Оглавление

Содержание

Вступительное слово от сообщества

ASP.NET Core – это профессиональный и гибкий инструмент для создания современных веб-приложений, и безопасность – это ключевой аспект, особенно когда речь идет о работе с конфиденциальной информацией. Сам по себе фреймворк ASP.NET Core безопасен благодаря разработкам и стандартам безопасности Microsoft. Однако, несмотря на старания Microsoft, разработчику требуется понимать основы кибербезопасности и уметь выстраивать механизмы защиты.

Создание безопасных приложений – это сложный и ответственный процесс, требующий глубоких знаний и опыта в области информационной безопасности. Автор рассказывает о лучших практиках и технологиях работы с секретными данными, векторах атак, защите веб-приложений как с точки зрения создания безопасного кода, так и с точки зрения защиты от попыток получения контроля извне.

Рекомендуем не только прочитать книгу, но и попробовать все практические примеры. Это эффективный способ узнать о том, как именно эксплуатируются наиболее популярные ошибки и уязвимости. Обязательно попробуйте разными способами «взломать» ненадежный магазин соков от OWASP, это увлекательный процесс.

Мы надеемся, что эта книга станет вашим надежным помощником в создании безопасных приложений на платформе .NET и поможет вам защитить свои приложения от различных угроз. Приятного чтения!

Над переводом работали представители сообщества DotNet.Ru:

- Игорь Лабутин;
- Дмитрий Жабин;
- Илья Лазарев;
- Рустам Сафин;
- Радмир Тагиров;
- Сергей Бензенко;
- Андрей Беленцов;
- Андрей Брижак;
- Алексей Ростов;
- Ринат Сафиуллин;
- Анатолий Кулаков.

Введение

Я до сих пор помню, как впервые столкнулся с темой безопасности веб-приложений, хотя в то время и не осознавал ее важности. Где-то в 1997 г., когда я занимался созданием веб-приложений (точнее, сайтов), услуги хостинга были очень дорогими. В случае с одним из моих проектов единственным вариантом, который я мог себе позволить, была возможность создать только одну страницу (!), и для этого мне приходилось использовать инструменты хостинг-провайдера – никакого собственного HTML или CSS. У меня было много свободного места на бесплатном хостинге, но я не мог использовать там собственный домен: у меня было что-то вроде http://home.someprovider.com/mysite.

Одной из очень немногих доступных мне функциональных возможностей была настройка ключевых слов страницы (раньше поисковые системы анализировали эту информацию). Если бы я использовал, например, словосочетание *web application security, hacking*, оно было бы преобразовано в следующую HTML-разметку:

```
<meta name="keywords" content="web application security, hacking">
```

Поэкспериментировав, я обнаружил, что могу попробовать такое «ключевое слово»:

```
"><meta http-equiv="refresh" content="0;
     url=http://home.someprovider.com/mysite"><"
```

Оказалось, что провайдер помещал эти данные в тег `<meta>` как есть, что привело к следующему результату (для удобочитаемости код отформатирован, а ввод выделен жирным шрифтом):

```
<meta name="keywords" content="">
<meta http-equiv="refresh" content="0;
     url=http://home.someprovider.com/mysite">
<"">
```

Таким образом я внедрил еще один тег `<meta>`, который перенаправлял посетителя на мой настоящий сайт, размещенный на бесплатном хостинге в другом месте.

Потребовалось время, прежде чем я понял значение того, что обнаружил, – на эту страницу можно было внедрить произвольный контент. Моя «атака» была безобидной, но можно было бы добавить и другую, более вредоносную разметку. Это пробудило во мне интерес к изучению безопасности веб-приложений, и с тех пор я не оглядывался назад. Я проверил бесчисленное количество приложений, работал с клиентами до и после аудита, учил разработчиков писать безопасные веб-приложения, выступал на трех континентах на конференциях, посвященных безопасности

веб-приложений, и изо всех сил старался сделать приложения, за которые я отвечал, максимально безопасными. В 2004 г. я впервые получил звание Microsoft MVP по ASP.NET. Я очень внимательно следил за API безопасности, подводными камнями и проблемами в этом фреймворке на протяжении многих лет.

Я думал написать книгу об опыте и знаниях, которые приобрел за последние 25 лет, но не было подходящего времени. В середине 2021 г. оно, наконец, настало, и я отправился в многомесячное путешествие, чтобы изложить все, что я знаю и считаю важным, в книге, которую вы собираетесь прочитать.

По моему опыту, недостаточно просто знать контрмеры, применяемые против определенных угроз. Разработчики должны понимать, как работают атаки, поскольку легче защититься от вещей, с которыми вы уже сталкивались. Вот почему во многих главах сначала будет продемонстрирована сама атака, а затем я объясню, как ее предотвратить. Помимо того что это облегчает восприятие материала, это еще и весело: мы видим, как что-то можно сломать, и называем это работой!

Как следует из названия, центральная тема книги – ASP.NET Core, куда входят Razor Pages и ASP.NET Core MVC. Также там, где это возможно, в ней рассматривается Blazor – еще один фреймворк для создания веб-приложений от компании Microsoft. Во всех примерах используется язык C#, а в качестве основы для них была выбрана платформа .NET 6 (в надежде, что они по-прежнему будут работать и со многими последующими версиями).

Благодарности

Многие, кто принимал участие в подготовке этой книги, упомянуты на обороте титульного листа (и это правильно!), и кроме них есть еще большое количество людей, которые помогли мне и внесли свой вклад.

Я в долгу как перед рецензентами, которые предоставляли полезные комментарии на различных этапах подготовки книги, так и перед читателями издания, которое было доступно по программе Manning Early Access Program (MEAP): Аль Пезевски, Билли Мигель Ванегас, Даниэль Васкес, Даррен Гиллис, Дэвид Пакку, Деннис Хейс Джордже, Дмитрий Сергеевич Дорогой, Дойл Тернер, Эммануил Шардалас, Гай Лэнгстон, Гарри Полдер, Джедиджа Буржуа, Джо Куэвас, Хосе Луис Перес , Марчин Сенк, Марек Петак, Маркус Вольф, Мэттью Харвелл, Майкл Холмс, Милош Тодорович, Ник МакГиннесс, Ник Римингтон, Онофри Джордж, Пол Браун, Ричард Вон, Рон Лиз, Сэмюэл Бош, Стэнли Анози, Сумит К. Сингх, Том Гет, Вьорель-Мариан Муазе и Уэйн Мэтер – спасибо за ваш вклад и помощь в улучшении книги.

Некоторые из моих близких друзей и коллег также оставили неоценимые отзывы и сделали книгу намного лучше. Спасибо всем вам за ваши идеи и поддержку!

Особая благодарность редактору по развитию Дагу Раддеру, который не только занимался поддержкой этого проекта, но и ловил меня каждый раз, когда я пытался «схалтурить».

Об этой книге

Название книги говорит само за себя: в ней рассказывается о безопасности приложений, созданных с использованием ASP.NET Core, поэтому здесь подробно описываются различные угрозы и риски для веб-приложений, основанных на технологии .NET. Я верю в принцип «показывай, а не рассказывай», поэтому вы увидите не только API и меры противодействия, но и то, как происходит атака. Основой для многих глав служат реальные инциденты.

Кому адресована эта книга?

Вы должны понимать основы .NET и владеть хотя бы одним из фреймворков ASP.NET Core (Razor Pages или MVC/Web API). Еще лучше, если вы хорошо разбираетесь в HTML и CSS («я понимаю это, когда вижу»). По крайней мере, в некоторых главах будет полезен поверхностный опыт работы с JavaScript. В качестве предпочтительного языка здесь используется C#, поэтому это еще одно необходимое условие, чтобы получить максимальную отдачу от прочтения книги.

Структура книги

Книга разбита на пять частей, которые содержат 16 глав. В первой части закладывается основа для последующих тем:

- в главе 1 обсуждается, почему важна безопасность веб-приложений и какие существуют параметры ASP.NET Core, а также то, как они могут быть затронуты. Вы получите краткую информацию о настройках проекта, которые предоставляет ASP.NET Core.

Во второй части показаны наиболее распространенные атаки на веб-приложения и способы защиты от них:

- в главе 2 рассматривается межсайтовый скриптинг (XSS) – очень распространенная атака, обычно основанная на внедрении вредоносного кода JavaScript. Пример с внедрением HTML, описанный во введении, также попадает в эту категорию;
- в главе 3 обсуждаются способы атаки на управление сессиями и как сделать сессии более безопасными. Сюда входят функциональные возможности, предоставляемые современными веб-браузерами;
- в главе 4 рассматривается межсайтовая подделка запросов (CSRF) – очень опасная атака, которую можно нейтрализовать

с помощью встроенных функций ASP.NET Core и механизмов защиты, которые есть в последних версиях браузеров;
- в главе 5 описываются потенциальные последствия использования данных, не прошедших валидацию, и как в этом может помочь ASP.NET Core. Сюда входит удобное и мощное средство: валидация модели;
- глава 6 посвящена внедрению SQL-кода, очень старой атаке, редко встречающейся в мире ASP.NET Core благодаря простым в использовании мерам противодействия и появлению инструментов объектно-реляционного отображения, в частности Entity Framework Core.

Третья часть посвящена безопасному хранению данных:

- в главе 7 рассказывается о том, как хранить секретные данные, в частности токены. Один из вариантов – использовать шифрование; другой – использовать предложения облачных провайдеров;
- в главе 8 обсуждается работа с паролями и способы их безопасного хранения. На самом деле пароли вообще не стоит хранить, в отличие от их хешей.

В четвертой части рассматриваются различные параметры конфигурации, связанные с безопасностью:

- в главе 9 подробно описаны HTTP-заголовки, поддерживаемые современными веб-браузерами, которые добавляют в приложение дополнительный уровень безопасности, а также рассказывается, как предотвратить отправку заголовков, отображающих скрытую информацию, клиенту;
- глава 10 знакомит вас с обработкой ошибок в приложениях ASP.NET Core с использованием передовых практик;
- в главе 11 рассматриваются две темы, которые отличаются друг от друга, но тем не менее в какой-то степени они связаны: журналирование может обеспечить сохранение диагностической информации о сайте для последующего ее извлечения, а проверка работоспособности обеспечивает механизм наблюдения за доступностью сайта и его сервисов.

Пятая часть посвящена проверке подлинности и авторизации в приложениях ASP.NET Core:

- в главе 12 вы познакомитесь с системой ASP.NET Core Identity, благодаря которой управлять пользователями и выполнять аутентификацию на сайте становится проще;
- в главе 13 описывается защита API и одностраничных приложений с помощью решения на основе токенов, а также рассматриваются протоколы OAuth и OpenID Connect с точки зрения ASP.NET Core.

Шестая часть охватывает ряд аспектов, являющихся частью процесса обеспечения безопасности:

- в главе 14 обсуждается, как обеспечить безопасность зависимостей, включая различные инструменты аудита;
- в главе 15 основное внимание уделяется инструментам аудита, которые могут помочь обнаружить уязвимости в веб-приложениях;
- в главе 16 рассматривается OWASP Top 10, регулярно обновляемый список десяти главных уязвимостей в веб-приложениях и то, как они описаны в этой книге.

Большинство глав не зависят друг от друга, но при необходимости приводятся соответствующие ссылки.

О коде

В этой книге содержится большое количество примеров исходного кода. Часть из них представляет собой пронумерованные листинги, а часть встречается в самом тексте. В обоих случаях исходный код отформатирован соответствующим шрифтом фиксированной ширины, чтобы отделить его от обычного текста. В некоторых случаях исходный код был переформатирован. Я добавил разделители строк и переделал отступы, чтобы уместить их по ширине книжных страниц. В редких случаях, когда этого оказалось недостаточно, в листинги были добавлены символы продолжения строки (➥). Также из многих листингов, описываемых в тексте, были убраны комментарии. Некоторые листинги сопровождают аннотации, выделяющие важные понятия.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге, – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если

вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Manning Publications очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Об авторе

Кристиан Венц – пионер Интернета, специалист по информационным технологиям и предприниматель. С 1999 г. он написал около 150 книг, посвященных веб-технологиям и смежным темам, которые были переведены на десять языков. На своей основной работе он занимается консультацией предприятий по вопросам цифровизации и Индустрии 4.0, или Четвертой промышленной революции. Будучи неизменным участником международных конференций для разработчиков, он выступал на трех континентах. С 2004 г. Кристиан носит звание MVP для ASP.NET, является ведущим автором официальной сертификации по PHP и время от времени участвует в проектах по разработке программного обеспечения с открытым исходным кодом. У него высшее образование в области компьютерных наук и бизнес-информатики, и он двукратный обладатель премии Кнута.

Об иллюстрации на обложке

Изображение на обложке книги – «Венецианка», взято из сборника Жака Грассе де Сен-Совера, опубликованного в 1797 г. Каждая иллюстрация тщательно прорисована и раскрашена вручную. В те дни было легко определить, где жили люди, какова их профессия или положение в обществе, просто по одежде. Мы в Manning отмечаем изобретательность и инициативу компьютерного бизнеса с помощью книжных обложек, основанных на разнообразии жизни в разных регионах несколько веков назад, которое оживает благодаря изображениям из этой коллекции.

Часть I

Первые шаги

Не проходит и недели без каких-либо громких инцидентов в области информационной безопасности: будь то утечка данных в открытый доступ, популярные библиотеки кода, получающие обновления с вредоносным ПО, распространение новых программ-вымогателей и сайты, подверженные уязвимостям. Многие из событий, о которых вы читаете в IT-новостях, стали возможными благодаря ошибкам в коде. Поскольку центральная тема этой книги – ASP.NET Core, в первой главе будут представлены возможности веб-приложений, предоставляемые этой технологией, и анализ того, где могут произойти атаки. Мы построим «ментальную модель» для понимания остальной книги.

1 О безопасности веб-приложений

В этой главе:

- почему важна безопасность веб-приложений;
- использование ASP.NET Core для создания веб-приложений и API;
- почему определенные части приложения находятся под угрозой;
- чего ожидать от этой книги.

Девять из десяти веб-приложений имеют уязвимости. Таков довольно пугающий вывод исследования, опубликованного в 2020 г. компанией Positive Technologies (http://mng.bz/mOj2), поставщиком различных решений в области безопасности. Очевидно, что те, кто проводит такие исследования, часто могут быть предвзяты, но ряд других исследований, проведенных в предыдущие годы, дал аналогичные результаты. Вот отчет по одному из исследований 2009 г.: http://mng.bz/5Qo1.

Авторы исследования также обнаружили, что примерно четыре из пяти уязвимостей веб-приложений находятся в коде, а не, скажем, в конфигурации сервера. Основываясь на этом, можно выделить две тенденции:

- основная угроза безопасности веб-приложений заключается в коде;

- проблема носит масштабный характер, и ситуация, похоже, не улучшается.

Часто недостатки в системе безопасности проявляются не сразу, а только когда станет слишком поздно и веб-приложение будет успешно взломано. Поэтому необходимо сделать безопасность веб-приложений главным приоритетом и использовать передовые методы для ее обеспечения с самого начала проекта.

Большинство угроз связаны с работой веб-браузеров, протокола HTTP, серверов баз данных и другими «сетевыми аспектами». Следовательно, эти риски не зависят от технологий. Вот один из примеров: теоретически внедрение кода JavaScript на сайт работает независимо от используемого языка сервера или фреймворка. На практике существуют следующие отличия:

1. некоторые языки и фреймворки обладают встроенными средствами противодействия, которые помогают предотвратить распространенные атаки без каких-либо дополнительных усилий на этапе разработки;
2. в разных технологиях и фреймворках функции, методы и API, используемые для защиты от определенных атак и рисков, естественно, называются по-разному.

Таким образом, книга, посвященная безопасности веб-приложений, должна представлять и описывать распространенные атаки в более или менее общем виде, после чего необходимо будет ввести меры противодействия, привязанные к определенной технологии. Стек, который мы будем использовать в этой книге, – это .NET; а поскольку мы говорим о веб-приложениях, то в центре внимания будет фреймворк ASP.NET Core. Книга была написана с использованием .NET 6 и ASP.NET Core 6, но надеюсь, что она будет полезна и при работе с более новыми версиями.

1.1 ASP.NET Core: история и фреймворки

У ASP.NET долгая история, связанная с историей платформы .NET, которая впервые была выпущена в виде бета-версии в 2001 г. и в качестве окончательной версии 1.0 в начале 2002 г. Тогда программный пакет назывался «.NET Framework» и содержал фреймворк для разработки серверных веб-приложений под названием ASP.NET (первые три буквы были взяты из названия предыдущей веб-технологии Microsoft ASP. Это сокращение от «Active Server Pages»). Вместе с .NET Framework появился новый язык программирования C#, который будет использоваться в этой книге, хотя существуют и другие варианты (Visual Basic for .NET, или функциональный язык F#).

1.1.1 История версий ASP.NET Core

ASP.NET и .NET Framework развивались на протяжении многих лет, но мы намеренно не рассматриваем их в этой книге. Возможно, это неожиданно, особенно учитывая название книги, но в 2010-х гг. компания Microsoft работала над новой версией .NET, кульминацией которой стал выпуск .NET Core 1.0 в середине 2016 г. Это была новая версия .NET с открытым исходным кодом, более или менее независимая от платформы, и она больше не была привязана к Windows. Слово *Core* использовалось, чтобы избежать путаницы с .NET Framework, особенно что касалось номеров версий. Сработало это или нет – другой вопрос, но, чтобы еще больше все запутать, Microsoft отказалась от этого слова в названии, когда появилась версия .NET 5.0. Причина: последняя и, вероятно, окончательная версия .NET Framework и ASP.NET – 4.8, поэтому .NET Framework 5 не будет; таким образом, «.NET 5» явно означает новую версию .NET.

Однако с ASP.NET все немного сложнее. У фреймворка ASP.NET MVC есть свои номера версий. Последний выпуск пакета NuGet ASP.NET MVC для .NET Framework – 5.2.8 (http://mng.bz/2nE0), поэтому на самом деле «ASP.NET 5» может означать три вещи:

- ASP.NET MVC 5 (на основе .NET Framework);
- ASP.NET Core 5 (на основе .NET 5, ранее известного как .NET Core);
- ASP.NET как часть .NET 5. Это предыдущее название проекта, который позже стал .NET Core 1.0.

Я думаю, можно согласиться с тем, что имело смысл оставить слово *Core*, чтобы сделать название продукта явным, поэтому на данный момент это *ASP.NET Core*. Не нужно быть пророком, чтобы предсказать, что в какой-то момент в будущем от него, скорее всего, откажутся. Но пока, если в названии есть это слово, мы говорим о текущей версии веб-фреймворка Microsoft, а не об устаревшей. В этой книге используется .NET 6, где слово *Core* все еще присутствует.

1.1.2 MVC

Архитектурный паттерн «модель–представление–контроллер» (MVC) был придуман в 70-х гг. прошлого века. Он появился в приложениях с графическим интерфейсом, но стал очень популярным при разработке веб-приложений. Написание HTML и CSS, отвечающих за внешний вид страницы, – это совершенно иной навык, нежели реализация серверной части. Поэтому отделение пользовательского интерфейса от логики имеет смысл, и MVC – один из доступных вариантов. Будучи адаптированным под веб-приложения, MVC в основном работает следующим образом (рис. 1.1):

- контроллер принимает пользовательский ввод (в случае с веб-приложением это данные HTTP-запроса);
- контроллер получает модель (часто данные из базы данных) и манипулирует ею, а затем передает эту модель представлению (обычно HTML-странице);
- клиент получает представление и может использовать его для создания нового запроса.

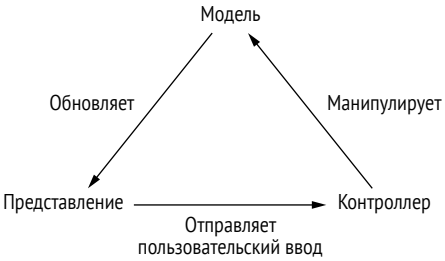

Рис. 1.1 Как работает паттерн MVC

В ASP.NET MVC эти компоненты обычно представлены следующим образом (поскольку ASP.NET MVC обладает широкими возможностями настройки, многие детали могут быть изменены, но мы опишем поведение по умолчанию):

- контроллер – это класс C#. Запросы сопоставляются с «методами действия». По сути, это общедоступные методы C#;
- модель обычно представляет собой объект или класс C#, часто заполненный содержимым базы данных (но не обязательно в соотношении 1:1). В примерах Microsoft обычно используют Entity Framework Core, инструмент объектно-реляционного отображения, что, конечно же, не обязательно. Контроллер получает доступ к модели и может манипулировать ею, а затем при необходимости передает ее представлению;
- представление – это, по сути, HTML-страница с дополнительной разметкой для привязки значений из модели или выполнения кода. Поскольку мы используем язык C#, эти страницы имеют расширение *cshtml*. Движок представлений Razor позволяет включать код C# в эти файлы с помощью специального символа @. Файлы компилируются, давая возможность выполнить C# код; браузер, разумеется, получает итоговый HTML-код.

При создании нового проекта в Visual Studio выбранный шаблон задает технологический стандарт для приложения. На рис. 1.2 показан ряд доступных шаблонов проектов. Обратите внимание, что четвертый вариант, ASP.NET Core Web App (Model-View-Controller), также включает веб-API, так как они очень похожи с точки зрения кода.