Тема 1. Безопасность

Привет! Поговорим о такой важной теме, как безопасность веб-приложений, и обсудим, какие проблемы можно тут встретить. Ведь действительно, мы знаем уже более чем достаточно, чтобы написать такое приложение, которым смогут пользоваться люди. Самое трудное тут идея, но оставим её за скобками. И вот мы написали приложение, у нас есть пользователи, может быть, даже много пользователей, но одним печальным днём мы видим новость, что база данных с чувствительной информацией продаётся на каком-то сайте с трудночитаемым названием. И тут даже может наступить конец нашему бизнесу.

Очевидно, что чем больше вещей становятся «онлайн», тем проще злоумышленникам получить данные, не предназначенные им. Тут и переписки, и финансовые операции — всё это происходит через интернет. Огромное количество приватных данных отправляется туда-сюда по сети каждый день. И чтобы защитить себя и пользователей, мы, разработчики, должны в обязательном порядке предпринимать шаги для предотвращения подобных инцидентов. Владельцы сайтов несут ответственность за создание безопасной системы, так что давайте разбираться, какие атаки могут нас ждать и что мы можем с ними сделать. Сразу оговоримся, что далее речь пойдёт о безопасности API. Ведь заниматься обеспечением безопасности можно и нужно также и на стороне фронтенда, сети, железа.

И так получается, что новые атаки или уязвимости, о которых много пишут в новостях, — явления довольно редкие. Как правило, большинство брешей в безопасности системы приходится на список хорошо известных проблем. Есть несколько стандартов безопасности. Следуя этим стандартам, а вернее защищаясь от проблем, которые в этих стандартах описаны, мы можем защитить себя от большинства видов атак.

К сожалению, ситуация с безопасностью у многих веб-приложений оставляет желать лучшего. Да, все проблемы давно известны, хорошо описаны и все знают, как с ними бороться. Но по разным причинам системы всё ещё имеют множество уязвимостей. Вспомним истории, как некто обнаружил незапароленную базу данных какой-нибудь компании. И вишенкой на торте в ней хранили пароли пользователей чистым текстом. Обычно принято хранить не сами пароли, а хеши от них. То есть у нас есть некоторая функция по конвертации текста пароля в другую строковую последовательность, причём такая конверта работает только в одну сторону. И когда пользователь вводит свой пароль, то мы сравниваем именно эти оба хеша, а не сами пароли.

В общем, не боги горшки обжигают, и нашими маленькими делами по обеспечению стандартов безопасности мы можем сделать интернет чуточку безопаснее.

Один из таких стандартов — OWASP (Open Web Application Security Project) — известен своими списками рисков в разных программных технологиях. Помимо самого списка, у них на сайте представлены и пути преодоления этих проблем. Нас, как бэкенд-разработчиков, будет прежде всего интересовать список проблем в API.

Рассмотрим некоторые из этих проблем:

1. API1:2019 Broken Object Level Authorization:

   Недостаточный контроль доступа к объектам.

   Некоторые из наших эндпоинтов не закрыты нужными проверками на права доступа. В особенности те, что работают с пользовательскими данными и пользовательским вводом. Это создаёт опасную ситуацию, когда рядовой пользователь может получить админские права. Решение этой проблемы, к сожалению, выходит за рамки курса, но вы найдёте под видео ссылку на возможные пути решения.

2. API2:2019 Broken User Authentication

   Следующая проблема — механизм аутентификации выполнен с ошибками или вовсе отсутствует, что может позволить атакующим систему выдать себя за другого пользователя.

3. API4:2019 Lack of Resources & Rate Limiting

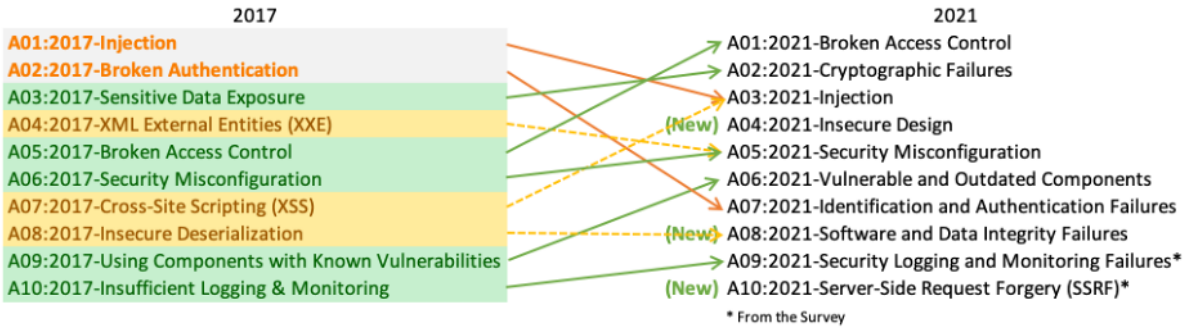
   Довольно часто API не накладывают никаких ограничений на размер или количество ресурсов, которые может запросить клиент. В итоге пользователь начинает делать слишком много запросов, на обработку которых у нас не хватает ресурсов. От этого начинают страдать другие пользователи. Чтобы избежать таких проблем, придётся уже очень хорошо думать над архитектурой системы.

4. API10:2019 Insufficient Logging & Monitoring

   Ещё одна интересная проблема: если мы недостаточно логируем события и мониторим состояние нашей системы, например долго не замечаем подозрительную нагрузку на базу данных, то это тоже уязвимость. Быстрое реагирование на проникновение может предотвратить кражу данных или другой возможный вред, а без должной системы мониторинга сделать это будет сложно.

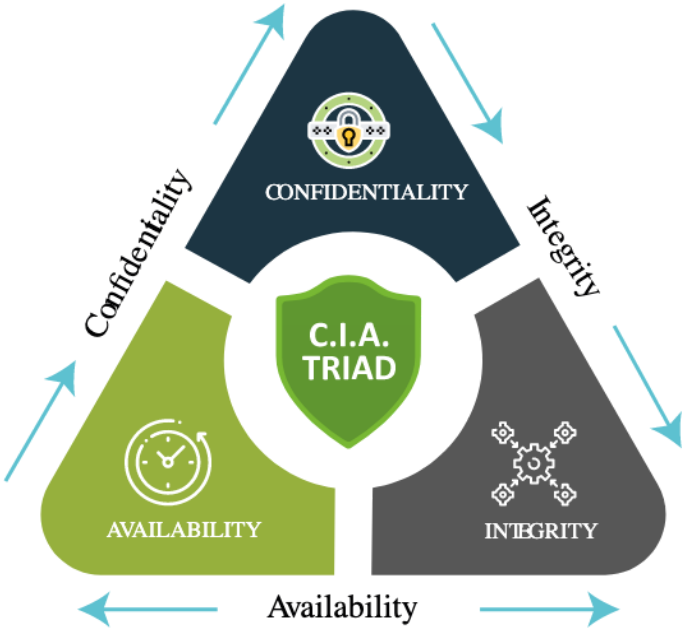
Наверное, у вас сложилось впечатление, что потенциальные проблемы с безопасностью могут возникнуть в самых неожиданных местах, и это касается не только взаимодействия с пользователями в момент HTTP-запроса, но и в целом поддержания надёжности системы. Оборону нужно держать по всем фронтам, ведь даже малейшая уязвимость может стоить очень многого.

Отдельного внимания заслуживает топ-10 проблем безопасности вообще, а не только в вебе. Заметим, что в некоторых моментах этот список пересекается с предыдущим. На картинке мы видим, как меняли свои места в рейтинге разные уязвимости. Например, проблема использования устаревших библиотек с известными уязвимостями поднялась на пятое место.



[Ссылка на этот рейтинг]

Как видим, проблем много, везде поджидают уязвимости, которыми могут воспользоваться злоумышленники. И как некоторый нарратив о том, какой должна быть безопасная система, в информационной безопасности и в вебе в частности появилась модель CIA MODEL. Это триада компонентов, которыми должна обладать безопасная система.



1. **Целостность** (англ. integrity). В мире информационной безопасности целостность относится к точности и полноте данных. Меры безопасности, ориентированные на целостность, предназначены для предотвращения изменения или неправомерного использования этих данных кем-то неавторизованным. Целостность предполагает поддержание согласованности и достоверности данных на протяжении всего их жизненного цикла. В хранилище, при транзите и в момент обработки.

   Например, сюда мы можем отнести различные методы шифрования или системы версионирования.

2. **Конфиденциальность** (англ. confidentiality). Когда мы говорим о конфиденциальности информации, мы говорим о защите информации от раскрытия посторонним лицам из-за утечки данных. Такие случаи могут быть

даже юридически наказуемыми, не говоря уже о широко известных репутационных издержках. Конфиденциальность охватывает спектр средств управления доступом и мер, которые защищают вашу информацию от неправомерного использования любым несанкционированным образом.

3. **Доступность** (англ. availability). Мы удостоверимся в том, что система доступна авторизованным пользователям. На практике это понятие значит, что наша система всегда или почти всегда работает корректно, она может нормально функционировать при программных сбоях, при отказе оборудования и человеческих ошибках. Например, при отказе сети пользователи не смогут получить доступ к данным. Разные политики безопасности подразумевают тут резервные копии данных и отдельных компонентов системы, чтобы обеспечить непрерывную работу всей системы целиком.

Кажется, у нас получилось нащупать айсберг информационной безопасности — мы понимаем, что это огромная и очень интересная тема. А в следующих темах попытаемся разобраться в некоторых частных проблемах и в том, как их можно решать.