Community Experience Distilled

RESTful Java Web Services Security

Secure your RESTful applications against common vulnerabilities

René Enriquez
Andrés Salazar C.

[PACKT] PUBLISHING open source*

Table of Contents

RESTful Java Web Services Security

RESTful Java Web Services Security

First published: July 2014

Production reference: 1180714

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham B3 2PB, UK.

ISBN 978-1-78398-010-9

www.packtpub.com

Cover image by Vivek Thangaswamy (<vivekthangaswamy@yahoo.com>)

Credits

Authors

René Enríquez

Andrés Salazar C.

Reviewers

Erik Azar

Ismail Marmoush

Debasis Roy

Acquisition Editor

Vinay Argekar

Content Development Editor

Adrian Raposo

Technical Editor

Shruti Rawool

Copy Editor

Sayanee Mukherjee

Project Coordinators

Melita Lobo

Harshal Ved

Proofreaders

Simran Bhogal

Paul Hindle

Indexers

Hemangini Bari

Rekha Nair

Graphics

Abhinash Sahu

Production Coordinator

Arvindkumar Gupta

Cover Work

Arvindkumar Gupta

About the Authors

René Enríquez is currently a software architect for a multinational company headquartered i India. He has previously worked on many projects related to security implementation using frameworks such as JAAS and Spring Security to integrate many platforms based on the Web, BPM, CMS, and web services for government and private sector companies. He is a technology and innovation enthusiast, and he is currently working with several programming languages. He has achieved the following certifications:

- Oracle Certified Professional, Java SE 6 Programmer
- Microsoft Technology Associate
- Cisco Network Operating Systems

Over the past few years, he has worked as a software consultant on various projects for private and government companies and as an instructor of courses to build enterprise and mobile applications. He is also an evangelist of best practices for application development an integration.

Andrés Salazar C. is currently working at one of the most prestigious government companie in Ecuador, performing tasks related to software development and security implementation based on JAAS and digital signatures for secure applications. He also has extensive knowledge of OAuth implementation on web projects. He is a technology and Agile enthusias and he has worked on several projects using the JEE technology and TDD. He has achieved the following certifications:

- Oracle Certified Professional, Java SE 6 Programmer
- Certified Scrum Developer

About the Reviewers

Erik Azar is a professional software developer with over 20 years of experience in the areas of system administration, network engineering and security, development, and architecture. Having worked in diverse positions in companies ranging from start-ups to Fortune 500 companies, he currently works as a REST API architect for Availity, LLC in Jacksonville, FL. He is a dedicated Linux hobbyist who enjoys kernel hacking while experimenting with Raspberry Pi and BeagleBone Black boards. In his spare time, he works on solutions using embedded microprocessor platforms, Bluetooth 4.0, and connects to the cloud using RESTfu APIs.

Ismail Marmoush is a Java and Machine Learning Certified Expert. He has published the open source projects RESTful Boilerplates for IAAS and PAAS (GAE), an artificial neural network framework, and crawlers/dataminers and some language code examples. You can find more about him, his work, and his tutorials on his personal blog (http://marmoush.com).

Thanks to my family and the Packt Publishing team.

Debasis Roy is working as the Team Lead / Scrum Master of the sports team for Vizrt Bangladesh based at Dhaka. He has 7 years of professional working experience as a softwa engineer in Java/C++-relevant technologies.

He has been working at Vizrt for the past 5 years. He started his journey here with a product called the Online Suite, also known as Escenic Content Engine/Studio, and he is now continuing with products related to Viz Sports. Vizrt provides real-time 3D graphics, studio automation, sports analysis, and asset management tools for the broadcast industry—interactive and virtual solutions, animations, maps, weather forecasts, video editing, and compositing tools.

Previously, he worked at SDSL/AfriGIS for 2 years, where he was involved mainly in the projects, Marbil and Grid. AfriGIS is a technology innovation company that creates geograph information and communication solutions.

Support files, eBooks, discount offers, and more

You might want to visit www.PacktPub.com for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at <service@packtpub.com> for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

http://PacktLib.PacktPub.com

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktL today and view nine entirely free books. Simply use your login credentials for immediate access.

This book is dedicated to my wife and son, who supported me through so many days and nights of work and gave me their love and support; my brother, who has always lent me his support; my father, who has been an example of struggle and tireless work; my mother,

who has always been concerned about me and has supported me throughout life, gracias mami; and finally, my great friends, who have always been supportive of me.

--René Enríque

I dedicate this book to my family. It is because of the work and love of my parents that I have had the chance to study and become a professional software engineer, and because of the support and love of my sisters that I want to keep improving myself. Also, I want to dedicate this book to my grandmother, Mariana, who is the strongest person in the world. Muchas gracias abuelita! Finally, I dedicate the book to my bear man, Steve, for his support and English lessons.

--Andrés Salazar C

Preface

The inherent advantages of the use of web services in computer systems development are the same that create the need for security management over them. Today, we can say that no company is able to work in complete isolation, without the need to interact with others and share and consume information. Furthermore, this is the most important asset of any company. For this reason, these requirements are also common between lines of code. This book presents real scenarios with applicable solutions, leading you by the hand all the way, s you can easily learn solutions and implementations that will resolve the most common needs that can arise.

RESTful web services offer several advantages over those based on SOAP. For example, when handling data types, depending on the programming language or the libraries you use t create them, you can find inconsistencies when using empty values ("") instead of NULL. Als you may find difficulties in mapping complex objects and compatibility issues in file transferrin when using different versions of libraries to create/consume the web service. In certain situations, even when consuming a web service created in Java from a .NET application, it ends up creating a service implemented in Java in the middle of both. This does not occur in RESTful web services, since in this case, the functionality is exposed through HTTP method invocations.

In order to protect information, the world of securities has many features that help to achieve this. For example, understanding how some issues such as authentication and authorization assist in the implementation of any selected mechanism, where the main objective is to make our applications safer and secure, is essential. The selection of each of the different ways to secure applications goes along with the problem you want to resolve; for this, we show usage scenarios for each of them.

Many times, we have seen large organizations spend time and effort in creating their own implementations to handle securities rather than using the standard that has already resolved what we need. Through the knowledge that we want to share with you, we hope to avoid this process of reinventing the wheel.

What this book covers

Chapter 1, *Setting Up the Environment*, helps us create our first functional application, something very similar to a *Hello World* example, but with some more functionality and very close to the real world. The main aim of this chapter is to familiarize ourselves with the tools we are going to use.

Chapter 2, *The Importance of Securing Web Services*, goes through all possible models of authentication in the Java platform. For your better understanding, we will go step by step an dive deep into how we can leverage each available authentication model. We will show you how the information is exposed and how it can be intercepted by third parties, and we will pla with Wireshark, which is a very good tool to explain it.

Finally, in this chapter, we will review the differences between authentication and authorization. Both concepts are very important and definitely impossible to put aside in the context of securities terms.

Chapter 3, *Security Management with RESTEasy*, shows how RESTEasy offers mechanisms to handle security, starting from a fairly basic model (coarse-grained) to a more elaborate on (fine-grained) in which you can perform more exhaustive controls, including managing not on configuration files, but also programmatical files.

Chapter 4, *RESTEasy Skeleton Key*, helps us study the OAuth implementation along with the token bearer implementation and Single Sign-On. All of them are used in order to limit the wa the resources are shared. As always, you will get hands-on with code and real examples. We want to show you how sharing resources and information between applications through these technologies has turned into one of the most useful and powerful techniques by allowing clients or users to use their credentials only once to access several services, limiting the access to third-party applications to your information or data, and implementing access control through the token bearer. You will learn to apply these technologies and concepts in order to build secure and flexible applications.

Chapter 5, *Digital Signatures and Encryption of Messages*, helps us understand the benefits of digital signatures using a simple example; you'll notice how the message's receiver can validate the identity of the sender. In addition, we will simulate when an external agent modifies data in transit and see how digital signatures can help us to detect it, in order to avoid working with corrupted data.

Finally, we will explain SMIME for body encryption and how it works, with an example that encrypts requests and responses for your better understanding.

What you need for this book

In order to implement and test all the examples in this book, we will use many free tools, suc as the following:

- Eclipse IDE (or any other Java IDE)
- JBoss AS 7
- Maven
- Wireshark
- SoapUI

Who this book is for

This book is intended for developers, software analysts, architects, or people who work with software development and RESTful web services. This book requires some previous knowledge of object-oriented programming concepts in Java or any other language.

No previous knowledge on security models is required because we explain the theory and apply it on practical examples in this book.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "We are going to modify the `web.xml` file."

A block of code is set as follows:

```
private boolean isUserAllowed(final String username, final String
password, final Set<String> rolesSet) {
    boolean isAllowed = false;
    if (rolesSet.contains(ADMIN)) {
      isAllowed = true;
    }
    return isAllowed;
  }
}
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
final List<String> authorizationList =
headersMap.get(AUTHORIZATION_PROPERTY);
```

Any command-line input or output is written as follows:

```
mvn clean install
```

New terms and **important** words are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "From the pop-up window, select the **SSL Settings** tab."

> ## Note
>
> Warnings or important notes appear in a box like this.

> ## Tip
>
> Tips and tricks appear like this.

Reader feedback

Feedback as suggestions or comments from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of and also to improve the way we transmit knowledge.

To send us general feedback, simply send an e-mail to <feedback@packtpub.com>, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you t get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at http://www.packtpub.com. If you purchased this book elsewhere, you can visit http://www.packtpub.com/support and register to have the files e-mailed directly to you. Also we highly suggest obtaining the source code from GitHub available at https://github.com/restful-java-web-services-security.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata please report them by visiting http://www.packtpub.com/submit-errata, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from http://www.packtpub.com/support.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at <copyright@packtpub.com> with a link to the suspected pirated material

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at <questions@packtpub.com> if you are having a problem with any aspec of the book, and we will do our best to address it.

Chapter 1. Setting Up the Environment

We extend you a very warm welcome to the first chapter of our journey. Let's give you an idea of what you will achieve here. After reading this chapter, you will have the basic and stimulating knowledge you need to set up a development environment to work with RESTful web services. Then, you will familiarize yourself with the development of a very basic project related to it. In addition, by the end, you will have a very clear idea of how to create applications using RESTful web services and how you can achieve this. This chapter will give you the information you need to work with web services of this kind in a very easy and comprehensive way.

In this chapter, we will cover the following topics:

- Installing the development environment
- Creating our first RESTful web services application
- Testing the RESTful web service

Downloading tools

First, we must obtain our work tools so that we get our hands into code. Tools specified here are used around the world, but you are free to choose your tools. Remember, "Tools do not make the artist". It doesn't matter if you use Windows, MAC OS X, or Linux; tools are available for every OS.

Let's explain briefly what each tool is for. We will develop the examples using Eclipse as our IDE, JBoss AS 7.1.1.Final as our application server, Maven to automatize the build process, and SoapUI as a tool to test the functionality of web services that we will create. In addition, we suggest that you should install the latest version of JDK, which is JDK 1.7.x. For help, we have obtained and included some links that you need to use to get the software to implemen the first example. Each link gives you more information about each tool, which can be profitable as you learn something about each one if you don't know about them already.

Downloading links

The following tools have to be downloaded:

- Eclipse IDE for Java EE Developers 4.3 (http://www.eclipse.org/downloads/)
- JBoss AS 7.1.1 Final (http://www.jboss.org/jbossas/downloads/)
- Apache Maven 3.1.1 or higher (http://maven.apache.org/download.cgi)
- SoapUI 4.6 or higher (http://www.soapui.org/)
- JDK 1.7.x (http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html)

Creating the base project

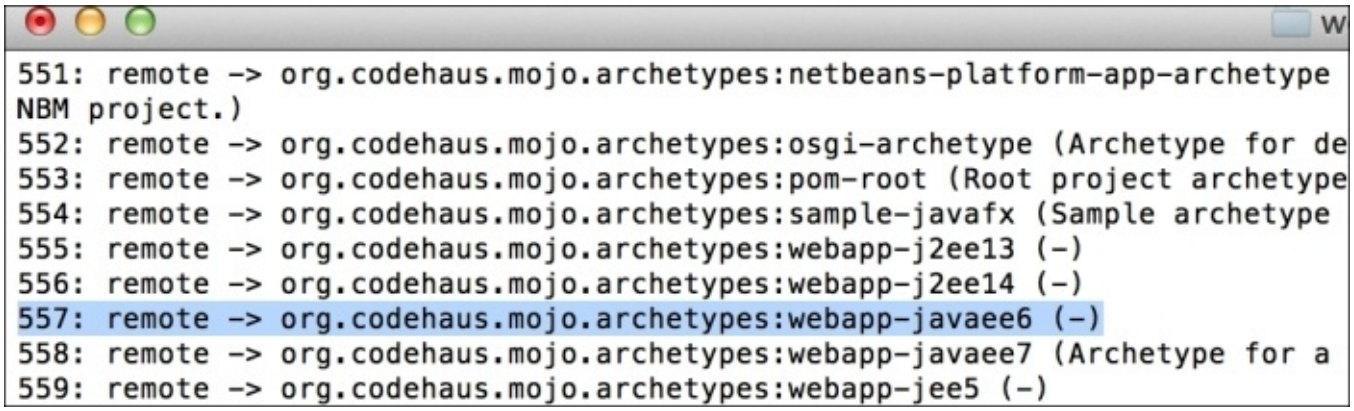
In order to make the process of building our sample project easier, we will use Maven. This wonderful software will create a base project at the blink of an eye, and our project can be easily compiled and packaged without depending on a specific IDE.

Maven uses archetypes for a specific kind of project. The archetypes are project templates that have been previously created; they allow us to create all kinds of applications from Java desktop applications to multimodule projects, where the EAR can contain several artifacts such as JAR and WAR. Its main objective is to get users up and running as quickly as possible by providing a sample project that demonstrates many of the features of Maven. If you want to learn more about Maven, you can find more information by visiting http://maven.apache.org/.

However, the information we described here is enough to keep moving on. We will use an archetype in order to create a basic project; if we want to be more specific, we will use an archetype to create a web application with Java. To do this, we will type the following command line in a terminal:

```
mvn archetype:generate
```

When we execute this command line in a terminal, we will obtain all available archetypes in Maven's repository. So, let's look for the archetype we need in order to create our web application; its name is `webapp-javaee6`, and it belongs to the group `org.codehaus.mojo.archetypes`. Also, we can search through it using a number that represents its ID; this number is `557`, as shown in the following screenshot. We recommend that you search by the name as the numbers are likely to change because some other archetypes may be added later:



Several questions will appear; we must provide the respective information for each question. Maven will use this information to create the archetype we selected before, as shown in the following screenshot:

```
Choose org.codehaus.mojo.archetypes:webapp-javaee6 version:
1: 1.0
2: 1.0.1
3: 1.0.2
4: 1.1
5: 1.2
6: 1.3
7: 1.4
8: 1.5
Choose a number: 8: 8
Define value for property 'groupId': : com.packtpub
Define value for property 'artifactId': : resteasy-examples
Define value for property 'version':  1.0-SNAPSHOT: :
Define value for property 'package':  com.packtpub: :
Confirm properties configuration:
groupId: com.packtpub
artifactId: resteasy-examples
version: 1.0-SNAPSHOT
package: com.packtpub
```

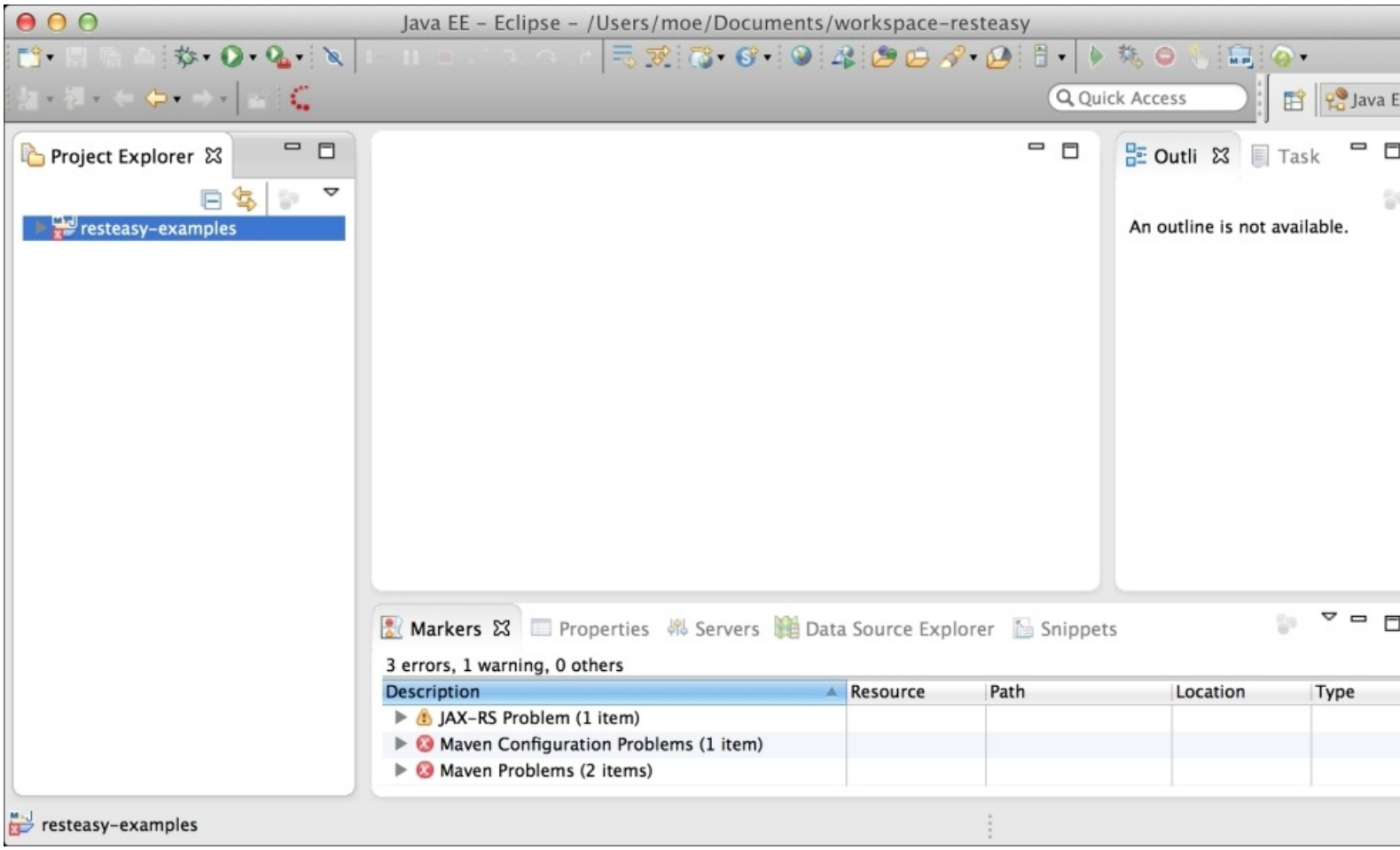
As you have probably noticed, each question asks you to define a property, and each property is explained as follows:

- `groupId`: This property represents the company's domain reversed order; this way we ca recognize which company is the code's owner
- `artifactId`: This property represents the project's name
- `version`: This property represents the project's version
- `package`: This property represents the base package's name where classes are going to be added

Class names and package names together shape the class's full name. This full name allows the class names to be identified in a unique way. Sometimes, when there are several classes with the same name, the package name helps to identify which library it belongs to.

The next step is to put the project into Eclipse's workspace; to do this, we must import our project into Eclipse by navigating through **File** | **Import** | **Maven** | **Existing Maven Projects**.

We should see the project in the IDE, as shown in the following screenshot:



Before moving on, let's fix the problems that have occurred in the file `pom.xml`.

The error shown in the following code is related to a bug that comes from Eclipse and Maven integration. In order to fix this, we have to add the `<pluginManagement>` tag after the `<build>` tag.

The `pom.xml` file should look like the following:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.packtpub</groupId>
  <artifactId>resteasy-examples</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>war</packaging>

  . . .

  <build>
    <pluginManagement>
      <plugins>
        <plugin>
          . . .
        </plugin>
      </plugins>
```

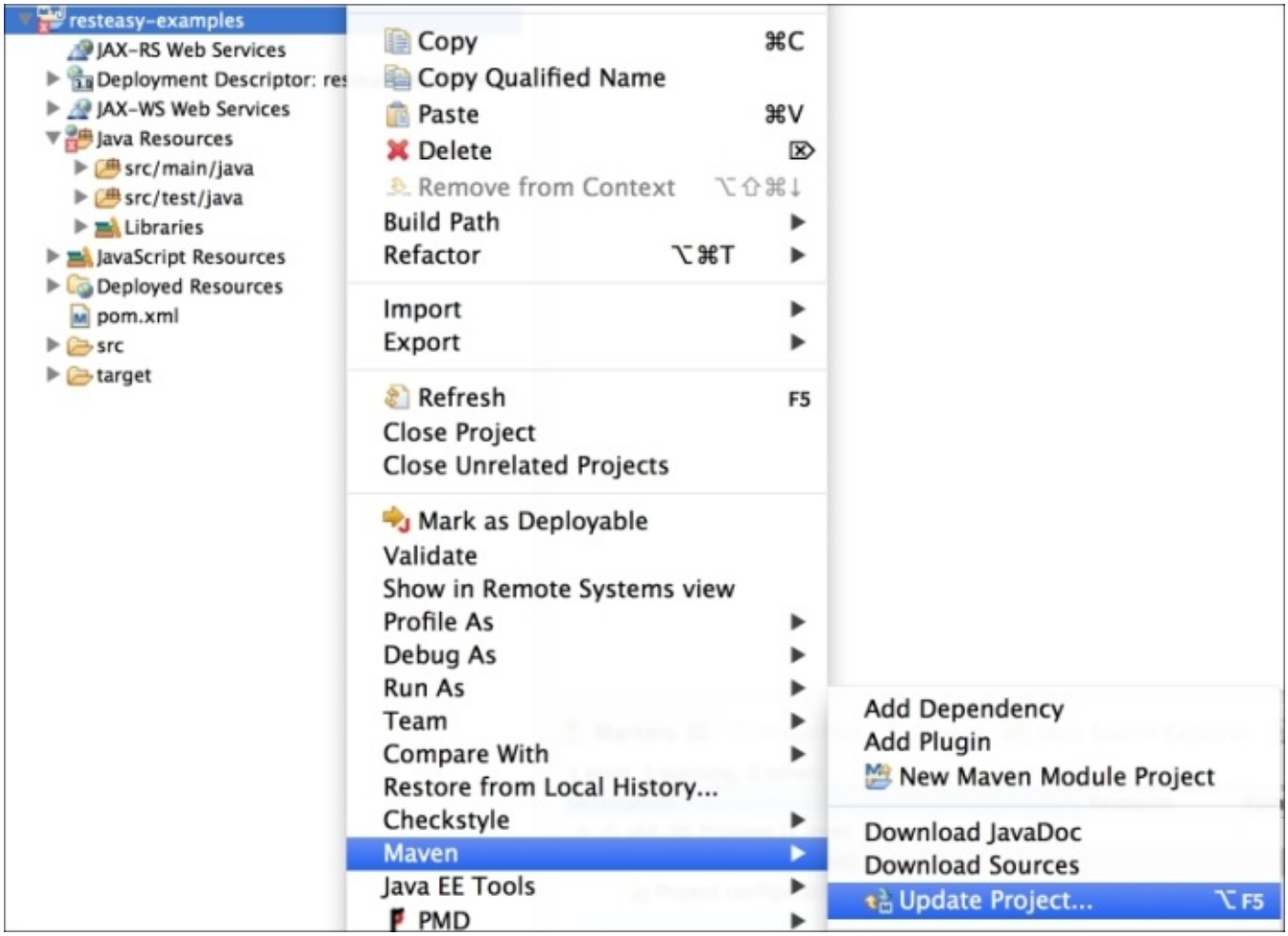
```
        </pluginManagement>
    </build>

</project>
```

Tip

Downloading the sample code

You can download the sample code files for all Packt books you have purchased from your account at http://www.packtpub.com. If you purchased this book elsewhere, you can visit http://www.packtpub.com/support and register to have the files e-mailed directly to you. Also, we highly suggest obtaining the source code from GitHub available at https://github.com/restful-java-web-services-security.

This will fix the error, and now we only need to update Maven's configuration in the project, a shown in the following screenshot:



After refreshing the project, the errors should go away because when we update Maven's configuration we are actually updating our project's dependencies, such as missing libraries. Through this, we will include them in our project and errors will disappear.

Inside the `src/main/webapp` path, let's create the `WEB-INF` folder.