

Pro ASP.NET Web API Security

Securing ASP.NET Web API



Badrinarayanan Lakshmiraghavan

Apress®

Pro ASP.NET Web API Security

Copyright © 2013 by Badrinarayanan Lakshmiraghavan

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

ISBN 978-1-4302-5782-0

ISBN 978-1-4302-5783-7 (eBook)

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

President and Publisher: Paul Manning

Lead Editor: Ewan Buckingham

Developmental Editor: Barbara McGuire

Technical Reviewer: Fabio Claudio Ferracchiati

Editorial Board: Steve Anglin, Mark Beckner, Ewan Buckingham, Gary Cornell, Louise Corrigan, Morgan Ertel,

Jonathan Gennick, Jonathan Hassell, Robert Hutchinson, Michelle Lowman, James Markham,

Matthew Moodie, Jeff Olson, Jeffrey Pepper, Douglas Pundick, Ben Renow-Clarke, Dominic Shakeshaft,

Gwenan Spearing, Matt Wade, Tom Welsh

Coordinating Editor: Mark Powers

Copy Editor: Teresa Horton

Compositor: SPi Global

Indexer: SPi Global

Artist: SPi Global

Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a Delaware corporation.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales-eBook Licensing web page at www.apress.com/bulk-sales.

Any source code or other supplementary materials referenced by the author in this text is available to readers at www.apress.com/9781430257820. For detailed information about how to locate your book's source code, go to www.apress.com/source-code/.

To Him, who is able to be both larger than the largest and smaller than the smallest.

To my mother and father.

Contents at a Glance

Foreword	xv
About the Author	xvii
About the Technical Reviewer	xix
Acknowledgments	xxi
Introduction	xxiii
■ Chapter 1: Welcome to ASP.NET Web API	1
■ Chapter 2: Building RESTful Services	13
■ Chapter 3: Extensibility Points	29
■ Chapter 4: HTTP Anatomy and Security.....	41
■ Chapter 5: Identity Management.....	81
■ Chapter 6: Encryption and Signing	103
■ Chapter 7: Custom STS through WIF.....	119
■ Chapter 8: Knowledge Factors	133
■ Chapter 9: Ownership Factors	163
■ Chapter 10: Web Tokens	191
■ Chapter 11: OAuth 2.0 Using Live Connect API.....	227
■ Chapter 12: OAuth 2.0 from the Ground Up	251
■ Chapter 13: OAuth 2.0 Using DotNetOpenAuth	283

■ **Chapter 14: Two-Factor Authentication.....319**

■ **Chapter 15: Security Vulnerabilities.....345**

■ **Appendix: ASP.NET Web API Security Distilled375**

Index.....381

Contents

Foreword	xv
About the Author	xvii
About the Technical Reviewer	xix
Acknowledgments	xxi
Introduction	xxiii
■ Chapter 1: Welcome to ASP.NET Web API	1
What Is a Web API, Anyway?.....	1
A Primer on RESTful Web API	3
Hello, ASP.NET Web API!.....	4
WCF vs. ASP.NET Web API.....	4
Programming Model Differences.....	5
Scenarios in Which ASP.NET Web API Shines	6
A Primer on Security	8
Summary.....	11
■ Chapter 2: Building RESTful Services	13
What Is a RESTful Service?	13
Identification of Resources.....	14
Manipulation of Resources Through Representations.....	15
Self-Descriptive Messages.....	16
Scenario 1: JSON Representation.....	17
Scenario 2: No Content Type.....	17
Scenario 3: XML Representation.....	17
Scenario 4: Mix and Match	18

Hypermedia as the Engine of Application State	18
Implementing and Consuming an ASP.NET Web API.....	19
Our First Attempt in Securing a Web API	23
Summary.....	28
■ Chapter 3: Extensibility Points	29
The What and Why of Extensibility Points	29
ASP.NET Web API Life Cycle.....	30
Filters	32
Authorize Filter	32
Subclassed Authorize Filter	33
ActionFilter	34
Message Handlers.....	34
HTTP Modules	38
Summary.....	40
■ Chapter 4: HTTP Anatomy and Security.....	41
HTTP Transaction.....	41
HTTP Request.....	42
Request Headers	43
HTTP Methods	43
Method Overriding	44
HTTP Response	45
Status Codes	46
The Curious Case of an Unhandled Exception	47
Response Headers	48
Response Body.....	49
Web Caching	50
Entity Tag	53
Implementing ETag in ASP.NET Web API	53
Testing ETag ActionFilter.....	55

ETags for Managing Concurrency	57
Cross-Origin Resource Sharing	59
Simple CORS.....	59
Preflighted Request	63
HTTP Cookies	66
Cookies and ASP.NET Web API	67
Proxy Server	70
HTTPS.....	71
Configuring HTTPS for ASP.NET Web API Hosted in IIS	73
Fiddler: A Tool for Web Debugging.....	74
Capturing and Decrypting HTTPS Traffic.....	75
Fiddler as Man-in-the-Middle.....	77
Summary.....	79
■ Chapter 5: Identity Management	81
Authentication and Authorization	81
Role-Based Security	82
Identity and Principal.....	82
Using Generic Identity in a WinForms Application	83
Using Windows Identity in a Console Application	85
The Curious Case of Thread.CurrentPrincipal.....	87
Claims-Based Security	88
Real-World Analogy	89
Claims-Based Access Control vs. Role-Based Access Control	90
Using Claims-Based Security	90
Implementing Role-Based Access Control Using Claims	91
Implementing Claims-Based Access Control Using Claims	92
Implementing Claims-Based ASP.NET Web API	94
Security Token	98
Token Formats	99
Summary.....	101

■ Chapter 6: Encryption and Signing	103
Cryptography	103
Encrypting a Message Using Symmetric Keys	105
Signing a Message Using Symmetric Keys	107
Encrypting a Message Using Asymmetric Keys	110
Signing a Message Using Asymmetric Keys	114
Token Encryption and Signing	115
Comparison to Cryptographic Handling in WIF	115
Summary	117
■ Chapter 7: Custom STS through WIF	119
WS-Trust	119
Trust Brokering	120
The Request–Response Pair of RST and RSTR	120
Proof of Possession	122
Building a Custom STS	125
Requesting a Token from a Custom STS	130
Summary	132
■ Chapter 8: Knowledge Factors	133
Basic Authentication	133
Implementing Basic Authentication in ASP.NET Web API	134
Testing Basic Authentication	138
Merits and Demerits of Basic Authentication	139
Digest Authentication	140
The Nuts and Bolts	140
Implementing Digest Authentication	144
Testing Digest Authentication	151
Merits and Demerits of Digest Authentication	152
Trying to Break Digest Authentication	152

Windows Authentication.....	156
Configuring Windows Authentication.....	156
Windows Authentication in Action	157
Impersonation.....	159
Testing Windows Authentication.....	160
Merits and Demerits of Windows Authentication.....	161
Summary.....	162
■ Chapter 9: Ownership Factors	163
Preshared Key	163
Designing a Preshared Key Security Mechanism	164
Implementing the Preshared Key Design	167
Merits and Demerits of a Preshared Key	170
X.509 Client Certificate.....	170
Server Certificate vs. Client Certificate.....	171
Using Client Certificate for Authentication in ASP.NET Web API	172
Merits and Demerits of a Client Certificate Mechanism	180
SAML Tokens	181
Implementing the Client Console Application	182
Accepting a SAML Token in ASP.NET Web API	184
Active Directory Federation Services	187
Merits and Demerits of SAML Tokens.....	188
Summary.....	189
■ Chapter 10: Web Tokens	191
Simple Web Token	191
Anatomy of a SWT	192
Using a SWT in a Console Application	193
JSON Web Token	200
Base64 URL Encoding.....	201
Anatomy of a Signed JSON Web Token.....	202

Using a Signed JSON Web Token in a Console Application.....	203
Anatomy of an Encrypted JSON Web Token.....	212
Using an Encrypted JSON Web Token in a Console Application.....	214
JWT Handler	223
Summary.....	225
■ Chapter 11: OAuth 2.0 Using Live Connect API.....	227
Use Case for OAuth: App-to-App Data Sharing.....	227
OAuth 2.0 Roles.....	228
OAuth 2.0 Client Types.....	229
OAuth 2.0 Client Profiles.....	229
OAuth 2.0 Authorization Grant Types	230
Authorization Code Grant.....	230
Implicit Grant	231
Resource Owner Password Grant.....	231
Client Credentials Grant.....	231
Access Token.....	232
Access Token as a Bearer Token	232
Refresh Token.....	233
Using Live Connect APIs	234
Registering Your Application in the Live Connect Portal.....	234
Using an Implicit Grant to Access Live Connect	235
Using an Authorization Code Grant to Access Live Connect	239
Using a Resource Owner Password Grant.....	247
Using a Client Credentials Grant.....	249
Summary.....	249
■ Chapter 12: OAuth 2.0 from the Ground Up	251
Scenario: Sharing Contact Information	251
Design	253
MyContacts Project.....	253
MyPromo Project	254

HTTP Transactions	255
Building the Contacts Manager Application	258
Building the Promotion Manager Application	263
Building the Authorization Server.....	267
Index Action Method for HTTP GET	269
Authenticate Action Method	273
Index Action Method for HTTP POST	274
Building the Resource Server.....	277
Security Considerations	280
Summary.....	282
■ Chapter 13: OAuth 2.0 Using DotNetOpenAuth	283
Design	284
MyContacts Project.....	284
MyPromo Project	285
HTTP Transactions	286
Implementation Ground Work.....	290
Building the Client Application	291
Building the Authorization Server.....	294
Creating the Infrastructure	295
Creating the IAuthorizationServerHost Implementation	301
Creating OAuth20Controller	306
Securing the OAuth20Controller Endpoints	309
Building the Resource Server.....	311
Implicit Grant.....	313
Summary.....	318
■ Chapter 14: Two-Factor Authentication.....	319
Two Ways to Implement TFA.....	319
Implementing Blanket TFA with ASP.NET Web API.....	320

Google Authenticator	322
How Does Google Authenticator Work from a User Perspective?	323
Under the Hood of Google Authenticator	325
Base32 Encoding and Decoding	327
Implementing TOTP Algorithm in a Console App	330
Implementing Constant Per-Request TFA.....	333
Implementing On-Demand Per-Request TFA.....	337
Two-Factor Security through Mobile Phones	341
Summary.....	342
■ Chapter 15: Security Vulnerabilities.....	345
OWASP Application Security Risks	345
Injection	346
Broken Authentication and Session Management.....	350
Cross-Site Scripting (XSS)	350
Insecure Direct Object References	351
Security Misconfiguration.....	351
Sensitive Data Exposure	352
Missing Function Level Access Control	358
Cross-Site Request Forgery (CSRF)	359
Using Known Vulnerable Components.....	367
Unvalidated Redirects and Forwards.....	367
Security = Hardware + Software + Process	367
Web Server Fingerprinting.....	368
Logging, Auditing, and Tracing.....	370
Implementing Tracing in ASP.NET Web API	370
Input Validation.....	372
Summary.....	373
■ Appendix: ASP.NET Web API Security Distilled	375
Index.....	381

Foreword

Everybody who knows me also knows that identity and access control in distributed applications are very near and dear to my heart. Having spent many years in the WS* security space (or WS-Deathstar as many called it), I was happy to see that Microsoft finally built a web service framework that really embraces HTTP instead of abstracting it away.

It is also fair to say that the “web API idea” has taken the world (and its developers) by storm. Even if the technology is not really new, having such capabilities in a mainstream framework like .NET makes adoption really easy. In the short period of time since its first release, it has gained a lot of traction.

As with many other Microsoft technologies, for the first version they mainly concentrated on the core framework, extensibility points, and a limited set of common use cases. The same is true for ASP.NET Web API: Although all the foundational work has been done, the main focus in the security space was Windows authentication and (simpler) AJAX scenarios. There was no built-in support for cross-domain scenarios like basic authentication, client certificates, and token-based authentication (SAML/JWT), let alone two-factor authentication or emerging standards like OAuth2, although it was technically totally possible.

Luckily Badri took that challenge and spent a lot of time exploring all these technologies and their integration into ASP.NET Web API for you. I was totally impressed with how complete and strong this book is on both the “broad” axis and the “deep” axis. In many ways this is the book that I wanted to write for years but never found the time for it. Excellent job!

You, the reader, have quite a journey ahead of you. The world is moving to the web API approach to model services, and the security scenarios are becoming even more complex. OAuth2 is the protocol that enables many of these new architectures, but it will make your head hurt at first. It is also really hard to write a good security system that does not get in the way of legitimate users. If you do your job really well no one will notice it, and for everything else they will blame you! But a working system is very rewarding, and I still very much enjoy doing security every single day.

With that said (and because I am not a big fan of overly long forewords), I wish you a lot of fun and many “a-ha” moments while reading this really comprehensive and interesting book! Mind those tokens!

Dominick Baier

<http://leastprivilege.com>

<http://thinktecture.com>

<https://twitter.com/leastprivilege>

About the Author



Badrinarayanan Lakshmiraghavan has more than fourteen years of information technology experience in all phases of the software development life cycle, including technology consulting and advisory roles in multiple technologies. He has been programming on the Microsoft technology stack from the days of Visual Basic 3.0.

Badri currently is a senior technology architect with Global Technology Consulting - Microsoft Center of Excellence of Cognizant (NASDAQ: CTSH), a Fortune 500 company. He speaks three languages: Tamil, English, and C#.

Badri's coordinates are 12.9758° N, 80.2205° E on the third rock from the yellow-dwarf star that lies close to the inner rim of the Orion arm of the Milky Way Galaxy.

About the Technical Reviewer

Fabio Claudio Ferracchiati, a prolific writer on cutting-edge technologies, has contributed to more than a dozen books on .NET, C#, Visual Basic, and ASP.NET. He is a .NET Microsoft Certified Solution Developer and lives in Milan, Italy. You can read his blog at Ferracchiati.com.

Acknowledgments

Whether you seek general information on .NET security or specific information on claims-based identity and ASP.NET Web API, you likely will find the answers you need on his blog at <http://leastprivilege.com> or in one of his posts in a technical forum such as MSDN. No points for guessing who it is: Dominick Baier, the ultimate voice of wisdom when it comes to ASP.NET Web API security! I deeply appreciate Dominick for all his help and guidance, including taking time from his busy schedule to write the foreword for this book.

Just about every book author acknowledges the team assembled by the publisher, and I won't be any different. Cliché or not, I must gratefully thank the following individuals who are part of the Apress team (in the same order as they got involved).

- Ewan Buckingham, lead editor, for his patience answering all my relevant and irrelevant questions and helping me all the way from the proposal stage to manuscript completion.
- Mark Powers, coordinating editor, for his helping nature and promptness (I have yet to see an instance where Mark has not replied to my mail two hours from the time I clicked the Send button despite being on the other side of the globe).
- Fabio Claudio Ferracchiati, technical reviewer, for catching the subtle things that I overlooked.
- Teresa Horton, copy editor, for putting up with my writing, notably my problem with the usage of articles.
- The SPi Global production team for diligently incorporating all the changes I asked for.
- Barbara McGuire, developmental editor, for her patience in reading through my jumbles, giving structure and order to the content. Thanks very much, Barbara; you might be last on this list, but definitely not the least!

My thanks also to Arvind TN of Cognizant GTC Microsoft CoE for asking THE question that resulted in this book.

Finally, a huge thank you to my family—my wife Poornima and my sons Anirudh and Aparajith—for their understanding and enormous patience. My special thanks to Anirudh for understanding, without any complaints, that his dad has to sit in front of the computer typing away, unable to watch with him such exciting things as an asteroid hitting the earth and obliterating Triceratops, T-Rex, Stegosaurus, and Alamosaurus.

Introduction

Risk comes from not knowing what you're doing.

—Warren Buffett

Few organizations can afford to have dedicated people working on application security. More often than not, a developer or a lead developer from the team is entrusted with the responsibility for retrofitting security into the application or a service. In this quest, the developer looks around, maybe Googles some information, asks a question or two in forums, and rolls his own security implementation without knowing fully the underlying concepts and the implications of the choices he made. This path of least resistance is usually taken because of the project schedule pressures and the lack of emphasis or the focus that the nonfunctional aspect of security generally deserves.

Not reinventing the wheel is a great policy for application development teams because reusable components like libraries and frameworks help get things done efficiently and the right way, incorporating best practices. The flip side of reusable components, open source or not, is that they result in a “black box” syndrome: Things just work and continue to work until the time they stop working. Also, if a reusable component provides options, a developer must know the different choices available as well as the advantages and disadvantages of those choices to make a knowledgeable decision on the methods to be employed for the security requirements at hand.

Compared to the SOAP-based Windows Communication Foundation (WCF) services that enjoy the support of mature security specifications such as WS-Trust, WS-Security, and so on, REST-based ASP.NET Web API currently has very little support. OAuth 2.0, which is the equivalent for WS-Trust and WS-Security in the REST world, is nascent: The OAuth 2.0 framework and the bearer token specifications were published in October 2012.

Even if you have simple security needs that can be met by the direct authentication pattern of a client presenting a password to your ASP.NET Web API for authentication, will you implement Windows Authentication, which is a popular choice for intranet ASP.NET applications, or Forms Authentication, which is a great choice for Internet ASP.NET applications, or widely supported HTTP-based basic or digest authentication? There are pros and cons with every option, and there is no one-size-fits-all solution available for securing a web API.

This is where this book comes in and presents to you the various options available for securing ASP.NET Web API, along with the merits and demerits of those options. Whether you roll your own security mechanism or use a reusable component in the form of a library or a framework, you will be able to make informed decisions by learning the underpinnings of the mechanisms and the implications of the choices you make.

However, this book does not give you any ready-made, penetration-tested code to copy and paste straight into your production implementation. It does not give you fish, but instead teaches you to catch fish. Using this book, you can gain a solid understanding of the security techniques relevant to ASP.NET Web API. All the underlying concepts are introduced from basic principles and developed to the point where you can use them confidently, knowing what you are doing. If you want to get your hands on proven, production-strength code, there are a couple of excellent open-source resources:

- **Thinkecture.IdentityModel.45** features an extensible authentication framework for ASP.NET Web API supporting SAML 1.1/2.0, JSON Web Token (JWT), Simple Web Token (SWT), access keys, and HTTP basic authentication. It also has support for protected cookies and Cross Origin Resource Sharing (CORS). See <https://github.com/thinkecture/Thinkecture.IdentityModel.45>.

- Thinktecture's **IdentityServer 2**, a lightweight STS built using the .NET Framework 4.5, ASP.NET MVC4, WCF, and web API that supports both WS-Trust and OAuth 2.0. See <https://github.com/thinktecture/Thinktecture.IdentityServer.v2>.

What You'll Learn

- Identity management and cryptography
- HTTP basic and digest authentication and Windows authentication
- HTTP advanced concepts such as web caching, ETag, and CORS
- Ownership factors of API keys, client X.509 certificates, and SAML tokens
- Simple Web Token (SWT) and signed and encrypted JSON Web Token (JWT)
- OAuth 2.0 from the ground up using JWT as the bearer token
- OAuth 2.0 authorization codes and implicit grants using DotNetOpenAuth
- Two-factor authentication using Google Authenticator
- OWASP Top Ten risks for 2013

How This Book Is Organized

Pro ASP.NET Web API Security is divided into fifteen chapters. Although it is not divided into parts, the chapters do tend to fall together into several related groups. The first three chapters constitute one such group that pertains to the core ASP.NET Web API framework. Chapter 4 is a stand-alone chapter on HTTP. Chapters 5, 6, and 7 form a group on .NET security topics of identity management and cryptography. Chapter 8 is a stand-alone chapter on knowledge-factor security, and Chapters 9 and 10 are related to ownership factors. Chapters 11, 12, and 13 form the OAuth 2.0 group. Chapter 14 is a stand-alone chapter on two-factor authentication. Finally, Chapter 15, another stand-alone chapter, focuses on OWASP security risks.

The way the chapters are organized in this book takes into account the dependencies one chapter might have on another. If you are confident, you can feel free to skip chapters, but trying to read the chapter on SWT without understanding the basics of digital signing will likely not be very productive. Similarly, trying to implement implicit grant flow without understanding the implications of same-origin policy and the related CORS will be a challenging experience. For this reason, the best way to derive the maximum benefit from this book is to read the chapters sequentially, starting with Chapter 1 and skimming any text that you are already familiar with.

Chapter 1: Welcome to ASP.NET Web API

We start off with understanding what a web API is in general before moving on to a primer on RESTful web API, followed by a review of how Microsoft's ASP.NET Web API framework can help you build web APIs. We complete the chapter with a primer on security that looks at all aspects of security, above and beyond a login screen accepting a username and password, which for many people is the meaning of the word *security*.

Chapter 2: Building RESTful Services

An HTTP service that handles XML and/or JSON requests and responds to HTTP methods such as GET, POST, PUT, and DELETE is not necessarily a RESTful service. This chapter introduces you to Roy T. Fielding's constraints that must be satisfied for an HTTP service to be called RESTful and builds our first web API, a simple Hello-World kind of API.

Chapter 3: Extensibility Points

The ASP.NET Web API framework has various points of extensibility built into the web API pipeline for us to extend the processing pipeline. This chapter focuses on understanding the web API extensibility points such as filters and message handlers from the point of view of leveraging the same for securing ASP.NET Web API to deal with threats at the earliest available opportunity. It also highlights the trade-offs associated with selecting the web API extensibility point of a message handler over the ASP.NET extensibility point of the HTTP module for authentication and authorization.

Chapter 4: HTTP Anatomy and Security

This chapter introduces you to Hypertext Transfer Protocol (HTTP), the protocol behind the World Wide Web. Understanding HTTP is a prerequisite to understanding the security aspects of ASP.NET Web API. Instead of fighting against it or abstracting it away, web API embraces HTTP. For this reason, understanding HTTP is all the more important: A house is only as strong as its foundation! This chapter also covers some of the advanced concepts of HTTP, things that are a must to create production-grade, performant, secure web APIs such as Web Caching, ETags, Cross-Origin Resource Sharing (CORS), cookies, proxy servers, HTTPS, and the ultimate tool of HTTP debugging, Fiddler.

Chapter 5: Identity Management

Identity management is an important aspect of application security. In this chapter, we focus on how a subject or an entity gets authenticated and how the actions an entity attempts to perform are authorized by an application in the context of the .NET Framework. This chapter introduces you to the interfaces `IIdentity` and `IPrincipal` that form the basis of role-based access control (RBAC) and compares it with the more flexible and granular claims-based access control (CBAC), which is built based on the claims. Readers get to take a first peek at the security tokens and the three major formats: SAML, SWT, and JWT.

Chapter 6: Encryption and Signing

Windows Identity Foundation (WIF) hides away the nuts and bolts of tokens and lets the developers work with a set of claims without bothering about the aspects of cryptography. As we step out of the realm of WCF/WIF, securing RESTful ASP.NET Web APIs without depending on WIF classes for the cryptographic heavy lifting means understanding the nuts and bolts of encryption and signing. This chapter covers encryption and decryption and signing and validation using symmetric keys and asymmetric keys: public-private keys generated using `RSACryptoServiceProvider` as well as a self-signed certificate generated using the `Makecert` tool.

Chapter 7: Custom STS through WIF

One of the key components in the WS-Trust scheme of things is Security Token Service (STS). WIF allows you to build your own custom STS, although it is highly recommended that you buy one instead of building one. This short chapter introduces you to WS-* protocols, specifically WS-Trust, and goes through the steps for creating a custom STS to enhance your understanding of STS and how STS creates and issues tokens.

Chapter 8: Knowledge Factors

A knowledge factor is something a user knows, such as a password or a PIN. This chapter explores the knowledge-factor authentication mechanisms that can be used to secure ASP.NET Web API. Login credentials of a user ID and password combination is probably the most widely used knowledge factor, and this chapter focuses on the mechanisms leveraging this factor: the two authentication schemes defined in HTTP specification, namely basic and digest authentication, and the Windows-OS-powered Integrated Windows Authentication (IWA), more commonly known as Windows Authentication.

Chapter 9: Ownership Factors

An ownership factor is something a user owns or possesses, such as a key, a certificate, or a token. This chapter examines ownership-factor authentication mechanisms for securing ASP.NET Web API, such as preshared keys (PSKs), more commonly called API keys, X.509 client certificates, and SAML tokens.

Chapter 10: Web Tokens

This chapter is an extension of the previous chapter on ownership-factor security, for web tokens are ownership factors just like SAML tokens. However, web tokens deserve a chapter of their own because they are a better fit for RESTful services. Hence, this chapter is dedicated to web tokens and takes an in-depth look at the two most popular web token formats by studying the anatomy of the Simple Web Token (SWT) and the JSON Web Token (JWT), including both signed (JWS) and encrypted (JWE) forms.

Chapter 11: OAuth 2.0 Using Live Connect API

OAuth 2.0 is an open standard for authorization. Roughly speaking, it can be considered the WS-* of the REST world. We start our exploration of OAuth 2.0, mainly from the point of view of a client consuming a web API that implements OAuth 2.0. We review the four types of grants and take a detailed look at implicit and authorization code-based grants using Microsoft Live Connect API.

Chapter 12: OAuth 2.0 from the Ground Up

In this chapter, we move to the other side of the table. Instead of focusing on a client that consumes an API, we now develop a web API implementing OAuth 2.0, specifically the authorization code-based grant. Implementation is performed from scratch using two ASP.NET MVC web applications so you can understand the nuts and bolts.

Chapter 13: OAuth 2.0 Using DotNetOpenAuth

Although it is possible to build on the OAuth 2.0 implementation from the previous chapter and develop your production-strength OAuth 2.0 implementation, this chapter implements the same authorization code-based grant using DotNetOpenAuth (DNOA), which is a well-established open source .NET library that helps you write production-grade OAuth 2.0-based authorization for your web API, in conformance to the principle of not reinventing the wheel.

Chapter 14: Two-Factor Authentication

When you have an authentication mechanism that leverages a combination of two of the knowledge, ownership, and inheritance factors, it is called two-factor authentication (TFA or 2FA). This chapter covers TFA by leveraging the knowledge factor of a password, the ownership factor of an X.509 client certificate, and TFA on a need basis realized through the use of TOTP codes provided by Google Authenticator.

Chapter 15: Security Vulnerabilities

This chapter looks at important and potential security risks or vulnerabilities, points of interest pertaining to ASP.NET Web API, and things to look out for while building a secure, production-strength ASP.NET Web API. The coverage includes the top risks, per OWASP 2013, as well as best practices such as logging and validation.

Appendix: ASP.NET Web API Security Distilled

This appendix is a grand summary of the book, a recap of the various security mechanisms covered in the book. Because there is no good or bad mechanism in an absolute sense, the idea of this book is to present you with all the mechanisms and let you decide based on your needs. This appendix provides an overview of the options.

What You Need to Use This Book

At a bare minimum, you need Microsoft Visual Studio 2010, although all the code listings and samples in this book were developed using Visual Studio 2012 targeting the .NET Framework 4.5. If you use Visual Studio 2010, you will need the WIF runtime as well as the WIF SDK, which are available as stand-alone installations.

One important point to note is that WIF has been fully integrated into the .NET Framework starting with the .NET Framework 4.5, both the tooling as well as the classes. As part of this process, there are changes to the classes and the namespaces the classes were part of in the .NET Framework 4.0 compared to the .NET Framework 4.5. If you use Visual Studio 2010 and the .NET Framework 4.0, you will need to look at sources outside of this book to figure out the .NET Framework 4.0 equivalents of the code and configuration settings used in this book.

The language of choice for all the code written in this book is C#. Although there are Visual Basic.NET folks out there, it is not feasible to show the Visual Basic.NET equivalent, as that would bloat the size of the book. Understanding C# syntax is not that hard, after all!

ASP.NET Web API is part of ASP.NET MVC 4.0. It ships with Visual Studio 2012. Again, if you have the constraint of having to work with Visual Studio 2010, you must install ASP.NET MVC 4.0 by visiting <http://www.asp.net/mvc/mvc4>.

The bottom line is that Visual Studio 2012 and the .NET Framework 4.5 are strongly recommended. If you are really determined, you can get away with using Visual Studio 2010 targeting the .NET Framework 4.0. However, you will not be able to run the code samples provided with this book as is, and you will need to massage the C# code and configuration settings to make them work with the .NET Framework 4.0. All the samples in this book are coded and tested in Windows 7 using Visual Studio 2012 targeting the .NET Framework 4.5. Also, you need IIS 7.0.

The browser we use is mostly Internet Explorer 9.0; for some specific cases, we use Mozilla Firefox or Google Chrome. We also use the HTTP debugging tool called Fiddler. One of the chapters optionally uses Google Authenticator software that runs in iOS, BlackBerry, and Android-based mobile phones.

Who This Book Is For

No prior experience with .NET security is needed to read this book. All security-related concepts are introduced from basic principles and developed to the point where you can use them confidently in a professional environment. A good working knowledge and experience of C# and the .NET Framework are the only prerequisites to benefit from this book.