя слово «лапки». База данных ищет подходящие под условия строки в таблице товаров и выдаёт их веб-приложению. После веб-приложение объединяет результат поиска с шаблоном страницы и наконец выдаёт результат браузеру.

Запрос с параметрами обрабатываются так, что в тексте запроса передаётся только имя параметра, а не его значение. Сами значения параметров передаются отдельно и ни в какой момент в текст запроса не вписываются, именно поэтому код с «инъекцией» не сработает. (habr.com, skillbox.ru)

1. Защита с помощью ORM. ORM – объектно-реляционное отображение, проще говоря – инструмент для оптимизации работы с базами данных. Данные фреймворки проверяют и санитируют пользовательский ввод от потенциально вредоносных символов или команд SQL, но стоит сказать о том, что ORM не защитит вас от скрытых инъекций, если параметризированный SQL не будет использоваться на протяжении всего написания сайта.
2. Минимальные привилегии. Суть принципа в том, что учётные записи базы данных, используемые приложением, имеют только необходимые права доступа. Это например, не давать привилегий на запись, обновление и удаление информации из базы данных, даже если был получен доступ на чтение. Проще говоря, сократить список доступных действий. Это облегчает определение любых действий, которые лежат вне возможностей пользователей. Таким образом, ограничение прав доступа к базе данных может значительно снизить последствия успешной атаки.

**Защита от XSS атак**

1. Экранирование данных. Как мы помним, xss атака основана на взаимодействии с сайтом, точнее куда мы можем ввести свой комментарий, туда же мы можем ввести вредоносный скрипт. Чтобы предотвратить это, вам нужно экранировать вводимые пользователем данные. Экранирование означает, что вы преобразуете ключевые символы данных, чтобы они не интерпретировались в опасном контексте.