**Table of Contents**

**Preface**

Introduction

Why OWASP Juice Shop exists

Architecture overview

**Part I - Hacking preparations**

Hacking preparations

Running OWASP Juice Shop

Vulnerability categories

Challenge tracking

Hacking exercise rules

Walking the "happy path"

Customization

Hosting a CTF event

**Part II - Challenge hunting**

Challenge hunting

Finding the Score Board

Injection

Broken Authentication

Sensitive Data Exposure

XML External Entities (XXE)

Improper Input Validation

Broken Access Control

Security Misconfiguration

Cross Site Scripting (XSS)

Insecure Deserialization

Vulnerable Components

Security through Obscurity

Unvalidated Redirects

Broken Anti-Automation

Cryptographic Issues

Miscellaneous

1.1

1.2

1.3

2.1

2.2

2.3

2.4

2.5

2.6

2.7

2.8

3.1

3.2

3.3

3.4

3.5

3.6

3.7

3.8

3.9

3.10 3.11 3.12 3.13 3.14 3.15 3.16 3.17

1

**Part III - Getting involved**

Getting involved

Provide feedback

Contribute to development

Codebase 101

Help with translation

Hacking Instructor tutorial scripts

Donations

**Appendix**

Challenge solutions

Trainer's guide

Troubleshooting

Jingle lyrics

Integration

Monitoring

Chatbot training data

**Postface**

About this book

4.1 4.2 4.3 4.4 4.5 4.6 4.7

5.1 5.2 5.3 5.4 5.5 5.6 5.7

6.1

2

**Pwning OWASP Juice Shop**

Written by Björn Kimminich

This is the official companion guide to the **OWASP Juice Shop** application. Being a web application with a vast number of intended security vulnerabilities, the OWASP Juice Shop is supposed to be the opposite of a *best practice* or *template application* for web developers: It is an awareness, training, demonstration and exercise tool for security risks in modern web applications. The OWASP Juice Shop is an open-source project hosted by the non-profit Open Web Application

Security Project® (OWASP) and is developed and maintained by volunteers. The content of this book was written for v12.0.2 of OWASP Juice Shop.

3

The book is divided into three parts:

**Part I - Hacking preparations**

Part one helps you to get the application running and to set up optional hacking tools.

**Part II - Challenge hunting**

Part two gives an overview of the vulnerabilities found in the OWASP Juice Shop including hints how to find and exploit them in the application.

**Part III - Getting involved**

Part three shows up various ways to contribute to the OWASP Juice Shop open source project.

*Please be aware that this book is not supposed to be a comprehensive introduction to Web Application Security in general. For every category of vulnerabilities present in the OWASP Juice Shop you will find a brief explanation - typically by quoting and referencing to existing content on the given topic.*

**Download a .pdf, .epub, or .mobi file from:**

https://leanpub.com/juice-shop (official release)

**Read the book online at:**

https://pwning.owasp-juice.shop

**Contribute content, suggestions, and fixes on GitHub:**

https://github.com/bkimminich/pwning-juice-shop

**Official OWASP Juice Shop project homepage:**

https://owasp-juice.shop



Open Web Application Security Project and OWASP are registered trademarks of the OWASP Foundation, Inc. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

4

**Why the Juice Shop exists**

To the unsuspecting user the Juice Shop just looks like a small online shop which sells - *surprise!* - fruit & vegetable juice and associated products. Except for the entirely overrated payment and delivery aspect of the e-commerce business, the Juice Shop is fully functional. But this is just the tip of the iceberg. The Juice Shop contains 100 challenges of varying difficulty where you are supposed to exploit underlying security vulnerabilities. These vulnerabilities were intentionally planted in the application for exactly that purpose, but in a way that actually happens in "real-life" web development as well!

Your hacking progress is tracked by the application using immediate push notifications for successful exploits as well as a score board for progress overview. Finding this score board is actually one of the (easiest) challenges! The idea behind this is to utilize gamification techniques to motivate you to get as many challenges solved as possible - similar to unlocking achievements in many modern video games.

Development of the Juice Shop started in September 2014 as the author's personal initiative, when his employer needed a more modern security training exercise environment for an in-house web application. The previously used environment was still from the era of server-side rendered ASP/JSP/Servlet and did not reflect the reality of current web technology. The Juice Shop was developed as open-source software without any corporate branding right from the beginning. By the end of 2014, most of the current e-commerce functionality was up and running

along with an initial number of planted vulnerabilities. Over the years more variants of vulnerabilities were added. In parallel, the application was kept up to-date with the latest web technology (e.g. WebSockets and OAuth 2.0) and frontend frameworks (i.e. by migrating from AngularJS with Bootstrap to Angular with Material Design). Some of these additional capabilities brought the chance to add corresponding vulnerabilities - and the list of challenges has been growing ever since.

Apart from the hacker and awareness training use case, penetration testing tools and automated security scanners are invited to use the Juice Shop as a sort of guinea pig-application to check how well their products cope with JavaScript heavy application frontends and REST APIs.

**Why *OWASP* Juice Shop?**

5

The Open Web Application Security Project (OWASP) is a nonprofit foundation that works to improve the security of software. Our programming includes:

Community-led open source software projects Over 275 local chapters worldwide Tens of thousands of members Industry-leading educational and training conferences We are an open community dedicated to enabling organizations to conceive, develop, acquire, operate, and maintain applications that can be trusted. All of our projects, tools, documents, forums, and chapters are free and open to anyone interested in improving application security. The OWASP Foundation launched on December 1st, 2001, becoming incorporated as a United States non-profit charity on April

21, 2004.

1

Two years after its inception the Juice Shop was submitted and accepted as an *OWASP Tool Project* by the Open Web Application Security Project in September 2016. This move increased the overall visibility and outreach of the project significantly, as it exposed it to a large community of application security practitioners.

Once in the OWASP project portfolio it took only eight months until Juice Shop was promoted from the initial *Incubator* maturity level to *Lab Projects* level. By the end of July 2018 the Juice Shop was promoted to the final *Flagship* maturity stage for OWASP projects.

**Why the name "Juice Shop"?**

In German there is a dedicated word for *dump*, i.e. a store that sells lousy wares and does not exactly have customer satisfaction as a priority: *Saftladen*. Reverse translating this separately as *Saft* and *Laden* yields *juice* and *shop* in English. That is where the project name comes from. The fact that the initials *JS* match with those commonly used for *JavaScript* was purely coincidental and not related to the choice of implementation technology.

**Why the logo?**

Other than the name, the Juice Shop logo was designed explicitly with *JavaScript* in mind:

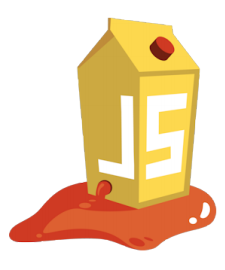
6

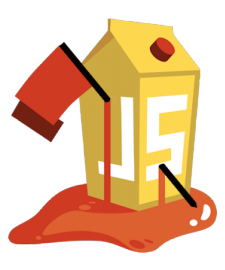
The author's idea was to convert one of the (unofficial but popular) *JavaScript* shield-logos into a **leaking juice box** because it had a quite matching shape for this shenanigans:



In 2017 the logo received a facelift and a spin-off when the Juice Shop introduced its Capture-the-flag extension (which is discussed in its own chapter Hosting a CTF event):

7





**Why yet another vulnerable web application?**

A considerable number of vulnerable web applications already existed before the Juice Shop was created. The OWASP Vulnerable Web Applications Directory (VWAD) maintains a list of these applications. When the Juice Shop came to life there were only *server-side rendered* applications in the VWAD, but *Rich Internet Application (RIA)* or *Single Page Application (SPA)* style applications were already a commodity at that time. Juice Shop was meant to fill that gap.

8

Many of the existing vulnerable web applications were very rudimentary in their functional scope. So the aim of the Juice Shop was also to give the impression of a functionally complete e-commerce application that could actually exist like this in the wild.

1

. hhttps://owasp.org/about/ ↩

9

**Architecture overview**

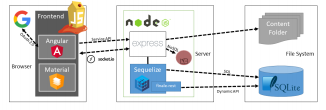
The OWASP Juice Shop is a pure web application implemented in JavaScript and TypeScript (which is compiled into regular JavaScript). In the frontend the popular Angular framework is used to create a so-called *Single Page Application*. The user interface layout is implementing Google's Material Design using Angular Material components. It uses Angular Flex-Layout to achieve responsiveness. All icons found in the UI are originating from the Font Awesome library.

JavaScript is also used in the backend as the exclusive programming language: An Express application hosted in a Node.js server delivers the client-side code to the browser. It also provides the necessary backend functionality to the client via a RESTful API. As an underlying database a light-weight SQLite was chosen, because of its file-based nature. This makes the database easy to create from scratch programmatically without the need for a dedicated server. Sequelize and finale-rest are used as an abstraction layer from the database. This allows using dynamically created API endpoints for simple interactions (i.e. CRUD operations) with database resources while still allowing the execution of custom SQL for more complex queries.

As an additional data store, a MarsDB is part of the OWASP Juice Shop. It is a JavaScript derivative of the widely used MongoDB NoSQL database and compatible with most of its query/modify operations.

The push notifications that are shown when a challenge was successfully hacked, are implemented via WebSocket Protocol. The application also offers convenient user registration via OAuth 2.0 so users can sign in with their Google accounts.

The following diagram shows the high-level communication paths between the client, server and data layers:

10

**Part I - Hacking preparations**

OWASP Juice Shop offers multiple ways to be deployed and used. The author himself has seen it run on

restricted corporate Windows machines

heavily customized Linux distros

all kinds of Apple hardware

overclocked Windows gaming notebooks

Chromebooks with native Linux support

various cloud platforms

Chance is pretty high that you will be able to get it running on your computer as well. This part of the book will help your install and run the Juice Shop as well as guide you through the application and some fundamental rules and hints for hacking it.

Should you run into issues during installation or launch of the application, please do not hesitate to ask for help in the community chat or by opening a GitHub issue! Please just make sure that you flipped through the appendix on troubleshooting first.

11

**Running OWASP Juice Shop System requirements**

To run a single instance of Juice Shop the following memory and CPU requirements apply. These resources are needed for the Juice Shop application process itself. Any additional resources needed by your environment (e.g. Docker or Vagrant) come on top.

**Minimum** system specification

128 MB RAM

100 millicpu CPU

300 MB free disk space

**Recommended** system specification

256 MB RAM

200 millicpu CPU

800 MB free disk space

*If installing from sources an additional 700 MB free disk space are required for the Git history in both minimum and recommended spec.*

**Run options**

In the following sections you find step-by-step instructions to deploy a running instance of OWASP Juice Shop for your personal hacking endeavours.

**One-click cloud instance**

****

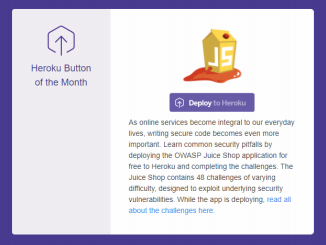
The quickest way to get a running instance of Juice Shop is to click the *Deploy to Heroku* button in the *Setup* section of the README.md on GitHub. You have to log in with your Heroku account and will then receive a single instance (or *dyno* in Heroku lingo) hosting the application. If you have forked the Juice Shop repository on GitHub, the *Deploy to Heroku* button will deploy your forked version of the application. To deploy the latest official version you must use the button of the original repository at https://github.com/bkimminich/juice-shop.

As the Juice Shop is supposed to be hacked and attacked - maybe even with aggressive brute-force scripts or automated scanner software - one might think that Heroku would not allow such activities on their cloud platform. Quite the opposite! When describing the intended use of Juice Shop to the Heroku support team they answered with:

That sounds like a great idea. So long as you aren't asking people to DDoS it that should be fine. People are certainly welcome to try their luck against the platform and your app so long as it's not DDoS.

12

As a little related anecdote, the OWASP Juice Shop was even crowned Heroku Button of the Month in November 2017 and once more in March 2019:

**Local installation**

To run the Juice Shop locally you need to have Node.js installed on your computer. The Juice Shop officially runs on versions 10.x, 12.x and 14.x of Node.js, closely following the official Node.js Long-term Support Release Schedule. During development and Continuous Integration (CI) the application is automatically tested with these current versions of Node.js. The officially recommended version to run Juice Shop is either the most recent *Long-term Support (LTS)* version or the *Current Release* version. Therefore Juice Shop recommends Node.js 14.x for its own v12.0.2 release.

**From sources**

1. Install Node.js on your computer.

2. On the command line run git clone https://github.com/bkimminich/juice shop.git .

3. Go into the cloned folder with cd juice-shop

4. Run npm install . This only has to be done before the first start or after you changed the source code.

5. Run npm start to launch the application.

6. Browse to http://localhost:3000

**From pre-packaged distribution**

1. Install a 64bit Node.js on your Windows, MacOS or Linux machine. 2. Download juice-shop-<version>\_<node-version>\_<os>\_x64.zip (or .tgz ) attached to the latest release on GitHub.

13

3. Unpack the archive and run npm start in unpacked folder to launch the application

4. Browse to http://localhost:3000

**Docker image**

You need to have Docker installed to run Juice Shop as a container inside it. Following the instructions below will download the current stable version (built from master branch on GitHub) which internally runs the application on the currently recommended Node.js version 14.x.

1. Install Docker on your computer.

2. On the command line run docker pull bkimminich/juice-shop to download the latest image described above.

3. Run docker run -d -p 3000:3000 bkimminich/juice-shop to launch the container with that image.

4. Browse to http://localhost:3000.

If you are using Docker on Windows - inside a VirtualBox VM - make sure that you also enable port forwarding from host 127.0.0.1:3000 to 0.0.0.0:3000 for TCP.

**Supported architectures**

The official Docker image is built automatically during CI/CD for linux/amd64 . Beginning with v11.1.1 an official linux/arm image is built for each tagged release as well as for latest . This build is currently executed manually on a RaspberryPi 4B model with Raspian 32bit. If an arm image is available, a compatible computer will automatically pull that image instead of the amd64 version when running docker pull bkimminich/juice-shop .

**Vagrant**

Vagrant is an open-source solution for building and maintaining virtual software development environments. It creates a Virtualbox VM that will launch a Docker container instance of the latest Juice Shop image v12.0.2.

1. Install Vagrant and Virtualbox

2. Run git clone https://github.com/bkimminich/juice-shop.git (or clone your own fork of the repository)

3. Run cd vagrant && vagrant up

4. Browse to 192.168.33.10

**Amazon EC2 Instance**

You need to have an account at Amazon Web Services in order to create a server hosting the Juice Shop there.

1. In the *EC2* sidenav select *Instances* and click *Launch Instance* 2. In *Step 1: Choose an Amazon Machine Image (AMI)* choose an *Amazon Linux AMI* or *Amazon Linux 2 AMI*

14

3. In *Step 3: Configure Instance Details* unfold *Advanced Details* and copy the script below into *User Data*

4. In *Step 6: Configure Security Group* add a *Rule* that opens port 80 for HTTP 5. Launch your instance

6. Browse to your instance's public DNS

#!/bin/bash

yum update -y

yum install -y docker

service docker start

docker pull bkimminich/juice-shop

docker run -d -p 80:3000 bkimminich/juice-shop

**AWS EC2 Launch Template**

1. In the *EC2* sidenav select *Launch Templates* and click *Create launch template*

2. Under *Launch template contents* select as *AMI ID* either *Amazon Linux AMI* or *Amazon Linux 2 AMI* (by using *Search for AMI*)

3. In the same section add a *Security Group* that opens port 80 for HTTP 4. Unfold *Advanced details* at the bottom of the screen and paste in the script above into *User Data*

5. Create your launch template

6. Launch one or multiple EC2 instances from your template

7. Browse to your instance's public DNS

**Azure Container Instance**

1. Open and login (via az login ) to your Azure CLI **or** login to the Azure Portal, open the *CloudShell* and then choose *Bash* (not PowerShell). 2. Create a resource group by running az group create --name <group name> -- location <location name, e.g. "centralus">

3. Create a new container by running az container create --resource-group <group name> --name <container name> --image bkimminich/juice-shop --dns-name-label <dns name label> --ports 3000 --ip-address public

4. Your container will be available at http://<dns name label>.<location name>.azurecontainer.io:3000

**Azure Web App for Containers**

1. Open your Azure CLI **or** login to the Azure Portal, open the *CloudShell* and then choose *Bash* (not PowerShell).

2. Create a resource group by running az group create --name <group name> -- location <location name, e.g. "East US">

3. Create an app service plan by running az appservice plan create --name <plan name> --resource-group <group name> --sku S1 --is-linux

4. Create a web app with the Juice Shop Docker image by running the following (on one line in the bash shell) az webapp create --resource-group <group name> -- plan <plan name> --name <app name> --deployment-container-image-name bkimminich/juice-shop

15

**Google Compute Engine Instance**

1. Login to the Google Cloud Console and open Cloud Shell.

2. Launch a new GCE instance based on the juice-shop container. Take note of the EXTERNAL\_IP provided in the output.

gcloud compute instances create-with-container owasp-juice-shop-app --container-image

1. Create a firewall rule that allows inbound traffic to port 3000 gcloud compute firewall-rules create juice-rule --allow tcp:3000

1. Your container is now running and available at http://<EXTERNAL\_IP>:3000/ **Installing a specific release version**

The installation instructions above will all give you the latest official release version of the Juice Shop. If you want to install a specific older version, you can easily do so by retrieving the corresponding tag from GitHub or Docker. For release v7.5.1 - which was the last version with the original AngularJS/Bootstrap frontend - for example:

From sources - Run git fetch --tags and then git checkout v7.5.1 before running npm install

Docker image - Run docker pull bkimminich/juice-shop:v7.5.1 instead of the usual docker pull bkimminich/juice-shop

From pre-packaged distribution - Just download the older release from https://github.com/bkimminich/juice-shop/releases or

https://sourceforge.net/projects/juice-shop/files/

To experience a preview of the next upcoming Juice Shop version you can do as follows:

Simply visit https://juice-shop-staging.herokuapp.com and take a look From sources - Run git fetch and then git checkout develop before running npm install

Docker image - Run docker pull bkimminich/juice-shop:snapshot instead of the usual docker pull bkimminich/juice-shop

ℹ Please be aware that support by the core team or community is limited (at best) for outdated and unreleased versions alike. To fully enjoy your OWASP Juice Shop experience, it is recommended to always use the latest version.

***Self-healing*-feature**

OWASP Juice Shop was not exactly designed and built with a high availability and reactive enterprise-scale architecture in mind. It runs perfectly fine and fast when it is attacked via a browser by a human. When under attack by an

16

automated tool - especially aggressive brute force scripts - the server might crash under the load. This could - in theory - leave the database and file system in an unpredictable state that prevents a restart of the application.

That is why - in practice - Juice Shop wipes the entire database and the folder users might have modified during hacking. After performing this *self-healing* the application is supposed to be restartable, no matter what kind of problem originally caused it to crash. For convenience the *self-healing* happens during the start-up (i.e. npm start ) of the server, so no extra command needs to be issued to trigger it.

**Single-user restriction**

There is one fundamental restriction that needs to be taken into account when working with the OWASP Juice Shop, especially in group trainings or lectures:

**A server instance of OWASP Juice Shop is supposed to be used by only a single-user!**

This restriction applies to all the Run Options explained above. It is technically necessary to make the *Self-healing*-feature work properly and consistently. Furthermore, when multiple users would attack the same instance of the Juice Shop all their progress tracking would be mixed leading to inevitable confusion for the individual hacker. The upcoming Challenge tracking chapter will illustrate this topic.

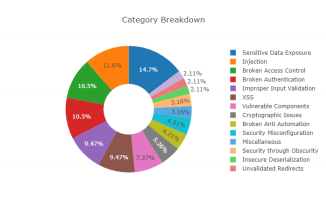
It should not go unmentioned that it is of course okay to have multiple users hack the same instance from a shared machine in a kind of *pair-hacking*-style.

If you want to centrally host Juice Shop instances for multiple users you find more information in section Hosting individual instances for multiple users of the trainer's guide.

17

**Vulnerability Categories**

The vulnerabilities found in the OWASP Juice Shop are categorized into several different classes. Most of them cover different risk or vulnerability types from well known lists or documents, such as OWASP Top 10, OWASP ASVS, OWASP Automated Threat Handbook and OWASP API Security Top 10 or MITRE's Common Weakness Enumeration. The following table presents a mapping of the Juice Shop's categories to OWASP, CWE and WASC threats, risks and attacks (without claiming to be complete).

**Category Mappings**

18

| **Category** | **OWASP** | **CWE** | **WASC** |
| --- | --- | --- | --- |
| Broken Access  Control | A5:2017, API1:2019,  API5:2019 | CWE-22,  CWE-285,  CWE-639 | WASC  02,  WASC  09,  WASC  16 |
| Broken Anti  Automation | OWASP-AT-004,  API4:2019, OWASP  AT-010, OAT-009, OAT 015, OAT-008 | CWE-362 | WASC  11,  WASC  21 |
| Broken  Authentication | A2:2017, API2:2019 | CWE-287,  CWE-352 | WASC  01,  WASC  49 |
| Cross Site  Scripting (XSS) | A7:2017 | CWE-79 | WASC  8 |
| Cryptographic  Issues | A3:2017 | CWE-326,  CWE-327,  CWE-328,  CWE-950 | - |
| Improper Input  Validation | ASVS V5, API6:2019 | CWE-20 | WASC  20 |
| Injection | A1:2017, API8:2019 | CWE-74,  CWE-89 | WASC  19,  WASC  28,  WASC  31 |
| Insecure  Deserialization | A8:2017 | CWE-502 | - |
| Miscellaneous | - | - | - |
| Security  Misconfiguration | A6:2017, A10:2017,  API7:2019, API9:2019, API10:2019 | CWE-209 | WASC  14,  WASC  15 |
| Security through Obscurity | - | CWE-656 | - |
| Sensitive Data  Exposure | A3:2017, API3:2019,  OTG-CONFIG-004 | CWE-200,  CWE-530,  CWE-548 | WASC  13 |
| Unvalidated  Redirects | A10:2013 | CWE-601 | WASC  38 |
| Vulnerable  Components | A9:2017 | CWE-829,  CWE-506,  CWE-1104 | - |
| XML External  Entities (XXE) | A4:2017 | CWE-611 | WASC  43 |

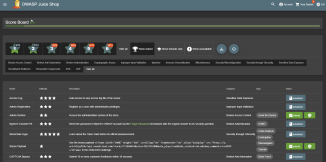
19

**Challenge tracking**

**The Score Board**

In order to motivate you to hunt for vulnerabilities, it makes sense to give you at least an idea what challenges are available in the application. Also you should know when you actually solved a challenge successfully, so you can move on to another task. Both these cases are covered by the application's score board.

On the score board you can view a list of all available challenges with a brief description. Some descriptions are *very explicit* hacking instructions. Others are just *vague hints* that leave it up to you to find out what needs to be done.

The challenges are rated with a difficulty level between ⭐ and ⭐⭐⭐⭐⭐⭐, with more stars representing a higher difficulty. To make the list of challenges less daunting, they are clustered by difficulty. By default only the 1-star challenges are unfolded. You can open or collapse all challenge blocks as you like. Collapsing a block has *no impact* on whether you can *solve* any of its challenges.

The difficulty ratings have been continually adjusted over time based on user feedback. The ratings allow you to manage your own hacking pace and learning curve significantly. When you pick a 5- or 6-star challenge you should *expect* a real challenge and should be less frustrated if you fail on it several times. On the other hand if hacking a 1- or 2-star challenge takes very long, you might realize quickly that you are on a wrong track with your chosen hacking approach.

Finally, each challenge states if it is currently *unsolved* or *solved*. The current overall progress is represented in a progress bar on top of the score board. Especially in group hacking sessions this allows for a bit of competition between the participants.

If not deliberately turned off (see Customization) you can hover over each *unsolved* label to see a hint for that challenge. If a "book" icon is also displayed within the label, you can click on it to be redirected to the corresponding hints section in Part 2 of this book.

20

**Challenge Filters**

Additional to the filtering by difficulty, you can filter the Score Board by challenge categories, e.g. to focus your hacking efforts on specific vulnerabilities. You can also hide all solved challenges to reduce the level of distraction on the Score Board.

�� Selecting *Show all* for all difficulties and all challenges might impact the load time of the Score Board significantly!

**Challenge Tags**

Starting with v12.0.0 tags were introduced to help classify challenges which either favor a certain hacking approach or share some trait orthogonal to the categories.

**Shenanigans** marks challenges which are not considered serious and/or realistic but exist more for entertainment

**Contraption** indicates that a challenge is not exactly part of a realistic scenario but might be a bit forced or crafted

**OSINT** marks challenges which require some Internet research or **social stalking** actvitiy outside the application

**Good Practice** highlights challenges which are less about vulnerabilities but promoting good (security) practices

**Danger Zone** marks potentially dangerous challenges which are disabled on Docker/Heroku by default due to RCE or other risks

**Good for Demos** highlights challenges which are suitable for live demos or awareness trainings

**Prerequisite** marks challenges which need to be solved before one or more other challenges can be (realistically) solved

**Brute Force** marks challenges where automation of some security tool or custom script is an option or even prerequisite

**Tutorial** marks challenges for which a Hacking Instructor script exists to assist newcomers

**Code Analysis** marks challenges where it can be helpful to rummage through some source code of the application or a third party

**Success notifications**

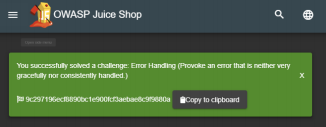
The OWASP Juice Shop employs a simple yet powerful gamification mechanism: Instant success feedback! Whenever you solve a hacking challenge, a notification is *immediately* shown on the user interface.

21

This feature makes it unnecessary to switch back and forth between the screen you are attacking and the score board to verify if you succeeded. Some challenges will force you to perform an attack outside of the Juice Shop web interface, e.g. by interacting with the REST API directly. In these cases the success notification will light up when you come back to the regular web UI the next time.

To make sure you do not miss any notifications they do not disappear automatically after a timeout. You have to dismiss them explicitly. In case a number of notifications "piled up" it is not necessary to dismiss each one individually, as you can simply Shift -click one of their *X*-buttons to dismiss all at the same time.

Depending on your application configuration, each challenge notification might also show a �� symbol with a character sequence next to it. If you are doing a hacking session just on your own, you can completely ignore this flag. The code is only relevant if you are participating in a CTF event. Please refer to chapter Hosting a CTF event for more information this topic.



**Automatic saving and restoring hacking progress**

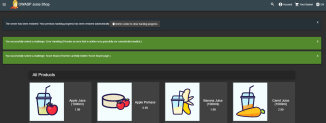
The *self-healing* feature - by wiping the entire database on server start - of Juice Shop was advertised as a benefit just a few pages before. This feature comes at a cost, though: As the challenges are also part of the database schema, they will be wiped along with all the other data. This means, that after every restart you

start with a clean 0% score board and all challenges in *unsolved* state.

22

To keep the resilience against data corruption but allow users to *pick up where they left off* after a server restart, your hacking progress is automatically saved whenever you solve a challenge - as long as you allow Browser cookies!

After restarting the server, once you visit the application your hacking progress is automatically restored:

The auto-save mechanism keeps your progress for up to 30 days after your previous hacking session. When the score board is restored to its prior state, a torrent of success notifications will light up - depending on how many challenges you solved up to that point. As mentioned earlier these can be bulk-dismissed by Shift -clicking any of the *X*-buttons.

If you want to start over with a fresh hacking session, simply click the *Delete cookie to clear hacking progress* button. After the next server restart, your score board will be blank.

**Manual progress and settings backup**

With the round *Backup* and *Restore* buttons on the Score Board you can save and later restore your hacking progress as well as language, Score Board filters, banner dismissal to a JSON file.

The backup format is independent of your system or browser, meaning you can use the backup file to conveniently transfer your progress and settings from one computer to another.

If during restore you see an error message Version X is incompatible with expected version Y your backup was taken before a semantically incompatible format change. The current backup schema version is 1.

**Hacking Instructor**

23



The built-in *Hacking Instructor* offers tutorials for some of the Juice Shop challenges. By default the welcome banner shown upon first launch of the application has a ��-button which will help you Find the carefully hidden 'Score Board' page.

On the Score Board itself you will then find similar ��-buttons on some of the challenges which will launch a corresponding tutorial for each as well. All tutorials consist of a scripted sequence of helpful hints and instructions.

24

The scripts often provide some interaction, like waiting for the user to make some specific input or having them visit another dialog before continuing. Some hints or instructions can be skipped by just clicking on them.

25

After successfully completing all steps of a tutorial, the Hacking Instructor will usually congratulate you and then go into hiding until summoned again for another hacking challenge via the Score Board.

ℹ The Hacking Instructor is a tool to help beginners getting started. It cannot offer a tutorial for *every challenge* as some are too complex or require too many steps outside the application. In Part III you can learn more about how to write Hacking Instructor tutorial scripts.

**Tutorial mode**

When using the Juice Shop in a classroom setup the trainer or teacher might want to set a slower pace at the beginning to give everyone a chance to get familiar with the application. Here the tutorial.yml configuration can be very useful,

26

which is available since v10.2.0 of Juice Shop. This mode hides all challenges without tutorials from the Score Board and disables all advanced filter options. In the tutorial mode challenges are only gradually unlocked by difficulty tiers.

Only when for example all 1-star challenges with a tutorial have been solved, the 2-star challenges with tutorials are displayed:

After solving **all** challenges with tutorials, the entire Score Board with all challenges is shown and all filters are enabled. Passing in the NODE\_ENV=tutorial environment variable will activate this mode.

**Potentially dangerous challenges**

Some challenges can cause potential harm or pose some danger for your computer, i.e. the XXE, SSTi and Deserialization challenges as well as two of the NoSQLi challenges and the possibility of an arbitrary file write. These simply cannot be sandboxed in a 100% secure way. These are only dangerous if you use

27

actually malicious payloads, so please do not play with payloads you do not fully understand. Furthermore be aware that all stored XSS vulnerabilities can - by their nature - be abused to perform harmful attacks on unsuspecting visitors.

For safety reasons all potentially dangerous challenges are disabled (along with their underlying vulnerabilities) in containerized environments. By default this applies to Docker and Heroku. These challenges are marked as 'unavailable' in the scoreboard as can be seen in the screenshot above.

To re-enable all challenges you can set the environment variable NODE\_ENV=unsafe or you can set safetyOverride: true in your own YAML configuration file. Please use the unsafe mode at your own risk, especially on publicly hosted instances.

28

**Hacking exercise rules**

✔ **Recommended hacking tools**

**Browser**

When hacking a web application a good internet browser is mandatory. The emphasis lies on *good* here, so you do *not* want to use Internet Explorer. Other than that it is up to your personal preference. Chrome and Firefox both work fine from the authors experience.

**Browser development toolkits**

When choosing a browser to work with you want to pick one with good integrated (or pluggable) developer tooling. Google Chrome and Mozilla Firefox both come with powerful built-in *DevTools* which you can open via the F12 -key.

When hacking a web application that relies heavily on JavaScript, **it is essential to your success to monitor the *JavaScript Console* permanently!** It might leak valuable information to you through error or debugging logs!

Other useful features of browser DevTools are their network overview as well as insight into the client-side JavaScript code, cookies and other local storage being used by the application.

29

**Tools for HTTP request tampering**

Tamper Chrome lets you monitor and - more importantly - modify HTTP requests *before* they are submitted from the browser to the server.

Mozilla Firefox has built-in tampering capabilities and does not need a plugin. On the *Network* tab of Firefox's DevTools you have the option to *Edit and Resend* every recorded HTTP request.

Tampering is extremely useful when probing for holes in the server-side validation logic. It can also be helpful when trying to bypass certain input validation or access restriction mechanisms, that are not properly checked *on the server* once more.

An API testing plugin like PostMan for Chrome allows you to communicate with the RESTful backend of a web application directly. Skipping the UI can often be useful to circumvent client-side security mechanisms or simply get certain tasks done faster. Here you can create requests for all available HTTP verbs ( GET ,

POST , PUT , DELETE etc.) with all kinds of content-types, request headers etc.

If you feel more at home on the command line, curl will do the trick just as fine as the recommended browser plugins.

30

**Scripting tools**

A small number of challenges is not realistically solvable manually unless you are cheating or are incredibly ��-lucky.

For these challenges you will require to write some scripts that for example can submit requests with different parameter values automatically in a short time. As long as the tool or language of choice can submit HTTP requests, you should be fine. Use whatever you are most familiar with.

If you have little experience in programming, best pick a language that is easy to get into and will give you results without forcing you to learn a lot of syntax elements or write much *boilerplate code*. Python, Ruby or JavaScript give you this simplicity and ease-of-use. If you consider yourself a "command-line hero", Bash or PowerShell will get the job done for you. Languages like Java, C# or Perl are probably less suitable for beginners. In the end it depends entirely on your preferences, but being familiar with at least one programming language is kind of mandatory if you want to get 100% on the score board.

In computer programming, boilerplate code or boilerplate refers to sections of code that have to be included in many places with little or no alteration. It is often used when referring to languages that are considered verbose, i.e.

1

the programmer must write a lot of code to do minimal jobs. **Penetration testing tools**

You *can* solve all challenges just using a browser and the plugins/tools mentioned above. If you are new to web application hacking (or penetration testing in general) this is also the *recommended* set of tools to start with. In case you have experience with professional pentesting tools, you are free to use those! And you are *completely free* in your choice, so expensive commercial products are just as fine as open source tools. With this kind of tooling you will have a competitive advantage for some of the challenges, especially those where *brute force* is a viable attack. But there are just as many multi-staged vulnerabilities in the OWASP Juice Shop where - at the time of this writing - automated tools would probably not help you at all.

In the following sections you find some recommended pentesting tools in case you want to try one. Please be aware that the tools are not trivial to learn - let alone master. Trying to learn about the web application security basics *and* hacking tools *at the same time* is unlikely to get you very far in either of the two topics.

**Intercepting proxies**

An intercepting proxy is a software that is set up as *man in the middle* between your browser and the application you want to attack. It monitors and analyzes all the HTTP traffic and typically lets you tamper, replay and fuzz HTTP requests in various ways. These tools come with lots of attack patterns built in and offer active as well as passive attacks that can be scripted automatically or while you are surfing the target application.

31

The open-source OWASP Zed Attack Proxy (ZAP) is such a software and offers many useful hacking tools for free:

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications. It is designed to be used by people with a wide range of security experience and as such is ideal for developers and functional testers who are new to penetration testing. ZAP provides

automated scanners as well as a set of tools that allow you to find security 2

vulnerabilities manually.

**Pentesting Linux distributions**

Instead of installing a tool such as ZAP on your computer, why not take it, add *several hundred* of other offensive security tools and put them all into a ready-to use Linux distribution? Entering Kali Linux and similar toolboxes:

Kali Linux is a Debian-based Linux distribution aimed at advanced Penetration Testing and Security Auditing. Kali contains several hundred

tools aimed at various information security tasks, such as Penetration 3

Testing, Forensics and Reverse Engineering.

The keyword in the previous quote is *advanced*! More precisely, Kali Linux is *easily overwhelming* when beginners try to work with it, as even the Kali development team states:

As the distribution’s developers, you might expect us to recommend that everyone should be using Kali Linux. The fact of the matter is, however, that Kali is a Linux distribution specifically geared towards professional penetration testers and security specialists, and given its unique nature, it is

**NOT** a recommended distribution if you’re unfamiliar with Linux [...]. Even 4

for experienced Linux users, Kali can pose some challenges.

32

Although there exist some more light-weight pentesting distributions, they basically still present a high hurdle for people new to the IT security field. If you still feel up to it, give Kali Linux a try!

**Internet**

You are free to use Google during your hacking session to find helpful websites or tools. The OWASP Juice Shop is leaking useful information all over the place if you know where to look, but sometimes you simply need to extend your research to the Internet in order to gain some relevant piece of intel to beat a challenge.

�� **Getting hints**

Frankly speaking, you are reading the *premium source of hints* right now! Congratulations! In case you want to hack more on your own than follow the breadcrumbs through the wood of challenges in part II, the most direct way to ask for specific hints for a particular challenge is the community chat on Gitter.im at https://gitter.im/bkimminich/juice-shop. You can simply log in to Gitter with your GitHub account.

If you prefer, you can also use the project's Slack channel at

https://owasp.slack.com/messages/project-juiceshop. You just need to self-invite you to OWASP's Slack first at https://owasp-slack.herokuapp.com. If you like it a bit more nostalgic, you can also join and post to the project Google group/mailing list at https://groups.google.com/a/owasp.org/forum/#!forum/juice-shop-project.

❌ **Things considered cheating**

**Reading a solution (** �� **) before trying**

33

The Challenge solutions appendix is there to help you in case you are stuck or have absolutely no idea how a specific challenge is solved. Simply going through the entire appendix back to back and follow the step-by-step instructions given there for each challenge, would deprive you of most of the fun and learning effect of the Juice Shop. You have been warned.

**Source code**

Juice Shop is supposed to be attacked in a "black box" manner. That means you cannot look into the source code to search for vulnerabilities. As the application tracks your successful attacks on its challenges, the code must contain checks to verify if you succeeded. These checks would give many solutions away immediately.

The same goes for several other implementation details, where vulnerabilities were arbitrarily programmed into the application. These would be obvious when the source code is reviewed.

Finally the end-to-end test suite of Juice Shop was built hack all challenges automatically, in order to verify they can all be solved. These tests deliver all the required attacks on a silver plate when reviewed.

**GitHub repository**

While stated earlier that "the Internet" is fine as a helpful resource, consider the GitHub repository https://github.com/bkimminich/juice-shop as entirely off limits. First and foremost because it contains the source code (see above).

Additionally it hosts the issue tracker of the project, which is used for idea management and task planning as well as bug tracking. You can of course submit an issue if you run into technical problems that are not covered by the Troubleshooting section of the README.md. You just should not read issues labelled challenge as they might contain spoilers or solutions.

Of course you are explicitly allowed to view the repository's README.md page, which contains no spoilers but merely covers project introduction, setup and troubleshooting. Just do not "dig deeper" than that into the repository files and folders.

**Database table Challenges**

The challenges (and their progress) live in one database together with the rest of the application data, namely in the Challenges table. Of course you could "cheat" by simply editing the state of each challenge from *unsolved* to *solved* by setting the corresponding solved column to 1 . You then just have to keep your fingers crossed, that nobody ever asks you to *demonstrate how* you actually solved all the 4- and 5-star challenges so quickly.

**Configuration REST API Endpoint**

34

The Juice Shop offers a URL to retrieve configuration information which is required by the Customization feature that allows redressing the UI and overwriting the product catalog: http://localhost:3000/rest/admin/application configuration

The returned JSON contains spoilers for all challenges that depend on a product from the inventory which might be customized. As not all customization can be prepared on the server side, exposing this REST endpoint is unavoidable for the Customization feature to work properly.

**Tutorial JavaScript file**

If enabled, the Hacking Instructor script tutorial-es2015.js (or tutorial-es5.js in legacy browsers) including all on-screen tutorials is loaded lazily by the *Score Board* and the *Welcome Banner*. You should exclude this file from all your manual or automated frontend code analysis. It contains step-by-step hints and unavoidably massive spoilers for several challenges via its condition checks that trigger progressing through each tutorial.

**Score Board HTML/CSS**

The Score Board and its features were covered in the Challenge tracking chapter. In the current context of "things you should not use" suffice it to say, that you could manipulate the score board in the web browser to make challenges *appear as solved*. Please be aware that this "cheat" is even easier (and more embarrassing) to uncover in a classroom training than the previously mentioned database manipulation: A simple reload of the score board URL will let all your local CSS changes vanish in a blink and reveal your *real* hacking progress. 1

. https://en.wikipedia.org/wiki/Boilerplate\_code ↩

2

. https://github.com/zaproxy/zap-core-help/wiki ↩

3

. http://docs.kali.org/introduction/what-is-kali-linux ↩

4

. http://docs.kali.org/introduction/should-i-use-kali-linux ↩

35

**Walking the "happy path"**

When investigating an application for security vulnerabilities, you should *never* blindly start throwing attack payloads at it. Instead, **make sure that you understand how it works** before attempting any exploits.

Before commencing security testing, understanding the structure of the application is paramount. Without a thorough understanding of the layout of the application, it is unlikely that it will be tested thoroughly. Map the target

1

application and understand the principal workflows.

A good way to gain an understanding for the application, is to *actually use it* in the way it was meant to be used by a normal user. In regular software testing this is often called "happy path" testing.

Also known as the "sunny day" scenario, the happy path is the "normal" path of execution through a use case or through the software that

implements it. Nothing goes wrong, nothing out of the normal happens, and 2

we swiftly and directly achieve the user's or caller's goal.

The OWASP Juice Shop is a rather simple e-commerce application that covers the typical workflows of a web shop. The following sections briefly walk you through these "happy path" use cases.

**Browse products**

When visiting the OWASP Juice Shop you will begin on the landing page #/ which initially displays all products offered in the shop. Clicking on the logo in the top left corner of the screen will always bring you back to this screen (or more precisely, to its alias #/search ).

This is of course the "bread & butter" screen for any e-commerce site. When you click on the small "eye"-button next to the price of a product, an overlay screen will open showing you that product details including a list of customer reviews for that product (if available). You can also enter a new (or edit an existing) product

review in this dialog. Authenticated users can upvote reviews they like.

36

You can use the *Search...* box in the navigation bar on the top of the screen to filter the table for specific products by their name and description. Using the controls at the bottom of the table, you can navigate through a the result list that exceeds the *Items per page* limit.

**User login**

You might notice that there seems to be no way to actually purchase any of the products. This functionality exists, but is not available to anonymous users. You first have to log in to the shop with your user credentials on the #/login page.

37

There you can either log in with your existing credentials (if you are a returning customer) or with your Google account.

**User registration**

In case you are a new customer, you must first register by following the corresponding link on the login screen to #/register . There you must enter your email address and a password to create a new user account. With these credentials you can then log in... and finally start shopping! During registration you also choose and answer a security question that will let you recover the account if you ever forget your password.

38

**Forgot Password**

By providing your email address, the answer to your security question and a new password, you can recover an otherwise inaccessible account.

39

**Choosing products to purchase**

After logging in to the application you will notice a "shopping cart"-icon in every row of the products table. Unsurprisingly this will let you add one or more products into your shopping basket. The *Your Basket* button in the navigation bar will bring you to the #/basket page, where you can do several things before actually confirming your purchase:

increase ("+") or decrease ("-") the quantity of individual products in the shopping basket

remove products from the shopping basket with the "trashcan"-button40

**Checkout**

During checkout you will be guided through a series of steps to set your delivery address, desired delivery method and credit card.

�� **Do not enter any real credit card or address data anywhere in the Juice Shop! Always remember that it is not a real shop and it is intentionally riddled with security and privacy flaws!**

In the *Add a coupon* section you can redeem a code for a discount. Unfold the *Other payment options* section to see links with donation and merchandise links of the Juice Shop open source project.

41

Finally you can click the *Checkout* button to issue an order. You will be forwarded to a confirmation of your order right away. It also includes a link to a printable PDF confirmation for your order and as well as a tracking link.

**User Menu**

Clicking the user icon right next to the application logo & title, you will give you access to several secondary use cases of the Juice Shop. This menu is obviously only available when you are logged in with your user account.

ℹ We will cover only a fraction of the available functionality from the user menu in the following sub-sections. It is recommended to explore the rest on your own before diving into any hacking exercises.

42

**User Profile**

Clicking you your email address in the user menu, you will get to the *User Profile* screen on /profile . Visiting it might break your user experience a bit, as it looks slightly less sophisticated than the rest of the shop's UI. It is fully functional nonetheless, as it allows you to upload a JPG -format picture of yourself (or link an existing Gravatar) and choose a username for your account.

**My saved addresses**

This page lists your saved addresses and provides you with the ability to edit or delete already saved addresses as well as add new ones.

43

�� **Do not enter real address data here!**

**My Payment Options**

This page lists your saved cards and provides you with the ability to delete already saved cards or to add new ones.

44

�� **Do not enter real credit card data here!**

**Juice Shop Wallet**

This page allows you to add money to your wallet and to check the existing balance. All the bonuses on your purchase are directly credited to your wallet.

**Order History**

This page allows you to view the details of all your current and previous orders and the status of their delivery.

45

**Privacy Policy**

This page informs you about the policies regarding the collection, use and disclosure of personal data when you use the OWASP Juice Shop and the choices you have when it comes to your data.

**Request Data Export**

This page allows you to obtain a copy of all your data saved in the Juice Shop.

46

**Request Data Erasure**

This page allows you to request a complete erasure of your account and any associated data from the Juice Shop.

**Change user password**

If you are currently logged in you will find the obligatory *Change Password* button in the navigation bar. On the #/privacy-security/change-password page you can then choose a new password. To prevent abuse you have of course to supply your current password to legitimate this change.

47

**2FA Configuration**

This page allows you to secure your account with an additional factor by providing you with a barcode to scan.

48

**Last Login IP**

This page displays the IP from which your account was last logged in. **Request Recycling Box**

When logged in you will furthermore see a *Recycle* button that brings you to the #/recycle page. This is a very innovative feature that allows eco-friendly customers to order pre-stamped boxes for returning fruit pressing leftovers to the Juice Shop.

49

For greater amounts of pomace the customer can alternatively order a truck to come by and pick it up at a chosen future date.

**Order Tracking**

Equipped with an order number from your confirmation PDF, you can invoke the #/track-order functionality by clicking *Track Orders*.

50

After entering a valid order number, you will be shown the products from your order along with a delivery status and expected delivery date.

*Just as there was no "real" payment was happening, you will hopefully understand that there is no "real" order delivery happening - no matter what the order tracking dialog suggested.*

**Menu**

The sidebar menu button left of the application logo reveals some more options to choose from.

51

**Customer Feedback**

Customers are invited to leave feedback about their shopping experience with the Juice Shop. Simply visit the #/contact page by clicking the *Customer Feedback* menu item. You might recognize that it is also possible to leave feedback as an anonymous user. The contact form is very straightforward with a free text *Comment* field and a *Rating* on a 1-5 stars scale. To prevent abuse, you have to solve a simple mathematical problem before being allowed to submit your feedback.

52

**Complain**

The *Complain?* menu item is shown only to logged in users. It brings you to the #/complain page where you can leave a free text *Message* and also attach an *Invoice* file in case you had some issues with a recent order at the Juice Shop.

**Support Chat**

In the *Support Chat* you can talk to an (almost) AI-powered chat bot and get answers to questions like product prices, deluxe membership benefits and more.

53

**About Us**

Like every proper enterprise, the OWASP Juice Shop has of course an #/about page titled *About Us*. There you find a summary of the interesting history of the shop along with a link to its official Terms of Use document. Additionally the page displays a fancy illustrated slideshow of all customer feedback. Beneath that you can find all important social media contact information of the shop.

**Photo Wall**

The OWASP Juice Shop also has an #/photo-wall page titled *Photo Wall* which allows its users to share their memories with other customers of the Juice Shop.

54

**Deluxe Membership**

The OWASP Juice Shop offers a deluxe membership to its customers which provides them with exclusive offers, free fast delivery and an unrestricted purchase of the items they like.

**Language selection**

From a dropdown menu in the navigation bar you can select a multitude of languages you want the user interface to be displayed in. Languages marked with a "flask"-icon next to them offer only rudimentary or partial translation.

55

*If you want to know more about (or even help with) the localization of OWASP Juice Shop, please refer to the Help with translation chapter in part III of this book.*

1

.

https://wiki.owasp.org/index.php/Map\_execution\_paths\_through\_applicatio n\_(OTG-INFO-007) ↩

2

. http://xunitpatterns.com/happy%20path.html ↩

56

**Customization**

One of the core usage scenarios for OWASP Juice Shop is in employee trainings in order to facilitating security awareness. With its not entirely serious user roster and product inventory the application might not be suited for all audiences alike.

In some particularly traditional domains or conservative enterprises it would be beneficial to have the demo application look and behave more like an internal application.

OWASP Juice Shop can be customized in its product inventory and look & feel to accommodate this requirement. It also allows to add an arbitrary number of fake users to make demonstrations - particularly those of UNION-SQL injection attacks -

even more impressive. Furthermore the Challenge solved!-notifications can be turned off in order to keep the impression of a "real" application undisturbed.

**How to customize the application**

The customization is powered by a YAML configuration file placed in /config . To run a customized OWASP Juice Shop you need to:

1. Place your own .yml configuration file into /config

2. Set the environment variable NODE\_ENV to the filename of your config without the .yml extension

On Windows: set NODE\_ENV=nameOfYourConfig

On Linux: export NODE\_ENV=nameOfYourConfig

3. Run npm start

You can also run a config directly in one command (on Linux) via NODE\_ENV=nameOfYourConfig npm start . By default the config/default.yml config is used which generates the original OWASP Juice Shop look & feel and inventory. Please note that it is not necessary to run npm install after switching customization configurations.

**Overriding default.yml in Docker container**

In order to override the default configuration inside your Docker container with one of the provided configs, you can pass in the NODE\_ENV environment variable with the -e parameter:

docker run -d -e "NODE\_ENV=bodgeit" -p 3000:3000

In order to inject your own configuration, you can use -v to mount the default.yml path inside the container to any config file on your outside file system:

docker run -d -e "NODE\_ENV=myConfig" -v /tmp/myConfig.yml:/juice-shop/config/myConfig.

57

**Overriding default.yml in Vagrant box**

Currently it is not possible to override the default configuration in the Vagrant box. It is not set up in a way where it could pass the NODE\_ENV environment variable to the Docker container that is spun up inside the box.

**YAML configuration file**

The YAML format for customizations is very straightforward. Below you find its schema along with an excerpt of the default settings.

**server section**

Offers technical configuration options for the web server hosting the application.

| **Property** | **Description** | **Default** |
| --- | --- | --- |
| port | Port to launch the server on. Would be  overwritten by PORT environment variable. | 3000 |
| basePath | When proxied in a subdirectory, this base path is used during redirects. Would be overwritten by BASE\_PATH environment variable. | '' |

**application section**

Defines customization options for texts, colors, images, URLs etc. within the application.

58

| **Property** | **Description** |
| --- | --- |
| domain | Domain used for all user email  addresses. |
| name | Name as shown in title and menu bar. |
| logo | Filename in  frontend/dist/frontend/assets/public/images *or* a URL of an image which will first be download to that folder and then used as a logo. |
| favicon | Filename in  frontend/dist/frontend/assets/public *or* a URL of an image in .ico format which will first be download to that folder and then used as a favicon. |
| theme | Name of the color theme used to render the UI. Options are bluegrey-lightgreen , blue-lightblue , deeppurple-amber , indigo pink , pink-bluegrey , purple-green and deeporange-indigo . See Material Color Themes for a sample screenshot of each theme. |
| showVersionNumber | Shows or hides the software version from the title. |
| showGitHubLinks | Shows or hides the *"GitHub"* button in the navigation and side bar as well as the info box about contributing on the *Score Board*. |
| localBackupEnabled | Enabled or disables the local backup feature for hacking progress and  filters/settings on the *Score Board*. |
| numberOfRandomFakeUsers | Represents the number of random user accounts to be created on top of the pre defined ones (which are required for several challenges). |
| altcoinName | Defines the name of the (fake) crypto currency that is offered on the *Token Sale* screen. |
| privacyContactEmail | The email address shown as contact in the *Privacy Policy*. |
| customMetricsPrefix | Prefix for all custom Prometheus metrics. Must be a lowercase letter single world by Prometheus conventions. |
| chatbot subsection | Specifies all characteristics of the bot answering user questions in the *Support Chat*. |
| social subsection | Specifies all social links embedded on various screens such as *About Us* or the *Photo Wall*. |
| recyclePage  subsection | Defines custom elements on the *Request Recycling Box* page. |

**Defa** 'jui

'OWA

Juic

favi

blue

light

true true

true

0 ,

add are

Juic

dono juice

juic

59

| **Property** | **Description** |
| --- | --- |
| welcomeBanner  subsection | Defines a dismissable welcome banner that can be shown when first visiting the application. |
| cookieConsent  subsection | Defines the cookie consent dialog shown in the bottom right corner. |
| securityTxt  subsection | Defines the attributes for the  security.txt file based on  thehttps://securitytxt.org/> Internet draft. |
| promotion subsection | Defines the attributes required for the /promotion screen where a marketing video with subtitles is rendered that hosts the XSS Tier 6 challenge. |
| easterEggPlanet  subsection | Defines the customizations for the 3D rendered planet easter egg. |
| googleOauth  subsection | Defines the client identifier and allowed redirect URIs for Google  OAuthintegration. |

**chatbot subsection**

**Defa**

Specifies all characteristics of the bot answering user questions in the *Support Chat*.

| **Property** | **Description** |
| --- | --- |
| name | Name the chat bot introduces itself with. |
| greeting | Initial greeting the chat bot uses when chatting with a user. |
| trainingData | Filename in data/chatbot *or* a URL of a JSON file which will first be download to that folder and then used as training data for the chat bot. |
| defaultResponse | Default response the chat bot uses when it could not understand the user's actual question. |
| avatar | Filename in  frontend/dist/frontend/assets/public/images *or* a URL of an image which will first be download to that folder and then used as a chat bot avatar. |

**social subsection**

**Default**

Juicy

"Nice to mee name>, I'm <

'botDefaultT

"Sorry I cou what you wer

'JuicyChatBo

Specifies all social links embedded on various screens such as *About Us* or the *Photo Wall*.

60

| **Property** | **Description** |
| --- | --- |
| twitterUrl | URL used as  the Twitter  link  promising  coupon  codes on the  *About Us* and *Your Basket*  screen. |
| facebookUrl | URL used as  the Facebook  link  promising  coupon  codes on the  *About Us* and *Your Basket*  screen. |
| slackUrl | URL used as  the Slack link  on the *About*  *Us* screen. |
| pressKitUrl | URL used as  the link to  logos and  media files  on the *About*  *Us* screen. |
| questionnaireUrl | URL used as  the link to a  user  questionnaire  on the *Score*  *Board*  screen. |

**recyclePage subsection**

**Default**

'https://twitter.com/owasp\_juiceshop'

'https://www.facebook.com/owasp.juiceshop

'http://owaspslack.com'

'https://github.com/OWASP/owasp

swag/tree/master/projects/juice-shop' ~

Defines custom elements on the *Request Recycling Box* page.

| **Property** | **Description** |
| --- | --- |
| topProductImage | Filename in  frontend/dist/frontend/assets/public/images/products to use as the image on the top of the info column on the page. |
| bottomProductImage | Filename in  frontend/dist/frontend/assets/public/images/products to use as the image on the bottom of the info column on the page. |

**welcomeBanner subsection**

Defines a dismissable welcome banner that can be shown when first visiting the application.

61

| **Property** | **Description** | **Default** |
| --- | --- | --- |
| showOnFirstStart | Shows or  hides the  banner. | true |
| title | Defines the  headline of  the banner. | Welcome to OWASP Juice Shop! |
| message | Defines the  body of the  banner. Can  contain  arbitrary  HTML. | <p>Being a web application with a vast number of intended security  vulnerabilities, the <strong>OWASP Juice Shop</strong> is supposed to be the opposite of a best practice or template application for web  developers: It is an awareness,  training, demonstration and exercise tool for security risks in modern web applications. The <strong>OWASP Juice Shop</strong> is an open-source project hosted by the non-profit <a  href='https://owasp.org'  target='\_blank'>Open Web Application Security Project (OWASP)</a> and is developed and maintained by volunteers. Check out the link below for more  information and documentation on the project.</p><h1><a href='https://owasp juice.shop'  target='\_blank'>https://owasp  juice.shop</a></h1> |

**cookieConsent subsection**

Defines the cookie consent dialog shown in the bottom right corner.

62

| **Property** | **Description** | **Default** |
| --- | --- | --- |
| backgroundColor | Color of the cookie  banner itself. | '#546e7a' |
| textColor | Color of the  message shown in  the cookie banner. | '#ffffff' |
| buttonColor | Defines the color of the button to  dismiss the banner. | '#558b2f' |
| buttonTextColor | Color of the  dismissText on the  button. | '#ffffff' |
| message | Explains the cookie usage in the  application. | 'This website uses fruit  cookies to ensure you get the juiciest tracking experience.' |
| dismissText | The text shown on  the button to  dismiss the banner. | 'Me want it!' |
| linkText | Caption of the link  that is shown after  the message to refer to further  information. | 'But me wait!' |
| linkUrl | URL that provides  further information  about cookie usage. | 'https://www.youtube.com/watch? v=9PnbKL3wuH4' |

**securityTxt subsection**

Defines the attributes for the security.txt file based on the https://securitytxt.org/ Internet draft.

| **Property** | **Description** |
| --- | --- |
| contact | An email  address, phone number or URL  to report  security  vulnerabilities  to. Can be fake  obviously. |
| encryption | URL to a public  encryption key  for secure  communication. Can be fake  obviously. |
| acknowledgements | URL a "hall of  fame" page.  Can be fake  obviously. |

**Default**

mailto:donotreply@owasp-juice.shop

https://keybase.io/bkimminich/pgp\_keys fingerprint=19c01cb7157e4645e9e2c863062

/#/score-board

63

**promotion subsection**

Defines the attributes required for the /promotion screen where a marketing video with subtitles is rendered that hosts the XSS Tier 6 challenge.

| **Property** | **Description** |
| --- | --- |
| video | Name of a file with video/mp4 content type in  frontend/dist/frontend/assets/public/videos *or* URL of an image to download to that folder and then use as the promotion video. |
| subtitles | Name of a Web Video Text Tracks Format file in  frontend/dist/frontend/assets/public/videos *or* URL of an image to download to that folder and then use as the promotion video. |

**easterEggPlanet subsection**

**Default**

JuiceShopJingle.mp JuiceShopJingle.vt

Defines the customizations for the 3D-rendered planet easter egg.

| **Property** | **Description** | **Default** |
| --- | --- | --- |
| name | Name of the 3D planet "easter egg" as shown in the page title. | Orangeuze |
| overlayMap | Filename in  frontend/dist/frontend/assets/private *or* URL of an image to download to that folder and then use as an overlay texture for the 3D planet "easter egg". | orangemap2k.jpg |

**googleOauth subsection**

Defines the client identifier and allowed redirect URIs for Google OAuth integration.

64

| **Property** | **Description** |
| --- | --- |
| clientId | Client  identifier of  the Google  Cloud  Platform  application  to handle  OAuth 2.0  requests  from  OWASP  Juice Shop. |
| authorizedRedirects  sub-sequence | Sub-list for  the redirect  URIs  authorized  for Google  OAuth. |

**authorizedRedirects sub-sequence**

**Default**

'1005568560502-

6hm16lef8oh46hr2d98vf2ohlnj4nfhq.apps.go

- { uri: 'https://demo.owasp-juice.shop 'http://demo.owasp-juice.shop' } - { uri shop.herokuapp.com' } - { uri: 'http://j shop.herokuapp.com' } - { uri: 'https:// juice.shop' } - { uri: 'http://preview.o { uri: 'https://juice-shop-staging.herok 'http://juice-shop-staging.herokuapp.com 'http://juice-shop.wtf' } - { uri: 'http proxy: 'http://local3000.owasp-juice.sho 'http://127.0.0.1:3000', proxy: 'http:// juice.shop' } - { uri: 'http://localhost 'http://local4200.owasp-juice.shop' } - 'http://127.0.0.1:4200', proxy: 'http:// juice.shop' } - { uri: 'http://192.168.9 'http://localmac.owasp-juice.shop' } - { 'http://penguin.termina.linux.test:3000' 'http://localchromeos.owasp-juice.shop'

Defines the allowed redirect URIs and their optional proxy for Google OAuth integration.

| **Property** | **Description** | **Conditions** | **Default** |
| --- | --- | --- | --- |
| uri | URI authorized on Google  Cloud Platform the Juice Shop is expected to be running on. | mandatory |  |
| proxy | Proxy URI authorized on  Google Cloud Platform that will itself redirect back to the  original uri . Necessary for  addresses not allowed as  Google OAuth redirect targets, such as localhost or IP  addresses. | optional | null |

**challenges section**

Defines configuration options for the hacking challenges within the Juice Shop.

65

| **Property** | **Description** |
| --- | --- |
| showSolvedNotifications | Shows or hides all instant  *"challenge solved"*-  notifications. Recommended to set to false for awareness demos. |
| showHints | Shows or hides hints for each challenge on hovering  over/clicking its *"unsolved"*  badge on the score board. |
| restrictToTutorialsFirst | Disables all *Score Board* filter options and hides those of the 100 challenges without a  tutorial until all challenges with a tutorial have been solved. |
| overwriteUrlForProductTamperingChallenge | URL that should replace the original URL defined in  urlForProductTamperingChallenge for the Product Tampering  challenge. |
| xssBonusPayload | This XSS payload is expected during the Bonus Payload  challenge. |
| safetyOverride | Enables all potentially  dangerous challenges  regardless of any harm they might cause when running in a containerized environment. ☠ **Use at your own risk!** |

**hackingInstructor section**

Allows to enable and customize the Hacking Instructor tutorial mode.

| **Property** | **Description** |
| --- | --- |
| isEnabled | Shows or hides the Hacking Instructor links from the *Score Board* and *Welcome Banner*. |
| avatarImage | Filename in  frontend/dist/frontend/assets/public/images *or* a URL of an image which will first be download to that folder and then used as an avatar in the tutorial speech bubbles. |

**products sequence**

**Default**

true

juicyBot.png

List of product mappings which, when specified, replaces **the entire list** of default products.

66

| **Property** |
| --- |
| name |
| description |
| price |
| deluxePrice |
| quantity |
| limitPerUser |
| image |
| deletedDate |
| urlForProductTamperingChallenge |
| useForChristmasSpecialChallenge |
| fileForRetrieveBlueprintChallenge |
| keywordsForPastebinDataLeakChallenge |
| reviews sub-sequence |

**reviews sub-sequence**

**Description**

Name of the product.

Description of the product.

Price of the product.

Price of the product for *Deluxe Membe* customers.

Available quantity of product in stock.

Maximum purchase limit for regular cus Does not apply to *Deluxe Membership*

Filename in

frontend/dist/frontend/assets/public/imag *or* URL of an image to download to tha then use as a product image.

Deletion date of the product in YYYY-MM

Sets the original link of the product whi target for the Product Tampering challe Overrides deletedDate with null .

Marks a product as the target for the "c special" challenge. Overrides deletedDa 2014-12-27 .

Filename in

frontend/dist/frontend/assets/public/imag *or* URL of a file download to that folder use as the target for the Retrieve Bluep challenge. If a filename is specified but does not exist in

frontend/dist/frontend/assets/public/imag the challenge is still solvable by just req from the server. ℹ *To make this challen realistically solvable, include some kind the blueprint file's name/type in the pro (e.g. its Exif metadata) or in the produ description.*

List of keywords which are all mandato mention in a feedback or complaint to s Leaked Unsafe Product challenge. Ove

deletedDate with 2019-02-1 . ℹ *To make challenge realistically solvable, provide keywords on e.g. PasteBin in an obscu that works well with the "dangerous ing an unsafe product" narrative.*

Sub-list which adds reviews to a produc

67

Sub-list which adds reviews to a product.

| **Property** | **Description** | **Conditions** |
| --- | --- | --- |
| text | Text of the review. | mandatory |
| author | Reference by key from  data/static/users.yml to the author of the review | mandatory |

**memories sequence**

List which, when specified, replaces all default *Photo Wall* entries except a hard coded one needed to solve the Retrieve the photo of Bjoern's cat in "melee combat-mode" challenge.

| **Property** |
| --- |
| image |
| caption |
| user |
| geoStalkingMetaSecurityQuestion |
| geoStalkingMetaSecurityAnswer |
| geoStalkingVisualSecurityQuestion |
| geoStalkingVisualSecurityAnswer |

**ctf section**

**Description**

Filename in

frontend/dist/frontend/assets/public/images/ *or* URL of an image to download to that fo then use as a *Photo Wall* image.

Text to show when hovering over the imag sending a Tweet about it.

Reference by key from data/static/users. the owner of the photo upload.

ID of the security question associated with Meta Geo Stalking challenge.

Answer to the security question associate the Meta Geo Stalking challenge. Should retrievable via the meta data of the assosi image.

ID of the security question associated with Visual Geo Stalking challenge.

Answer to the security question associate the Visual Geo Stalking challenge. Should retrievable via some (not too obvious) visu in the assosicated image.

Section to enable and configure the Capture-the-Flag mode built into OWASP Juice Shop.

68

| **Property** |
| --- |
| showFlagsInNotifications |
| showCountryDetailsInNotifications |
| countryMapping sub-mapping |

**Description**

Shows or hides the CTF flag codes in the *"challenge solved"*-notifications. Is ignored when

application.showChallengeSolvedNotifications is set to false .

Determines if the country (from

countryMapping ) mapped to the solved challenge is displayed in the notification. Can be none , name , flag or both . Only useful for CTFs using FBCTF.

List of mappings which associates challenges to countries on the challenge map of FBCTF. Only needed for CTFs using FBCTF.

**countryMapping sub-mapping**

List of mappings which associates challenges to countries on the challenge map of FBCTF. Only needed for CTFs using FBCTF:

Challenge key from data/static/challenges.yml

name the name of the country

code the two-letter ISO code of the country

ℹ *When specifying countryMapping , it is mandatory to map* ***all challenges*** *in order to produce a valid configuration file. It is recommended to use config/fbctf.yml as a template for that purpose.*

**Configuration example**

69

server:

port: 3000

application:

domain: juice-sh.op

name: 'OWASP Juice Shop'

logo: JuiceShop\_Logo.png

favicon: favicon\_js.ico

theme: bluegrey-lightgreen

showVersionNumber: true

showGitHubLinks: true

localBackupEnabled: true

numberOfRandomFakeUsers: 0

altcoinName: Juicycoin

privacyContactEmail: donotreply@owasp-juice.shop

customMetricsPrefix: juiceshop

chatBot:

name: 'Juicy'

greeting: "Nice to meet you <customer-name>, I'm <bot-name>"

trainingData: 'botDefaultTrainingData.json'

defaultResponse: "Sorry I couldn't understand what you were trying to say" avatar: 'JuicyChatBot.png'

social:

twitterUrl: 'https://twitter.com/owasp\_juiceshop'

facebookUrl: 'https://www.facebook.com/owasp.juiceshop'

slackUrl: 'http://owaspslack.com'

redditUrl: 'https://www.reddit.com/r/owasp\_juiceshop'

pressKitUrl: 'https://github.com/OWASP/owasp-swag/tree/master/projects/juice-shop' questionnaireUrl: ~

recyclePage:

topProductImage: fruit\_press.jpg

bottomProductImage: apple\_pressings.jpg

welcomeBanner:

showOnFirstStart: true

title: 'Welcome to OWASP Juice Shop!'

message: "<p>Being a web application with a vast number of intended security vulne cookieConsent:

backgroundColor: '#546e7a'

textColor: '#ffffff'

buttonColor: '#558b2f'

buttonTextColor: '#ffffff'

message: 'This website uses fruit cookies to ensure you get the juiciest tracking dismissText: 'Me want it!'

linkText: 'But me wait!'

linkUrl: 'https://www.youtube.com/watch?v=9PnbKL3wuH4'

securityTxt:

contact: 'mailto:donotreply@owasp-juice.shop'

encryption: 'https://keybase.io/bkimminich/pgp\_keys.asc?fingerprint=19c01cb7157e46 acknowledgements: '/#/score-board'

promotion:

video: JuiceShopJingle.mp4

subtitles: JuiceShopJingle.vtt

easterEggPlanet:

name: Orangeuze

overlayMap: orangemap2k.jpg

googleOauth:

clientId: '1005568560502-6hm16lef8oh46hr2d98vf2ohlnj4nfhq.apps.googleusercontent.c authorizedRedirects:

- { uri: 'https://demo.owasp-juice.shop' }

- { uri: 'http://demo.owasp-juice.shop' }

- { uri: 'https://juice-shop.herokuapp.com' }

- { uri: 'http://juice-shop.herokuapp.com' }

- { uri: 'https://preview.owasp-juice.shop' }

- { uri: 'http://preview.owasp-juice.shop' }

- { uri: 'https://juice-shop-staging.herokuapp.com' }

- { uri: 'http://juice-shop-staging.herokuapp.com' }

70

- { uri: 'http://juice-shop.wtf' }

- { uri: 'http://localhost:3000', proxy: 'http://local3000.owasp-juice.shop' } - { uri: 'http://127.0.0.1:3000', proxy: 'http://local3000.owasp-juice.shop' } - { uri: 'http://localhost:4200', proxy: 'http://local4200.owasp-juice.shop' } - { uri: 'http://127.0.0.1:4200', proxy: 'http://local4200.owasp-juice.shop' } - { uri: 'http://192.168.99.100:3000', proxy: 'http://localmac.owasp-juice.shop'

- { uri: 'http://penguin.termina.linux.test:3000', proxy: 'http://localchromeos. challenges:

showSolvedNotifications: true

showHints: true

restrictToTutorialsFirst: false

safetyOverride: false

overwriteUrlForProductTamperingChallenge: 'https://owasp.slack.com' xssBonusPayload: '<iframe width="100%" height="166" scrolling="no" frameborder="no" hackingInstructor:

isEnabled: true

avatarImage: juicyBot.png

products:

-

name: 'Apple Juice (1000ml)'

price: 1.99

description: 'The all-time classic.'

image: apple\_juice.jpg

reviews:

- { text: 'One of my favorites!', author: admin }

# ~~~~~ ... ~~~~~~

-

name: 'OWASP SSL Advanced Forensic Tool (O-Saft)'

description: 'O-Saft is an easy to use tool to show information about SSL certific price: 0.01

image: orange\_juice.jpg

urlForProductTamperingChallenge: 'https://www.owasp.org/index.php/O-Saft' -

name: 'Christmas Super-Surprise-Box (2014 Edition)'

description: 'Contains a random selection of 10 bottles (each 500ml) of our tastie price: 29.99

image: undefined.jpg

useForChristmasSpecialChallenge: true

-

name: 'OWASP Juice Shop Sticker (2015/2016 design)'

description: 'Die-cut sticker with the official 2015/2016 logo. By now this is a r price: 999.99

image: sticker.png

deletedDate: '2017-04-28'

# ~~~~~ ... ~~~~~~

-

name: 'OWASP Juice Shop Logo (3D-printed)'

description: 'This rare item was designed and handcrafted in Sweden. This is why i price: 99.99

image: 3d\_keychain.jpg

fileForRetrieveBlueprintChallenge: JuiceShop.stl

# ~~~~~ ... ~~~~~~

memories:

-

image: 'magn(et)ificent!-1571814229653.jpg'

caption: 'Magn(et)ificent!'

user: bjoernGoogle

-

image: 'my-rare-collectors-item!-[̲̅$̲̅(̲̅-͡°-͜ʖ-͡°̲̅)̲̅$̲̅]-1572603645543.jpg' caption: 'My rare collectors item! [̲̅$̲̅(̲̅ ͡° ͜ʖ ͡°̲̅)̲̅$̲̅]'

user: bjoernGoogle

ctf:

showFlagsInNotifications: false

showCountryDetailsInNotifications: none

countryMapping: ~

71

**Overriding default settings**

When creating your own YAML configuration file, you can rely on the existing default values and only overwrite what you want to change. The provided config/ctf.yml file for capture-the-flag events for example is as short as this:

application:

logo: JuiceShopCTF\_Logo.png

favicon: favicon\_ctf.ico

showVersionNumber: false

showGitHubLinks: false

welcomeBanner:

showOnFirstStart: false

challenges:

showHints: false

safetyOverride: true

hackingInstructor:

isEnabled: false

ctf:

showFlagsInNotifications: true

**Testing customizations**

You can validate your custom configuration file against the schema by running npm run lint:config -- -f /path/to/myConfig.yml . This validation automatically happens on server startup as well.

To verify if your custom configuration will not break any of the challenges, you should run the end-to-end tests via npm run protractor . If they pass, all challenges will be working fine!

**Material Color Themes**

The application.theme property allows certain pre-defined color schemes. The table below shows sample screenshots for each of these.

72

| **Theme** | **Sample screenshot** |
| --- | --- |
| bluegrey  lightgreen |  |
| blue  lightblue |  |
| deeppurple  amber |  |
| indigo  pink |  |
| pink  bluegrey |  |

73

| **Theme** | **Sample screenshot** |
| --- | --- |
| purple  green |  |
| deeporange  indigo |  |

**Provided customizations**

The following fully re-themed customizations are provided out of the box by OWASP Juice Shop for demonstration purposes:

7 Minute Security: Full conversion https://7ms.us-theme for the first podcast that picked up the Juice Shop way before it was famous! ��

Mozilla-CTF: Another full conversion theme harvested and refined from the Mozilla Austin CTF-event!

AllDayDeflOps: This full conversion had its live debut at the All Day DevOps 2019 conference and was released the same day! ��

The BodgeIt Store: An homage to our server-side rendered ancestor. May it rest in JSPs! ��

OWASP Juice Box: If you find *jo͞osbäks* much easier to pronounce than *jo͞osSHäp*, this customization is for you.

Furthermore these convenience customizations are provided out-of-the-box to simplify usage of OWASP Juice Shop in specific use cases and situations:

CTF-mode: Keeps the Juice Shop in its default layout but disabled hints while enabling CTF flag codes in the *"challenge solved"*-notifications. Refer to Hosting a CTF event to learn more about running a CTF-event with OWASP Juice Shop. ��

Quiet mode: Keeps the Juice Shop in its default layout but hides all *"challenge solved"*-notifications, GitHub ribbon and challenge hints. �� Tutorial mode: Restricts the user to first solve all challenges with Hacking Instructor tutorials before the entire *Score Board* gets unlocked and filterable. �� Hidden challenges can still be solved and users will receive corresponding success notifications!

74

Unsafe mode: Keeps everything at default settings except *enabling* all potentially dangerous challenges even in containerized environments. ☠ **Use at your own risk!**

**Limitations**

When running a customization (except default.yml ) that overwrites the property application.domain , the description of the challenges *Ephemeral Accountant*, *Forged Signed JWT* and *Unsigned JWT* will always be shown in English.

Configurations (except default.yml ) do not support translation of custom product names and descriptions as of v12.0.2.

Several Hacking Instructor scripts depend on product inventory and product reviews that might not exist in the required form when you overwrote the default products list. Consider turning off the tutorials by setting hackingInstructor.isEnabled to false in that case.

**Additional Browser tweaks**

Consider you are doing a live demo with a highly customized corporate theme. Your narrative is, that this *really* is an upcoming eCommerce application *of that company*. Walking the "happy path" might now lure you into two situations which could spoil the immersion for the audience.

75

**Coupon codes on social media**

If you configured the twitterUrl / facebookUrl as the company's own account/page, you will most likely not find any coupon codes posted there. You will probably fail to convince the social media team to tweet or retweet some coupon code for an application that does not even exist!

**OAuth Login**

Another immersion spoiler occurs when demonstrating the *Log in with Google* functionality, which will show you the application name registered on Google Cloud Platform: *OWASP Juice Shop*! There is no way to convince Google to show anything else for obvious trust and integrity reasons.

76

ℹ *Since v10.0.0 you can overwrite the googleOauth subsection to use your own application on Google Cloud Platform for handling OAuth. This is a relatively high effort, so maybe you want to kill two birds with one stone instead as described in the next section.*

**On-the-fly text replacement**

You can solve both of the above problems *in your own Browser* by replacing text on the fly when the Twitter, Facebook or Google-Login page is loaded. For Chrome Word Replacer II is a plugin that does this work for you with very little setup effort. For Firefox FoxReplace does a similar job. After installing either plugin you have to create two text replacements:

1. Create a replacement for OWASP Juice Shop (as it appears on Google-Login) with your own application name. Best use application.name from your configuration.

2. Create another replacement for a complete or partial Tweet or Facebook post with some marketing text and an actual coupon code. You can get valid coupon codes from the OWASP Juice Shop Twitter feed:

https://twitter.com/owasp\_juiceshop.

77

3. Enable the plugin and verify your replacements work:

78

79