Речь на защиту курсовой

**2 слайд**

С ростом популярности и вседоступности интернета возросло количество web-сайтов следовательно число пользователей этих интернет ресурсов возросло в несколько тысяч раз. Большое количество людей пользуются десятками web-сайтами ежедневно. Web-сервисы постоянно собирают информацию о своих пользователях: от персональных данных при регистрации до информации о действиях пользователя на сайте.

**3 слайд**

Для разработки комплекса мер безопасности web-сайта необходимо определить актуальные проблемы безопасности, возможные уязвимости web-ресурсов и виды угроз, то есть способы, которыми злоумышленник может воспользоваться, чтобы заполучить конфиденциальную информацию.

**Информационная безопасность основывается на следующих принципах:**

- Целостность информационных данных. способность информации сохранять изначальный вид и структуру в процессе хранения и после неоднократной передачи. Право на изменение, удаление или дополнение информации имеет только владелец или пользователь с легальным доступом к данным.

- Конфиденциальность данных. необходимость ограничения доступа к информационным ресурсам для определенного круга лиц. В процессе действий и операций информация становится доступной только пользователям, которые включены в информационную системы и успешно прошли идентификацию.

- Доступность информационных ресурсов. Информация, находящаяся в свободном доступе, должна предоставляться полноправным пользователям ресурсов своевременно и беспрепятственно.

- Достоверность. Данный принцип указывает на принадлежность информации доверенному лицу или владельцу, который одновременно выступает в роли источника информации.

**4 слайд**

**Можно выделить следующие основные виды уязвимостей сайтов:**

- SQL-инъекции. SQL-инъекция – это метод внедрения кода, который может разрушить базу данных. Размещение вредоносного кода в операторах SQL происходит через ввод web-страницы;

- XSS (межсайтовый скриптинг). XSS-атаки позволяют злоумышленникам внедрять клиентские скрипты в web-страницы, просматриваемые другими пользователями, данная уязвимость межсайтового сценария может использоваться злоумышленниками для обхода средств контроля доступа;

- незащищенная работа с персональной информацией. Данная уязвимость проявляется в хранении особо важной информации (например, паролей пользователей) в базе данных в открытом виде, а также в незащищенности трафика клиентов;

- проблемы распределения уровней доступа. Суть данной проблемы заключается в том, чтобы пользователь не имел доступ к особой служебной информации сайта;

- плохая конфигурация модулей. После установки программного обеспечения на сервер, необходимо настроить отдельные модулей для предотвращения несанкционированного доступа.

- брутфорс-атака. Данная атака представляет собой метод подбора пароля, путем поочередного перебора возможных комбинаций;

- нарушенная аутентификация и управление сеансом. Некорректная аутентификация может позволить злоумышленнику использовать различные методы как ручные, так и автоматические, чтобы попытаться получить контроль над любой учётной записью в системе или полный контроль над системой;

- подделка межсайтовых запросов. Подделка межсайтовых запросов подразумевает поддельный запрос, пришедший с перекрестного сайта.

**5 слайд – аналитика**

**Было проведено Исследование проблем безопасности web-сайтов. Аналитика acunetix**

Уязвимость - это недостаток в приложении или устройстве, который может быть использован злоумышленниками. Злоумышленники могут использовать уязвимость для достижения такой цели, как кража конфиденциальной информации, взломать систему, сделав ее недоступной (в сценарии отказа в обслуживании), или повредить данные.

По степени тяжести уязвимости делятся на высокую степень серьезности, среднюю и низкую.

**Высокая степень серьезности** указывает на то, что **злоумышленник может полностью нарушить конфиденциальность, целостность или доступность системы** без специального доступа, взаимодействия с пользователем или обстоятельств, которые находятся вне контроля злоумышленника.

**Уровень средней степени тяжести** указывает на то, что **злоумышленник может частично нарушить конфиденциальность, целостность или доступность целевой системы**. Им может потребоваться специальный доступ, взаимодействие с пользователем или обстоятельства, не зависящие от злоумышленника. Такие уязвимости могут использоваться вместе с другими уязвимостями для эскалации атаки.

**Низкая степень серьезности означает**, что **злоумышленник может ограниченным образом нарушить** конфиденциальность, целостность или доступность целевой системы. Им нужен специальный доступ, взаимодействие с пользователем или обстоятельства, не зависящие от злоумышленника. Для расширения атаки такие уязвимости необходимо использовать вместе с другими уязвимостями.

**7 слайд – как защищать**

**SQL-инъекция** - это тип уязвимости безопасности web-приложений, при которой злоумышленник пытается использовать код приложения для доступа или повреждения содержимого базы данных. В случае успеха это позволяет злоумышленнику создавать, читать, обновлять, изменять или удалять данные, хранящиеся во внутренней базе данных.

1 использовать подготовленные операторы с параметризованными запросами. Они гарантируют, что ни одна из динамических переменных, которые нужны в запросе, не сможет выйти из своей позиции. Предварительно определяется основной запрос, а после него указываются аргументы и их типы. Поскольку запрос знает тип ожидаемых данных, то он точно знает, как интегрировать их в запрос, не вызывая проблем.

2 использовать хранимые процедуры. Хранимые процедуры – это частые операции SQL, которые хранятся в самой базе данных и различаются только своими аргументами. Хранимые процедуры значительно усложняют злоумышленникам выполнение своего вредоносного SQL-кода, поскольку его нельзя динамически вставлять в запросы.

3 проверять ввод разрешенного списка. Нужно выполнить проверку списка разрешений, чтобы проверить вводимые пользователем данные по сравнению с существующим набором известных, утвержденных и определенных вводов. Если полученные данные не соответствуют заданным значениям, то они отклоняются.

**Cross-site Scripting (XSS)** – это атака путем внедрения кода на стороне клиента.

Для реализации внедрения вредоносного кода уязвимый web-сайт должен напрямую включать пользовательский ввод на свои страницы. Затем злоумышленник может вставить вредоносную строку, которая будет использоваться на web-странице и обрабатываться браузером жертвы как исходный код.

Концепция межсайтового сценария основана на том, что **небезопасный пользовательский ввод напрямую отображается на web-странице**. Если вводимые пользователем данные должным образом дезинфицированы, атаки с использованием межсайтовых сценариев будут невозможны.

1 следует ограничить ввод пользователя определенным списком разрешений. Такая практика гарантирует, что на сервер отправляются только известные и безопасные значения.

2 нужно ограничивать HTML во входных данных. Если используется на странице контент, созданный пользователями, нужно убедиться, что он не приведет к созданию HTML-контента путем замены небезопасных символов соответствующими объектами. Сущности имеют тот же вид, что и обычные символы, но не могут использоваться для создания HTML.

3 использовать флаги HTTPOnly для файлов cookie. Сеансовые файлы cookie - это механизм, который позволяет web-сайту распознавать пользователя между запросами, и злоумышленники часто крадут сеансы администратора, отфильтровывая свои файлы cookie. После кражи cookie злоумышленники могут войти в свою учетную запись без учетных данных или авторизованного доступа. Для запрета чтения JavaScript содержимого файла cookie можно использовать файлы cookie HttpOnly, что затруднит злоумышленнику захват сеанса.

8 слайд

**Уязвимости подделки межсайтовых запросов** (CSRF) возникают, когда web-сервер получает неавторизованный запрос от доверенного браузера. Запросы браузера, отправленные на web-сервер, могут включать файлы cookie сеанса пользователя - это почти всегда происходит, если пользователь уже вошел на сайт. представляет собой атаку, при которой злоумышленник обманом заставляет жертву выполнять действия от своего имени.

Подделка межсайтовых запросов будет эффективна только в том случае, если жертва аутентифицирована. Это означает, что для успешной атаки жертва должна войти в систему. Поскольку CSRF-атаки используются для обхода процесса аутентификации, могут быть некоторые элементы, на которые эти атаки не влияют, это общедоступный контент. CSRF применяется только к ситуациям, когда жертва может выполнять действия, доступные не всем.

1 использование токена вызова, который связан с конкретным пользователем и отправляется как скрытое значение в каждой форме изменения состояния в web-приложении. Этот токен, называемый токеном анти-CSRF, работает следующим образом:

- web-сервер генерирует токен и сохраняет его;

- токен статически установлен как скрытое поле формы;

- форма отправлена пользователем;

- токен включен в данные POST-запроса;

- приложение сравнивает токен, сгенерированный и сохраненный приложением, с токеном, отправленным в запросе;

- если эти токены совпадают, запрос действителен;

- если эти токены не совпадают, запрос недействителен и отклоняется.

Этот метод защиты CSRF называется шаблоном токена синхронизатора. Он защищает форму от атак с подделкой межсайтовых запросов, поскольку злоумышленнику также необходимо угадать токен, чтобы успешно обманом заставить жертву отправить действительный запрос. Токен также должен стать недействительным через некоторое время и после выхода пользователя из системы.

2 криптографически безопасным. Токен нелегко угадать, поэтому он не может быть сгенерирован на основе предсказуемого шаблона.

3 Атаки CSRF возможны только потому, что **файлы cookie всегда отправляются с любыми запросами, которые отправляются в конкретный источник, связанный с этим файлом cookie**. Можно установить **флаг для файла cookie, который превращает его в файл cookie того же сайта**. Файл **cookie того же сайта - это файл cookie, который может быть отправлен только в том случае, если запрос сделан из источника, связанного с файлом cookie (не междоменным**). Считается, что cookie и источник запроса имеют одно и то же происхождение, если **протокол и хост (но не IP-адрес) одинаковы для обоих**. В настоящее время файлы cookie на том же сайте лучше подходят в качестве **дополнительного уровня защиты** из-за этого ограничения. Следовательно, вы должны использовать их только вместе с другими механизмами защиты от CSRF.

**Атака DDoS** проверяет пределы ресурсов web-сервера, сети и приложения, отправляя пики поддельного трафика. Некоторые атаки представляют собой просто короткие пакеты вредоносных запросов на уязвимые конечные точки, такие как функции поиска. DDoS-атаки используют целую армию зомби-устройств, называемых ботнетами. Эти ботнеты обычно состоят из взломанных устройств Интернета вещей, web-сайтов и компьютеров.

1 Использование брандмауэра. Большинство атак на web-сайты происходят из определенных стран, таких как Китай, Россия и Турция. Брандмауэр дает возможность заблокировать их взаимодействие (POST) с сайтом. Можно заблокировать некоторые страны, чтобы IP-адреса из этих стран по-прежнему могли просматривать весь контент на web-сайте, но не могли регистрироваться, отправлять комментарии или пытаться войти.

2 отслеживать трафик сайта, чтобы быть в курсе пиков трафика и DDoS-атак. Резкое увеличение коэффициента исходящего трафика является признаком DDoS-атак. Поэтому стоит пользоваться службами мониторинга трафика.

3 DDoS-атаки могут служить отвлечением для других атак, особенно когда злоумышленники разрабатывают атаки, нацеленные на финансовые учреждения. Кража данных может происходить, когда все взоры прикованы к DDoS-атаке. Поэтому, когда сайт подвержен DDoS-атаке, также стоит уделить внимание поиску других атак.

9 слайд – общие положения

Самый простой способ защитить web-сайт от SQL-инъекций это поддерживать программное обеспечение и компоненты в актуальном состоянии

Фильтруя потенциально опасные web-запросы, брандмауэры web-приложений могут перехватывать и предотвращать SQL-инъекции.

Также можно использовать WAF для защиты от атак межсайтового скриптинга.

Также рекомендуется использовать параметры защиты от CSRF в популярных фреймворках, таких как AngularJS, и воздерживаться от создания собственных механизмов. Это позволяет избежать ошибок и ускоряет и упрощает внедрение.

Для защиты от данной уязвимости следует придерживаться следующих правил:

- поддерживать осведомлённость о рисках, связанных с уязвимостями CSRF;

- использовать токены против CSRF;

- использовать файлы cookie SameSite;

- проводить регулярное сканирование сайта на уязвимость.

Активация брандмауэра web-приложения для защиты от DDoS-атак – это уровень защиты, который находится между сайтом и получаемым трафиком

10 слайд – тестирование

**Тестирование безопасности** - это стратегия тестирования, используемая для проверки безопасности системы, а также для анализа рисков, связанных с обеспечением целостного подхода к защите приложения, атак хакеров, вирусов, несанкционированного доступа к конфиденциальным данным.

**Чек лист проверки безопасности**

**Контроль доступа** – определяет проблемы, связанные с несанкционированным доступом пользователей к информации и функциям в зависимости от предоставленной роли. Тестирование конфигурации ролевой модели.

**Аутентификация** – позволяет удостовериться в отсутствии возможности обойти процедуру регистрации и авторизации;

**Валидация входных значений** – используется для проверки алгоритмов обработки данных, включая некорректные значения, прежде, чем на них будет ссылаться приложение.

**Криптография** – обнаруживает проблемы, связанные с шифрованием, дешифрованием, подписью, верификацией подлинности, в том числе включая уровень сетевых протоколов, работу с временным файлами и cookies.

**Механизмы обработки ошибок** – включает проверку системных ошибок приложения на отсутствие факта раскрытия информации о внутренних механизмах безопасности (например, посредством демонстрации исключений, программного кода).

**Конфигурация сервера** – ищет в многопоточных процессах ошибки, связанные с доступностью значений переменных для совместного использования другими приложениями и запросами.

**Интеграция со сторонними сервисами** – позволяет убедиться в невозможности манипуляции данными, передаваемыми между приложением и сторонними компонентами, например, платежными системами или соцсетями.

**Проверка устойчивости к Dos/DDos атакам** – проверяет способность приложения обрабатывать незапланированно высокие нагрузки и большие объемы данных, которые могут быть направлены на выведение приложения из строя.

11 слайд

Вот 8 инструментов с открытым исходным кодом, которые популярны среди тестировщиков:

**Vega** – это инструмент для сканирования и тестирования уязвимостей, написанный на Java. Он работает с платформами OS X, Linux и Windows. **содержит автоматический сканер и перехват прокси**. Позволяет обнаруживать как инъекции **SQL, вставка заголовков, межсайтовый скриптинг**

**ZED Attack Proxy (ZAP)** платформ Windows, Unix/Linux и Macintosh. Может применяться как сканер или перехватывать прокси для ручного тестирования веб-страницы.

**Wapiti** выполняет сканирование методом BlackBox и вводит полезные данные, чтобы проверить, уязвим ли сценарий. раскрытие файлов, включение файлов, межсайтовый скриптинг (XSS), слабую конфигурацию .htaccess

**W3af** – фреймворк для аудита веб-приложений и предотвращения атак, Инструмент для автоматизации процесса тестирования имеет графический интерфейс со средствами, которые могут использоваться для отправки HTTP-запросов и кластерных HTTP-ответов. Если сайт защищен, W3af может использовать модули проверки подлинности для сканирования.

**Iron Wasp** – Он может проверить сайт на наличие 25 видов веб-уязвимостей, обнаружить ложные срабатывания и ложные отрицания.

**SQLMap** — это средство автоматизации тестирования сайтов обнаруживает уязвимость в виде SQL-инъекций в базе данных сайта. Инструмент может напрямую подключаться к базе данных без использования SQL-инъекций. А также имеет функции для создания цифрового отпечатка базы данных;

**Google Nogotofail** – средство тестирования безопасности сетевого трафика. Проверяет приложение на наличие известных уязвимостей TLS/SSL и неправильных конфигураций. Сканирует SSL/TLS-шифрованные соединения и проверяет, являются ли они уязвимыми для атак типа «человек посередине». Может быть настроен как маршрутизатор, VPN-сервер или прокси-сервер;

**BeEF** (Browser Exploitation Framework) позволяет выявить слабые места приложения, используя уязвимости браузера. Данный инструмент автоматизации тестирования приложений использует векторы атак на стороне клиента для проверки безопасности.