Министерство образования Республики Беларусь
Учреждение образования
Белорусский государственный университет
информатики и радиоэлектроники

УДК

Лобах Владислав Александрович

Система защиты данных веб-приложения дистанционного обучения

АВТОРЕФЕРАТ

на соискание степени магистра технических наук
по специальности 1-45 80 02 Телекоммуникационные системы
и компьютерные сети

Научный руководитель
д.т.н., доцент Цветков В.Ю.

Минск 2020

ВВЕДЕНИЕ

Всемирная паутина – это давно уже не просто веб-страницы и веб-сайты, а полноценные информационно-коммуникационные системы с веб-интерфейсом. В минимальной конфигурации есть веб-сервер, операционная система веб-сервера, сервер БД и сетевая служба обновления веб-сайта. Все эти компоненты должны быть безопасными, поскольку в случае взлома любого из них злоумышленники могут получить доступ к данным.

Уязвимость – ошибка или недостаток в системе, используя который можно намеренно нарушить ее конфиденциальность, целостность или доступность.

Речь пойдет не только о защите самого веб-приложения, а и о уязвимостях веб-приложений, которые позволяют киберпреступникам получать доступ к расположенным за ним сетевым устройствам, серверам приложений и базам данных. Некоторые уязвимости известны только теоретически, другие же могут активно использоваться при наличии эксплойта – программного кода, использующего уязвимость в программном обеспечении и применяемого для проведения атаки на систему.

Уязвимости веб-приложений гораздо проще выявить, эффективнее использовать и при этом скрыть свое присутствие. Уязвимости могут эксплуатироваться не только из интернета, но и изнутри (внутренний нарушитель). Не всегда они являются дефектами разработки самого приложения, а часто возникают в результате проблем с безопасностью обработки информации, неправильных конфигураций, слабого администрирования и порой даже отсутствия осведомленности у лиц, ответственных за информационные технологии и их безопасность.

Целью магистерской диссертации является разработка технологии создания защищенных веб-приложений и поддержка системы управления информационной безопасностью, в том числе обеспечение постоянного информирования о новых проблемах безопасности.

Для выполнения данной цели в диссертации поставлены следующие задачи:

– провести анализ современных способов реализации веб-приложений;

– провести анализ методик и технологий разработки защищенных веб-приложений;

– разработка технологии создания защищенных веб-приложений;

– провести анализ эффективности защиты веб-приложения.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Основной целью данной магистерской диссертации является разработка технологии создания защищенных веб-приложений и поддержка системы управления информационной безопасностью, в том числе обеспечение постоянного информирования о новых проблемах безопасности. В ходе выполнения работы был проведен анализ информационных источников по способам построения современных безопасных приложений и анализ методик и технологий разработки защищенных веб-приложений, который позволил определить оптимальный набор технологий и средств, цели и задачи данной работы, а также разработать методику создания защищенных веб-приложений.

В первой главе проведен анализ существующих технических решений и технологий веб-приложений. Рассмотрена архитектура современных веб-приложений, проведен обзор уязвимостей и защитных мер. Также были рассмотрены принципы функционирования веб-приложений, основные типы атак и способов защиты.

Во второй главе подробно разработана методика создания защищенных веб-приложений на основании рассмотренных уязвимостей веб-приложений.

В третьей главе рассмотрены оценка эффективности разработанной методики создания защищенных веб-приложений, способы обнаружения уязвимостей, а также был выполнен поиск уязвимостей в тестовом веб-приложении.

Работа была апробирована на 54-й научно-технической конференции аспирантов, магистрантов и студентов.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Программное обеспечение в современном мире становится все сложнее, поскольку на него возлагается решение все более сложных задач. Коммерческие компании и государственные организации стремятся автоматизировать все больше своих процессов, как внутренних, так и тех, что связаны с общением с внешним миром. Необходимые при этом программные системы обычно должны поддерживать одновременную работу многих географически распределенных пользователей с централизованными и интегрированными хранилищами данных организаций. При этом должны обеспечиваться эффективность обработки запросов, высокая надежность работы, предоставление результатов в удобном пользователям виде, разграничение областей доступа и разрешенных операций для разных групп пользователей и т.д. Разработка таких приложений (в тех случаях, когда подходящей системы нет на рынке готовых продуктов), их внедрение и поддержка становятся все дороже из-за возрастающего количества предоставляемых ими функций, ограничений, которым они должны удовлетворять, составляющих их компонентов, возможных связей и взаимодействий между ними.

Есть, тем не менее, факторы, которые помогают значительно снизить расходы на создание подобных систем. Если программное обеспечение использует для связи между своими элементами базовые протоколы Интернет (TCP/IP и HTTP) и предоставляет пользовательский интерфейс в виде страничек HTML, которые можно просматривать в любом браузере, то практически каждый его потенциальный пользователь не имеет технических препятствий для обращения к этому ПО. Не нужно прокладывать сети, тратить усилия на настройку связи с серверами, разрабатывать различные клиентские компоненты для разных операционных систем, устанавливать клиентам специальное оборудование и программные компоненты и т.д. Интернет предоставляет готовую инфраструктуру для создания крупномасштабных программных систем, в рамках которых десятки тысяч компонентов работают совместно и миллионы людей пользуются предоставляемыми ими услугами.

Таким образом, возник особый класс программных систем – веб-приложения.

Веб-приложение – клиент-серверное приложение, основная часть которой содержится на удаленном сервере, а пользовательский интерфейс (UI) отображается в браузере в виде веб-страниц. Для запуска веб-приложения пользователю не нужно устанавливать никаких дополнительных программ, оно запускается на любом устройстве с браузером и с доступом в интернет. Работа

клиента не зависит от операционной системы, стоящей на компьютере пользователя, поэтому при разработке веб-приложений нет необходимости писать отдельные версии для Windows, Linux, Mac OS и других операционных систем. Для создания серверной части веб-приложений используются такие языки программирования, как: PHP, ASP, ASP.NET, Perl, C/C++, Java, Python, Ruby, NodeJS. Для реализации клиентской части используют HTML, CSS, JavaScript, Ajax.

Клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него. Серверная часть получает запрос от клиента, выполняет вычисления, после этого формирует веб-страницу и отправляет её клиенту по сети с использованием протокола HTTP.

Общая схема архитектуры веб-приложений заключается в следующем. Пользователь ищет в Google «Strong Beautiful Fog And Sunbeams In The Forest». Первый результат отправляет его на Storyblocks. Пользователь нажимает на ссылку, которая перенаправляет его браузер на страницу с изображением. Тем временем браузер отправляет запрос на DNS-сервер, чтобы установить соединение со Storyblock, и, получив ответ, отправляет запрос на сайт.

Запрос попадает на балансировщик нагрузки, который случайным образом выбирает для обработки запроса один из десяти (или около того) веб-серверов, на которых запущен сайт. Веб-сервер достаёт некоторую часть информации об изображении из службы кэширования, а остальное – из основной базы данных. Если цветовой профиль для изображения ещё не был вычислен, задача «определить цветовой профиль» будет добавлена в очередь заданий. Эти задания серверы обрабатывают асинхронно и соответствующим образом обновляют базу данных с результатами.

Затем происходит поиск похожих фотографий: в службу полнотекстового поиска отправляется запрос с заголовком фотографии в качестве входных данных. Если пользователь авторизовался в Storyblocks, информация о его учётной записи загружается из базы данных учётных записей. Наконец, данные о просмотре страницы отправляются в firehose-хранилище для последующей записи в облачную систему хранения. В конечном счёте, эти данные будут загружены в хранилище данных, которое аналитики используют для обработки.

Теперь сервер рендерит страничку в HTML и отправляет её обратно браузеру пользователя, проходя сначала через балансировщик нагрузки. Страница содержит Javascript- и CSS-файлы, которые загружены в облачную систему хранения, подключённую к CDN, поэтому браузер связывается с CDN для получения содержимого. Наконец, браузер отображает страницу пользователю.

SPA-приложения, Single Page Application, или «приложение одной страницы» – это тип веб-приложений, в которых загрузка необходимого кода происходит на одну страницу, что позволяет сэкономить время на повторную загрузку одних и тех же элементов.

Особенность архитектуры SPA заключается в том, что все элементы, необходимые для работы софта: элементы CSS, скрипты, стили и пр. на одной странице. Они загружаются при инициализации. Также данный вид приложений загружает дополнительные модули после запроса от пользователя. Любая пользовательская активность фиксируется для удобства навигации. Это позволяет скопировать ссылку и открыть софт на том же этапе взаимодействия на другой вкладке, браузере или устройстве.

При загрузке новых модулей в SPA контент на них обновляется только частично, так как элементам, не нуждающимся в изменении, нет необходимости загружаться повторно, замедляя тем самым скорость ответа и передаваемый объем данных между браузером и сервером.

SPA-приложения обладают рядом преимуществ:

- Доступность. Можно получить моментальный доступ к функционалу с любого типа устройства без проблем с совместимостью, достаточным объемом памяти, необходимыми вычислительными мощностями или с затратой времени на установку.

- Универсальность. Использовать софт можно практически с любого устройства, если на нем есть доступ к интернету. Если при разработке интерфейса учитывались различные разрешения экрана, то использовать SPA одинаково удобно и с ПК и со смартфона.

- Возможность задействовать большие объемы данных. Размер приложения и используемых им данных не ограничен памятью устройства.

- Скорость. Одна страница, содержащая весь необходимый интерфейс не только экономит время на повторную загрузку данных, но и повышает производительность работы с веб-приложением.

Рассмотрим основные атаки на веб-приложения:

– XSS атаки;

– SQL инъекция;

– отказ в обслуживании;

– Path Traversal;

– подделка межсайтовых запросов;

– Brute-force атака;

– PHP инъекция.

Cross-Site Scripting (XSS) – этот тип атаки на веб-приложения заключается во внедрении, в выдаваемую веб-системой страницу, вредоносного

кода. Когда пользователь открывает эту страницу, то встроенный код незамедлительно выполняется и взаимодействует с web-сервером злоумышленника.

SQL инъекция – это один из популярных способов взлома сайтов и программ, которые работают с базами данных. Атака основывается на внедрении в запрос произвольного SQL-кода. Внедрение SQL, в зависимости от типа используемой СУБД и условий внедрения, может дать возможность атакующему выполнить произвольный запрос к базе данных (например, прочитать содержимое любых таблиц, удалить, изменить или добавить данные), получить возможность чтения и/или записи локальных файлов и выполнения произвольных команд на атакуемом сервере.

Слепая SQL инъекция – еще один из видов атак. Атака немного сложнее обычной атаки и требует больше времени и сил. Результат SQL запроса не всегда отображается на сайте. Атака на форму входа является одним из примеров данной атаки.

Атака отказ в обслуживании (Denial of Service, DoS) – это атака на вычислительную систему. Все сетевые устройства имеют ограничения по количеству обрабатываемых запросов одновременно. Кроме этого, канал, через который сервер связывается с интернетом, также обладает ограниченной пропускной способностью.

Подделка межсайтовых запросов (CSRF) – вид атак на посетителей веб-сайтов, использующий недостатки протокола HTTP. Если жертва заходит на сайт, созданный злоумышленником, то от её лица тайно отправляется запрос на другой сервер (например, на сервер платёжной системы), осуществляющий некую вредоносную операцию (например, перевод денег на счёт злоумышленника).

Brute-force атака – атака на пароли методом полного перебора всевозможных вариантов с целью найти подходящий пароль.

Path Traversal – это атака направлена на получения доступа к файлам, директориям и командам, находящимся вне основной директории веб-сервера.

В результате анализа атак и уязвимостей были определены самые распространенные атаки, такие как: Path Traversal, DoS атака, SQL инъекция и XSS атака. На них нужно обратить внимание в первую очередь. Все атаки, кроме атаки отказ в обслуживании, направлены на хищение чувствительной информации.

Вышеуказанные методы защиты подходят и для нерассмотренных атак. Хостинг провайдер предоставит услугу резервного копирования и SSL сертификат.

Также рассмотренные инструменты и технологии позволяют спроектировать безопасное и качественное веб-приложение. Они обладают широким функционалом работы

Для обеспечения надежности работы предпочтительнее использовать современный javascript-фреймворк на стороне клиента, как Angular, React или Vue. В качестве сервера предпочтительно использовать облачную платформу, как Amazon Web Services (AWS) или Google Cloud, так как система должна быть распределенной и быстро реагировать на пользовательские запросы из любой точки мира. Также необходимо выбрать CDN.

Необходимо организовать безопасную передачу данных в веб-приложении, защитить пользователя от всевозможных атак и организовать безопасную авторизацию.

Аутентификация – это процесс проверки учётных данных пользователя (логин/пароль). Проверка подлинности пользователя путём сравнения введённого им логина/пароля с данными сохранёнными в базе данных [16].

Для аутентификации был выбран Token-Based Authentication алгоритм.

Token-Based Authentication использует подписанный сервером токен (bearer token), который клиент передает на сервер в заголовке Authorization HTTP с ключевым словом Bearer или в теле запроса.

ЗАКЛЮЧЕНИЕ

В ходе выполнения магистерской диссертации проведен анализ информационных источников по способам построения современных безопасных приложений и анализ методик и технологий разработки защищенных веб-приложений, который позволил определить оптимальный набор технологий и средств, цели и задачи данной работы, а также разработать методику создания защищенных веб-приложений.

Определены требования к разработке методики создания защищенного веб-приложения,

Разработана технология создания защищенных веб-приложений.

Также поддержана система управления информационной безопасностью, в том числе обеспечение постоянного информирования о новых проблемах безопасности.

Анализ тестового веб-приложения с помощью сканера показал, что следование представленной технологии создания безопасного веб-приложения обеспечивает отсутствие потенциальных уязвимостей.