

3.3 CORS (Cross-Origin Resource Sharing)

クロスオリジンであっても、シンプルなWebリクエストであれば、情報アクセスが可能である。**シンプルなリクエストの定義**とは

HTTPメソッドは

- GET
- HEAD
- POST

XMLHttpRequestオブジェクトのsetRequestHeaderメソッドで設定するリクエストヘッダは

- Accept
- Accept-Language
- Content-Language
- Content-Type

Content-Type は次のいずれか

- application/x-www-form-urlencoded
- multipart/form-data
- text/plain

上記の条件以外のクロスオリジンのWebリクエストを行う前に、そのリクエストが受け入れられるかどうかを事前にチェックするのが、**プリフライトリクエスト**です。CORS仕様では、Webブラウザを使用して情報をやり取りすることができる定義として、サーバーが新たなHTTPヘッダーを追加します。さらに、やり取りの結果、副作用を引き起こす可能性があるHTTPリクエストメソッド（特にGET以外のHTTPメソッドや、特定のMIMEタイプを伴うPOSTの使い方）のために、WebブラウザがOPTIONSメソッドを用いて、「プリフライト」（サーバーから対応するメソッドの一覧を収集すること）を行います。

プリフライトリクエストでやり取りするヘッダ

| 要求の種類 | リクエスト | 例 |
|------------|--------------------------------|--|
| メソッドに対する許可 | Access-Control-Request-Method | Access-Control-Request-Method: POST |
| ヘッダに対する許可 | Access-Control-Request-Headers | Access-Control-Request-Headers: accept, origin, content-type |
| オリジンに対する許可 | Origin | Origin: http://example.jp |

| 要求の種類 | レスポンス | 例 |
|------------|------------------------------|--|
| メソッドに対する許可 | Access-Control-Allow-Methods | Access-Control-Allow-Methods: GET,POST,OPTIONS |
| ヘッダに対する許可 | Access-Control-Allow-Headers | Access-Control-Allow-Headers: Content-Type |
| オリジンに対する許可 | Access-Control-Allow-Origin | Access-Control-Allow-Origin: http://example.jp |

認証情報を含むやり取りでは

① 認証情報を含むリクエストの許可を設定する（サーバ側）

・Access-Control-Allow-Credentials ヘッダを true に指定します。

② 信頼するオリジンに対して、認証情報を含むリクエストを送る場合（クライアント側）

・XMLHttpRequest クラスの、withCredentials プロパティを true に指定します。

33-001 : シンプルなリクエスト

【ブラウザ】

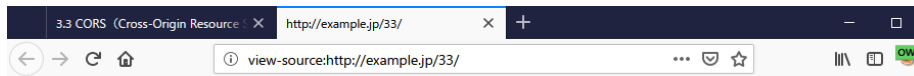


3.3 CORS (Cross-Origin Resource Sharing)

1. [33-001 : シンプルなリクエスト](#)
2. [33-001a: シンプルなリクエスト\(CORS対応\)](#)
3. [33-003 : プリフライトリクエストなし](#)
4. [33-003a: プリフライトリクエストあり](#)
5. [33-003a: プリフライトリクエスト \(完成\)](#)
6. [33-005 : アクセスカウンタ \(withCredentialsなし\)](#)
7. [33-005a: アクセスカウンタ \(withCredentialsあり\)](#)
8. [33-005b: アクセスカウンタ \(Access-Control-Allow-Credentialsあり\)](#)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="33-001a.html">33-001a: シンプルなリクエスト(CORS対応)</a></li>
8 <li><a href="33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="33-003a.html">33-003a: プリフライトリクエストあり</a></li>
10 <li><a href="33-003b.html">33-003a: プリフライトリクエスト (完成) </a></li>
11 <li><a href="33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="33-005a.html">33-005a: アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="33-005b.html">33-005b: アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="phpinfo.php">phpinfo</a><br>
16 <a href="#">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-001.html】

```
/var/www/html/33/33-001.html - wasbook@192.168.56.101 - エディタ - WinSCP
/var/www/html/33/33-001.html
<body>
<script>
var req = new XMLHttpRequest();
req.open('GET', 'http://api.example.net/33/33-002.php');
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    alert(req.responseText);
  }
};
req.send(null);
</script>
送信しました
</body>
```

【サーバ: 33/33-002.php】

```
/var/www/html/33/33-002.php - wasbook@192.168.56.101 - エディタ - WinSCP
/var/www/html/33/33-002.php
'Content-Type: application/json' は
シンプルなWebリクエストの条件に合わない
<?php
header('Content-Type: application/json');
echo json_encode(['zipcode' => '100-0100', 'address' => '東京都大島町']);

郵便番号、住所をJSON形式で返すAPI
```

【ブラウザ→サーバ: リクエスト 33/33-001.html → レスポンス】

無題セッション - 20181215-183532 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

クイック スタート リクエスト レスポンス

コンテキスト

- 既定コンテキスト
- サイト

デフォルトビュー

```
GET http://example.jp/33/33-001.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 12:46:36 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 304
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "130-56c2a2decde9b-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
var req = new XMLHttpRequest();
req.open("GET", 'http://api.example.net/33/33-002.php');
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
alert(req.responseText);
}
};
req.send(null);
</script>
送信しました
</body>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|----------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 25 | 18/12/15 21:46:34 | GET | http://example.jp/33/33-001.html | 200 | OK | 4 ms | 304 bytes | Medium | | Script |

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0

【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-002.php → レスポンス】

The screenshot shows the OWASP ZAP interface. On the left, the 'Contexts' pane shows the selected site context. The 'Request' pane displays the following details:

- GET http://api.example.net/33/33-002.php HTTP/1.1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
- Accept: */*
- Accept-Language: ja,en-US;q=0.7,en;q=0.3
- Accept-Encoding: gzip, deflate
- Referer: http://example.jp/33/33-001.html
- Origin: http://example.jp
- DNT: 1
- Connection: keep-alive
- Host: api.example.net

The 'Response' pane shows the following details:

- HTTP/1.1 200 OK
- Server: nginx/1.10.3
- Date: Sat, 15 Dec 2018 12:46:36 GMT
- Content-Type: application/json
- Content-Length: 71
- Connection: keep-alive
- X-UA-Compatible: IE=edge

The response body contains a JSON object: `{ "zipcode": "100-0100", "address": "\u6771\u4eac\u90fd\u5927\u5cf6\u753a" }`. A green callout bubble points to this JSON data with the text: "JSON形式の郵便番号、住所がレスポンスされている".

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|--------------------------------------|----------|------------|-------------|-------------|--------|-----|------|
| 27 | 18/12/15 21:46:34 | GET | http://api.example.net/33/33-002.php | 200 | OK | 6 ms | 71 bytes | Low | | JSON |

【ブラウザ】

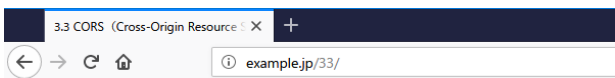
The screenshot shows a browser window with the address bar at `example.jp/33/33-001.html`. The page content says "送信しました". The developer console shows a yellow error message: "クロスオリジン要求をブロックしました: 同一生成元ポリシーにより、http://api.example.net/33/33-002.php にあるリモートリソースの読み込みは拒否されます (理由: CORS ヘッダー 'Access-Control-Allow-Origin' が足りない)。" A green callout bubble points to this error with the text: "サーバからレスポンスは送られているが、クロスオリジン要求をブロックしている。Access-Control-Allow-Origin が足りないので、JSONデータが表示されていない。".

The screenshot shows the browser's source code view for `http://example.jp/33/33-001.html`. The code is as follows:


```
1 <body>
2 <script>
3   var req = new XMLHttpRequest();
4   req.open('GET', 'http://api.example.net/33/33-002.php');
5   req.onreadystatechange = function() {
6     if (req.readyState == 4 && req.status == 200) {
7       alert(req.responseText);
8     }
9   };
10  req.send(null);
11 </script>
12 送信しました
13 </body>
14
```

33-001a: シンプルなリクエスト(CORS対応)

【ブラウザ】

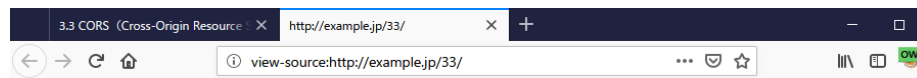


3.3 CORS (Cross-Origin Resource Sharing)

1. [33-001 : シンプルなリクエスト](#)
2. [33-001a: シンプルなリクエスト\(CORS対応\)](#) 
3. [33-003 : プリフライトリクエストなし](#)
4. [33-003a: プリフライトリクエストあり](#)
5. [33-003a: プリフライトリクエスト \(完成\)](#)
6. [33-005 : アクセスカウンタ \(withCredentialsなし\)](#)
7. [33-005a: アクセスカウンタ \(withCredentialsあり\)](#)
8. [33-005b: アクセスカウンタ \(Access-Control-Allow-Credentialsあり\)](#)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="/33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="/33-001a.html">33-001a: シンプルなリクエスト(CORS対応)</a></li>
8 <li><a href="/33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="/33-003a.html">33-003a: プリフライトリクエストあり</a></li>
10 <li><a href="/33-003a.html">33-003a: プリフライトリクエスト (完成) </a></li>
11 <li><a href="/33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="/33-005a.html">33-005a: アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="/33-005b.html">33-005b: アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="/phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-001a.html】

```
/var/www/html/33/33-001a.html - wasbook@192.168.56.101 - エディタ - WinSCP
k?body>
<script>
var req = new XMLHttpRequest();
req.open('GET', 'http://api.example.net/33/33-002a.php');
req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
        alert(req.responseText);
    }
};
req.send(null);
</script>
送信しました
</body>
```

【サーバ: 33/33-002a.html】

```
/var/www/html/33/33-002a.php - wasbook@192.168.56.101 - エディタ - WinSCP
k?php
header(['Content-Type: application/json']);
header(['Access-Control-Allow-Origin: http://example.jp']);
echo json_encode(['zipcode' => '100-0100', 'address' => '東京都大島町']);
```

クロスオリジンからの
読み出しを許可

【ブラウザ→サーバ: リクエスト 33/33-001a.html → レスポンス】

無題セッション - 20181215-183532 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

クイック スタート リクエスト レスポンス

コンテキスト

- 既定コンテキスト
- サイト

デフォルトビュー

```
GET http://example.jp/33/33-001a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 13:09:21 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 305
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "131-56c2a2decde9b-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
var req = new XMLHttpRequest();
req.open("GET", 'http://api.example.net/33/33-002a.php');
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
alert(req.responseText);
}
};
req.send(null);
</script>
送信しました
</body>
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|-----------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 31 | 18/12/15 22:09:19 | GET | http://example.jp/33/33-001a.html | 200 | OK | 6 ms | 305 bytes | Medium | | Script |

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0

【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-002a.html → レスポンス】

The screenshot shows the OWASP ZAP interface with the request and response details for a GET request to `http://api.example.net/33/33-002a.php`. The request headers include `Origin: http://example.jp` and `Access-Control-Allow-Origin: http://example.jp`. The response headers include `Access-Control-Allow-Origin: http://example.jp` and `Content-Type: application/json`. A callout box points to the `Access-Control-Allow-Origin` header in the response, stating: "クロスオリジンからの読み出しを許可" (Allow cross-origin loading).

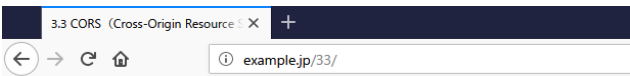
| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|------|
| 32 | 18/12/15 22:09:19 | GET | http://api.example.net/33/33-002a.php | 200 | OK | 6 ms | 71 bytes | Low | | JSON |

【ブラウザ】

The screenshot shows a browser window with the address `example.jp/33/33-001a.html`. A dialog box displays the JSON response: `{ "zipcode": "100-0100", "address": "\u6771\u4eac\u90fd\u5927\u5cf6\u753a" }`. A callout box points to the `Access-Control-Allow-Origin` header in the response, stating: "Access-Control-Allow-Origin: http://example.jp でクロスオリジンからの読み出しを許可しているのでJSONデータが表示できた。" (Since cross-origin loading is allowed by Access-Control-Allow-Origin: http://example.jp, the JSON data could be displayed).

33-003 :プリフライトリクエストなし

【ブラウザ】

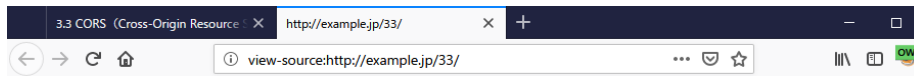


3.3 CORS (Cross-Origin Resource Sharing)

1. [33-001 :シンプルなリクエスト](#)
2. [33-001a:シンプルなリクエスト\(CORS対応\)](#)
3. [33-003 :プリフライトリクエストなし](#)
4. [33-003a:プリフライトリクエストあり](#)
5. [33-003a:プリフライトリクエスト \(完成\)](#)
6. [33-005 :アクセスカウンタ \(withCredentialsなし\)](#)
7. [33-005a:アクセスカウンタ \(withCredentialsあり\)](#)
8. [33-005b:アクセスカウンタ \(Access-Control-Allow-Credentialsあり\)](#)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="/33-001.html">33-001 :シンプルなリクエスト</a></li>
7 <li><a href="/33-001a.html">33-001a:シンプルなリクエスト (CORS対応)</a></li>
8 <li><a href="/33-003.html">33-003 :プリフライトリクエストなし</a></li>
9 <li><a href="/33-003a.html">33-003a:プリフライトリクエストあり</a></li>
10 <li><a href="/33-003a.html">33-003a:プリフライトリクエスト (完成)</a></li>
11 <li><a href="/33-005.html">33-005 :アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="/33-005a.html">33-005a:アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="/33-005b.html">33-005b:アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="/phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-003.html】

```
/var/www/html/33/33-003.html - wasbook@192.168.56.101 - エディタ - WinSCP
kbody>
<script>
var req = new XMLHttpRequest();
req.open('POST', 'http://api.example.net/33/33-004.php');
req.setRequestHeader('content-type', 'application/json');
data = '{"zipcode": "100-0100"}';
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    alert(req.responseText);
  }
};
req.send(data);
</script>
送信しました
</body>
```

【サーバ: 33/33-004.php】

```
/var/www/html/33/33-004.php - wasbook@192.168.56.101 - エディタ - WinSCP
k?php
header('Content-Type: application/json');
header('Access-Control-Allow-Origin: http://example.jp');
echo json_encode(['zipcode' => '100-0100', 'address' => '東京都大島町']);
```

Content-Type:
application/json

【ブラウザ→サーバ: リクエスト 33/33-003.html → レスポンス】

The screenshot displays the OWASP ZAP 2.7.0 interface. The top menu includes 'ファイル', '編集', '表示', 'ポリシー', 'レポート', 'ツール', 'オンライン', and 'ヘルプ'. The main window is split into three panes: 'コンテキスト' (Context) on the left, 'リクエスト' (Request) in the middle, and 'レスポンス' (Response) on the right. The 'リクエスト' pane shows the following details:

```
GET http://example.jp/33/33-003.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

The 'レスポンス' pane shows the following details:

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 14:08:58 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 402
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "192-56c2a2ded0d7b-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge
```

The response body contains the following JavaScript code:

```
<body>
<script>
var req = new XMLHttpRequest();
req.open("POST", "http://api.example.net/33/33-004.php");
req.setRequestHeader("content-type", "application/json");
data = {"zipcode": "100-0100"};
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    alert(req.responseText);
  }
};
req.send(data);
</script>
送信しました
</body>
```

At the bottom, a table lists the request details:

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|----------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 35 | 18/12/15 23:08:56 | GET | http://example.jp/33/33-003.html | 200 | OK | 6 ms | 402 bytes | Medium | | Script |

The status bar at the bottom indicates '現在のスキャン' (Current Scan) with various icons and counts.

【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-004.html → レスポンス】

無題セッション - 20181215-183532 - OWASP ZAP 2.7.0

標準モード

コンテキスト: 既定コンテキスト

リクエスト:

```

OPTIONS http://api.example.net/33/33-004.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: http://example.jp
DNT: 1
Connection: keep-alive
Host: api.example.net
    
```

レスポンス:

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 14:08:58 GMT
Content-Type: application/json
Content-Length: 71
Connection: keep-alive
Access-Control-Allow-Origin: http://example.jp
X-UA-Compatible: IE=edge

{"zipcode":"100-0100","address":"東京都千代田区千代田"}
    
```

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|--------------------------------------|----------|------------|-------------|-------------|--------|-----|------|
| 36 | 18/12/15 23:08:56 | OPTIONS | http://api.example.net/33/33-004.php | 200 | OK | 5 ms | 71 bytes | Low | | JSON |

【ブラウザ】

example.jp/33/33-003.html

example.jp/33/33-003.html

送信しました

出力を絞り込み

クロスオリジン要求をブロックしました: 同一生成元ポリシーにより、http://api.example.net/33/33-004.php にあるリモートリソースの読み込みは拒否されます (理由: CORS プリフライトチャンネルからの CORS ヘッダー 'Access-Control-Allow-Headers' にトークン 'content-type' が足りない)。 [詳細]

クロスオリジン要求をブロックしました: 同一生成元ポリシーにより、http://api.example.net/33/33-004.php にあるリモートリソースの読み込みは拒否されます (理由: CORS 要求が成功しなかった)。 [詳細]

クロスオリジン要求をブロックしました:
同一生成元ポリシーにより、リモートリソースの読み込みは拒否されます
(理由: CORS プリフライトチャンネルからの CORS ヘッダー 'Access-Control-Allow-Headers' にトークン 'content-type' が足りない)。

example.jp/33/33-003.html

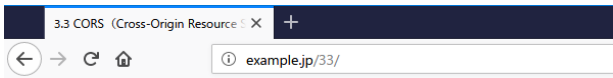
view-source:http://example.jp/33/33-003.html

```

1 <body>
2 <script>
3   var req = new XMLHttpRequest();
4   req.open("POST", "http://api.example.net/33/33-004.php");
5   req.setRequestHeader("content-type", "application/json");
6   data = { "zipcode": "100-0100" };
7   req.onreadystatechange = function() {
8     if (req.readyState == 4 && req.status == 200) {
9       alert(req.responseText);
10    }
11  };
12  req.send(data);
13 </script>
14 送信しました
15 </body>
16
    
```

33-003a:プリフライトリクエストあり

【ブラウザ】

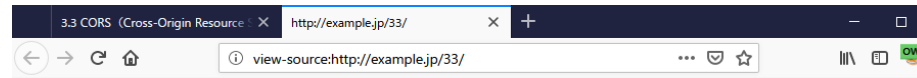


3.3 CORS (Cross-Origin Resource Sharing)

1. [33-001 : シンプルなリクエスト](#)
2. [33-001a: シンプルなリクエスト\(CORS対応\)](#)
3. [33-003 : プリフライトリクエストなし](#)
4. [33-003a: プリフライトリクエストあり](#)
5. [33-003a: プリフライトリクエスト \(完成\)](#)
6. [33-005 : アクセスカウンタ \(withCredentialsなし\)](#)
7. [33-005a: アクセスカウンタ \(withCredentialsあり\)](#)
8. [33-005b: アクセスカウンタ \(Access-Control-Allow-Credentialsあり\)](#)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="/33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="/33-001a.html">33-001a: シンプルなリクエスト(CORS対応)</a></li>
8 <li><a href="/33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="/33-003a.html">33-003a: プリフライトリクエストあり</a></li>
10 <li><a href="/33-003b.html">33-003a: プリフライトリクエスト (完成) </a></li>
11 <li><a href="/33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="/33-005a.html">33-005a: アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="/33-005b.html">33-005b: アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="/phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-003a.html】

```
/var/www/html/33/33-003a.html - wasbook@192.168.56.101 - エディタ - WinSCP
k?body>
<script>
var req = new XMLHttpRequest();
req.open('POST', 'http://api.example.net/33/33-004a.php');
req.setRequestHeader('content-type', 'application/json');
data = '{"zipcode": "100-0100"}';
req.onreadystatechange = function() {
    if (req.readyState == 4 && req.status == 200) {
        alert(req.responseText);
    }
};
req.send(data);
</script>
送信しました
</body>
```

【サーバ: 33/33-004a.html】

```
/var/www/html/33/33-004a.php - wasbook@192.168.56.101 - エディタ - WinSCP
k?php
if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {
    if ($_SERVER['HTTP_ORIGIN'] === 'http://example.jp') {
        header('Access-Control-Allow-Origin: http://example.jp');
        header('Access-Control-Allow-Methods: POST, GET, OPTIONS');
        header('Access-Control-Allow-Headers: Content-Type');
        header('Access-Control-Max-Age: 1728000');
        header('Content-Length: 0');
        header('Content-Type: text/plain');
    } else {
        header('HTTP/1.1 403 Access Forbidden');
        header('Content-Type: text/plain');
        echo "このリクエストは継続できません";
    }
} else {
    die('Invalid Request');
}
```

【ブラウザ→サーバ: GETリクエスト 33/33-003a.html → レスポンス】

無題セッション - 20181215-183532 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

クイック スタート リクエスト レスポンス

コンテキスト

- 既定コンテキスト
- サイト

デフォルトビュー

```

GET http://example.jp/33/33-003a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
    
```

デフォルトビュー

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 14:37:23 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 403
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "193-56c2a2decafa-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
var req = new XMLHttpRequest();
req.open("POST", "http://api.example.net/33/33-004a.php");
req.setRequestHeader("content-type", "application/json");
data = {"zipcode": "100-0100"};
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
alert(req.responseText);
}
};
req.send(data);
</script>
送信しました
</body>
    
```

履歴 検索 アラート アウトプット

フィルタ: オフ エクスポート

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 38 | 18/12/15 23:37:21 | GET | http://example.jp/33/33-003a.html | 200 | OK | 5 ms | 403 bytes | Medium | | Script |
| 41 | 18/12/15 23:37:21 | OPTIONS | http://api.example.net/33/33-004a.php | 200 | OK | 4 ms | 0 bytes | | | |
| 44 | 18/12/15 23:37:21 | POST | http://api.example.net/33/33-004a.php | 200 | OK | 8 ms | 15 bytes | Medium | | |

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0 0 0

【ブラウザ→APIサーバ: OPTIONSリクエスト api.example.net/33/33-004a.html → レスポンス】

OPTIONS http://api.example.net/33/33-004a.php HTTP/1.1
 User-Agent: Mozilla/5.0 (Windows.NT.10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
 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
 Access-Control-Request-Method: POST
 Access-Control-Request-Headers: content-type
 Origin: http://example.jp
 DNT: 1
 Connection: keep-alive
 Host: api.example.net

HTTP/1.1 200 OK
 Server: nginx/1.10.3
 Date: Sat, 15 Dec 2018 14:37:23 GMT
 Content-Type: text/plain; charset=UTF-8
 Content-Length: 0
 Connection: keep-alive
 Access-Control-Allow-Origin: http://example.jp
 Access-Control-Allow-Methods: POST, GET, OPTIONS
 Access-Control-Allow-Headers: Content-Type
 Access-Control-Max-Age: 1728000
 X-UA-Compatible: IE=edge

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 38 | 18/12/15 23:37:21 | GET | http://example.jp/33/33-003a.html | 200 | OK | 5 ms | 403 bytes | Medium | | Script |
| 41 | 18/12/15 23:37:21 | OPTIONS | http://api.example.net/33/33-004a.php | 200 | OK | 4 ms | 0 bytes | | | |
| 44 | 18/12/15 23:37:21 | POST | http://api.example.net/33/33-004a.php | 200 | OK | 8 ms | 15 bytes | Medium | | |

【ブラウザ→APIサーバ: POSTリクエスト api-example.net/33/33-004a.html → レスポンス】

POST http://api.example.net/33/33-004a.php HTTP/1.1
 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
 Accept: */*
 Accept-Language: ja,en-US;q=0.7,en;q=0.3
 Accept-Encoding: gzip, deflate
 Referer: http://example.jp/33/33-003a.html
 content-type: application/json
 Content-Length: 24
 Origin: http://example.jp
 DNT: 1
 Connection: keep-alive
 Host: api.example.net
 {"zipcode": "100-0100"}

HTTP/1.1 200 OK
 Server: nginx/1.10.3
 Date: Sat, 15 Dec 2018 14:37:23 GMT
 Content-Type: text/html; charset=UTF-8
 Content-Length: 15
 Connection: keep-alive
 X-UA-Compatible: IE=edge
 Invalid Request

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 38 | 18/12/15 23:37:21 | GET | http://example.jp/33/33-003a.html | 200 | OK | 5 ms | 403 bytes | Medium | | Script |
| 41 | 18/12/15 23:37:21 | OPTIONS | http://api.example.net/33/33-004a.php | 200 | OK | 4 ms | 0 bytes | | | |
| 44 | 18/12/15 23:37:21 | POST | http://api.example.net/33/33-004a.php | 200 | OK | 8 ms | 15 bytes | Medium | | |

【ブラウザ】



The screenshot shows a browser window with the address bar containing 'example.jp/33/33-003a.html'. Below the address bar, there is a notification bar that says '送信しました' (Sent). The developer tools console shows a yellow error message: 'クロスオリジン要求をブロックしました: 同一生成元ポリシーにより、http://api.example.net/33/33-004a.php にあるリモートリソースの読み込みは拒否されます (理由: CORS ヘッダー 'Access-Control-Allow-Origin' が足りない)。' (Blocked cross-origin request: Remote resource loading is denied by the same-origin policy for http://api.example.net/33/33-004a.php (reason: CORS header 'Access-Control-Allow-Origin' is missing)).

クロスオリジン要求をブロックしました:
同一生成元ポリシーにより、リモートリソースの読み込みは拒否されます
(理由: CORS ヘッダー 'Access-Control-Allow-Origin' が足りない)。

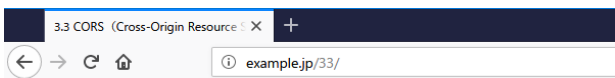


The screenshot shows a browser window with the address bar containing 'view-source:http://example.jp/33/33-003a.html'. The source code is displayed in a monospace font, showing a JavaScript script that makes a POST request to 'http://api.example.net/33/33-004a.php' with a JSON body containing a 'zipcode' field.


```
1 <body>
2 <script>
3   var req = new XMLHttpRequest();
4   req.open('POST', 'http://api.example.net/33/33-004a.php');
5   req.setRequestHeader("content-type", "application/json");
6   data = '{"zipcode": "100-0100"}';
7   req.onreadystatechange = function() {
8     if (req.readyState == 4 && req.status == 200) {
9       alert(req.responseText);
10    }
11  };
12  req.send(data);
13 </script>
14 送信しました
15 </body>
16
```

33-003a:プリフライトリクエスト(完成)

【ブラウザ】

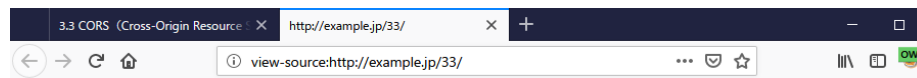


3.3 CORS (Cross-Origin Resource Sharing)

1. 33-001 : シンプルなリクエスト
2. 33-001a: シンプルなリクエスト(CORS対応)
3. 33-003 : プリフライトリクエストなし
4. 33-003a: プリフライトリクエストあり
5. 33-003a: プリフライトリクエスト (完成) 
6. 33-005 : アクセスカウンタ (withCredentialsなし)
7. 33-005a: アクセスカウンタ (withCredentialsあり)
8. 33-005b: アクセスカウンタ (Access-Control-Allow-Credentialsあり)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="33-001a.html">33-001a: シンプルなリクエスト (CORS対応)</a></li>
8 <li><a href="33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="33-003a.html">33-003a: プリフライトリクエストあり</a></li>
10 <li><a href="33-003b.html">33-003b: プリフライトリクエスト (完成) </a></li>
11 <li><a href="33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="33-005a.html">33-005a: アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="33-005b.html">33-005b: アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-003b.html】

```
 /var/www/html/33/33-003b.html - wasbook@192.168.56.101 - エディタ - WinSCP
<body>
<script>
var req = new XMLHttpRequest();
req.open('POST', 'http://api.example.net/33/33-004b.php');
req.setRequestHeader('content-type', 'application/json');
data = '{"zipcode": "100-0100"}';
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    alert(req.responseText);
  }
};
req.send(data);
</script>
送信しました
</body>
```

【サーバ: 33/33-004a.html】

```
 /var/www/html/33/33-004a.php - wasbook@192.168.56.101 - エディタ - WinSCP
<?php
if ($_SERVER['REQUEST_METHOD'] === "OPTIONS") {
  if ($_SERVER['HTTP_ORIGIN'] == "http://example.jp") {
    header('Access-Control-Allow-Origin: http://example.jp');
    header('Access-Control-Allow-Methods: POST, GET, OPTIONS');
    header('Access-Control-Allow-Headers: Content-Type');
    header('Access-Control-Max-Age: 10');
    header("Content-Length: 0");
    header("Content-Type: text/plain");
  } else {
    header("HTTP/1.1 403 Access Forbidden");
    header("Content-Type: text/plain");
    echo "このリクエストは継続できません";
  }
} elseif ($_SERVER['REQUEST_METHOD'] === "POST") {
  header('Content-Type: application/json');
  header('Access-Control-Allow-Origin: http://example.jp');
  header('Access-Control-Max-Age: 10');
  echo json_encode(['zipcode' => '100-0100', 'address' => '東京都大島町']);
} else {
  die('Invalid Request');
}
```

プリフライトリクエスト
をやり取り

【ブラウザ→サーバ: GETリクエスト 33/33-003b.html → レスポンス】

無題セッション - 20181215-183532 - OWASP ZAP 2.7.0

ファイル 編集 表示 ポリシー レポート ツール オンライン ヘルプ

標準モード

サイト クイックスタート リクエスト レスポンス

コンテキスト

- 既定コンテキスト
- サイト

デフォルトビュー

```
GET http://example.jp/33/33-003b.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 14:57:44 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 403
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "193-56c2a2dece3b-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
var req = new XMLHttpRequest();
req.open("POST", "http://api.example.net/33/33-004b.php");
req.setRequestHeader("content-type", "application/json");
data = {"zipcode": "100-0100"};
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    alert(req.responseText);
  }
};
req.send(data);
</script>
送信しました
</body>
```

履歴 検索 アラート アウトプット

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 47 | 18/12/15 23:57:42 | GET | http://example.jp/33/33-003b.html | 200 | OK | 5 ms | 403 bytes | Medium | | Script |
| 50 | 18/12/15 23:57:42 | OPTIONS | http://api.example.net/33/33-004b.php | 200 | OK | 5 ms | 0 bytes | | | |
| 53 | 18/12/15 23:57:42 | POST | http://api.example.net/33/33-004b.php | 200 | OK | 7 ms | 71 bytes | Low | | JSON |

アラート 0 1 2 0

現在のスキャン 0 0 0 0 0 0 0 0

【ブラウザ→APIサーバ: OPTIONSリクエスト api.example.net/33/33-004b.html → レスポンス】

The screenshot shows the OWASP ZAP 2.7.0 interface. The top toolbar includes buttons for 'サイト' (Site), 'クイックスタート' (Quick Start), 'リクエスト' (Request), and 'レスポンス' (Response). The main area is split into two panes: 'リクエスト' (Request) on the left and 'レスポンス' (Response) on the right. Both panes are currently displaying the details of an OPTIONS request and its corresponding response.

Request Details:

```

OPTIONS http://api.example.net/33/33-004b.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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
Access-Control-Request-Method: POST
Access-Control-Request-Headers: content-type
Origin: http://example.jp
DNT: 1
Connection: keep-alive
Host: api.example.net
    
```

Response Details:

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sat, 15 Dec 2018 14:57:44 GMT
Content-Type: text/plain;charset=UTF-8
Content-Length: 0
Connection: keep-alive
Access-Control-Allow-Origin: http://example.jp
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: Content-Type
Access-Control-Max-Age: 10
X-UA-Compatible: IE=edge
    
```

At the bottom, the '履歴' (History) table shows a list of requests:

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 47 | 18/12/15 23:57:42 | GET | http://example.jp/33/33-003b.html | 200 | OK | 5 ms | 403 bytes | Medium | | Script |
| 50 | 18/12/15 23:57:42 | OPTIONS | http://api.example.net/33/33-004b.php | 200 | OK | 5 ms | 0 bytes | | | |
| 53 | 18/12/15 23:57:42 | POST | http://api.example.net/33/33-004b.php | 200 | OK | 7 ms | 71 bytes | Low | | JSON |

The status bar at the bottom indicates '現在のスキャン' (Current Scan) with various icons and counts.

【ブラウザ→APIサーバ: POSTリクエスト api.example.net/33/33-004b.html → レスポンス】

The screenshot shows the OWASP ZAP interface. The left pane shows the context tree with 'コンテキスト' expanded. The main pane is split into 'リクエスト' (Request) and 'レスポンス' (Response) views. The request is a POST to http://api.example.net/33/33-004b.php with headers including User-Agent, Accept, and Referer. The response is an HTTP 200 OK with headers including Server, Date, Content-Type, and Content-Length. The response body is a JSON object: {"zipcode": "100-0100"}. Below the main pane is a table of request history.

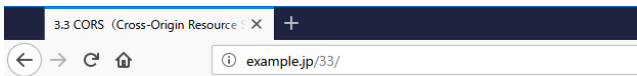
| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|---------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 47 | 18/12/15 23:57:42 | GET | http://example.jp/33/33-003b.html | 200 | OK | 5 ms | 403 bytes | Medium | | Script |
| 50 | 18/12/15 23:57:42 | OPTIONS | http://api.example.net/33/33-004b.php | 200 | OK | 5 ms | 0 bytes | | | |
| 53 | 18/12/15 23:57:42 | POST | http://api.example.net/33/33-004b.php | 200 | OK | 7 ms | 71 bytes | Low | | JSON |

【ブラウザ】


The screenshot shows a browser window with the address bar containing 'example.jp/33/33-003b.html'. A dialog box is displayed with the message '送信しました' (Message sent) and the JSON response: {"zipcode": "100-0100", "address": "\u6771\u4eac\u90fd\u5927\u5cf6\u753a"}. An 'OK' button is visible at the bottom of the dialog box.

33-005 :アクセスカウンタ (withCredentialsなし)

【ブラウザ】

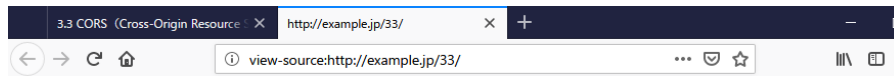


3.3 CORS (Cross-Origin Resource Sharing)

1. 33-001 : シンプルなリクエスト
2. 33-001a : シンプルなリクエスト (CORS対応)
3. 33-003 : プリフライトリクエストなし
4. 33-003a : プリフライトリクエストあり
5. 33-003a : プリフライトリクエスト (完成)
6. 33-005 : アクセスカウンタ (withCredentialsなし) 
7. 33-005a : アクセスカウンタ (withCredentialsあり)
8. 33-005b : アクセスカウンタ (Access-Control-Allow-Credentialsあり)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="33-001a.html">33-001a : シンプルなリクエスト (CORS対応)</a></li>
8 <li><a href="33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="33-003a.html">33-003a : プリフライトリクエストあり</a></li>
10 <li><a href="33-003b.html">33-003a : プリフライトリクエスト (完成) </a></li>
11 <li><a href="33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="33-005a.html">33-005a : アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="33-005b.html">33-005b : アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-005.html】

```
/var/www/html/33/33-005.html - wasbook@192.168.56.101 - エディタ - WinSCP
<body>
<script>
var req = new XMLHttpRequest();
req.open('GET', 'http://api.example.net/33/33-006.php');
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    var span = document.getElementById('counter');
    span.textContent = req.responseText;
  }
};
req.send(null);
</script>
呼び出しカウンター:<span id="counter"></span>
</body>
```

【サーバ: 33/33-006.php】

```
/var/www/html/33/33-006.php - wasbook@192.168.56.101 - エディタ - WinSCP
<?php
session_start();
header('Content-Type: application/json');
header('Access-Control-Allow-Origin: http://example.jp');
if (empty($_SESSION['counter'])) {
  $_SESSION['counter'] = 1;
} else {
  $_SESSION['counter']++;
}
echo json_encode(array('count' => $_SESSION['counter']));
```

【ブラウザ→サーバ: リクエスト 33/33-005.html → レスポンス】

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

- Request:**

```
GET http://example.jp/33/33-005.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
```
- Response:**

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sun, 16 Dec 2018 00:32:25 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 408
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "198-56c2a2decfdb-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
var req = new XMLHttpRequest();
req.open("GET", "http://api.example.net/33/33-006.php");
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    var span = document.getElementById("counter");
    span.textContent = req.responseText;
  }
};
req.send(null);
</script>
呼び出しカウンター: <span id="counter"></span>
</body>
```

The bottom of the interface shows a table with the following data:

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|------------------|------|----------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 8 | 18/12/16 9:32:25 | GET | http://example.jp/33/33-005.html | 200 | OK | 7 ms | 408 bytes | Medium | | Script |

Alerts: 0 Critical, 1 High, 3 Medium, 0 Low, 0 Info, 0 Warning, 0 Error. Current Scans: 0

【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-006.php → レスポンス】

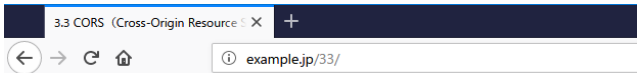
The screenshot shows the OWASP ZAP 2.7.0 interface. The left pane displays the request details for a GET request to `http://api.example.net/33/33-006.php`. The right pane displays the response details, which is an HTTP 200 OK with a JSON body `{'count':1}`. Below the panes is a table of request logs.

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|------------------|------|--------------------------------------|----------|------------|-------------|-------------|--------|-----------------|----|
| 11 | 18/12/16 9:32:25 | GET | http://api.example.net/33/33-006.php | 200 | OK | 5 ms | 11 bytes | Low | SetCookie, JSON | |

【ブラウザ】

The browser window shows the page content: `呼び出しカウンター:{"count":1}`

```
1 </body>
2 <script>
3   var req = new XMLHttpRequest();
4   req.open('GET', 'http://api.example.net/33/33-006.php');
5   req.onreadystatechange = function() {
6     if (req.readyState == 4 && req.status == 200) {
7       var span = document.getElementById('counter');
8       span.textContent = req.responseText;
9     }
10  };
11  req.send(null);
12 </script>
13 呼び出しカウンター:<span id="counter"></span>
14 </body>
15
```



3.3 CORS (Cross-Origin Resource Sharing)

1. [33-001 : シンプルなリクエスト](#)
2. [33-001a : シンプルなリクエスト \(CORS対応\)](#)
3. [33-003 : プリフライトリクエストなし](#)
4. [33-003a : プリフライトリクエストあり](#)
5. [33-003a : プリフライトリクエスト \(完成\)](#)
6. [33-005 : アクセスカウンタ \(withCredentialsなし\)](#)
7. [33-005a : アクセスカウンタ \(withCredentialsあり\)](#)
8. [33-005b : アクセスカウンタ \(Access-Control-Allow-Credentialsあり\)](#)

さらにもう一回クリック

[phpinfo](#)
[ホームに戻る](#)

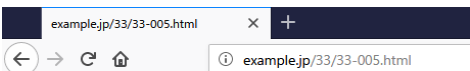
【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-006.php → レスポンス】

cookie がセットされていない

cookie がセットされていないので、カウントがアップされない

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|------------------|------|--------------------------------------|----------|------------|-------------|-------------|--------|-----------------|----|
| 14 | 18/12/16 9:55:29 | GET | http://api.example.net/33/33-006.php | 200 | OK | 13 ms | 11 bytes | Low | SetCookie, JSON | |

【ブラウザ】

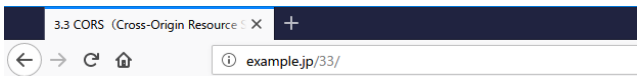


呼び出しカウンター: {"count":1}


カウントアップされない

33-005a:アクセスカウンタ(withCredentialsあり)

【ブラウザ】

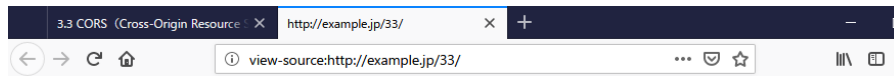


3.3 CORS (Cross-Origin Resource Sharing)

1. 33-001 : シンプルなリクエスト
2. 33-001a : シンプルなリクエスト(CORS対応)
3. 33-003 : プリフライトリクエストなし
4. 33-003a : プリフライトリクエストあり
5. 33-003a : プリフライトリクエスト (完成)
6. 33-005 : アクセスカウンタ (withCredentialsなし)
7. 33-005a : アクセスカウンタ (withCredentialsあり) 
8. 33-005b : アクセスカウンタ (Access-Control-Allow-Credentialsあり)

[phpinfo](#)

[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="33-001a.html">33-001a : シンプルなリクエスト(CORS対応)</a></li>
8 <li><a href="33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="33-003a.html">33-003a : プリフライトリクエストあり</a></li>
10 <li><a href="33-003b.html">33-003a : プリフライトリクエスト (完成) </a></li>
11 <li><a href="33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="33-005a.html">33-005a : アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="33-005b.html">33-005b : アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-005a.php】

```
/var/www/html/33/33-005a.html - wasbook@192.168.56.101 - エディタ - WinSCP
kbody>
<script>
var req = new XMLHttpRequest();
req.open('GET', 'http://api.example.net/33/33-006.php');
req.withCredentials = true;
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
var span = document.getElementById('counter');
span.textContent = req.responseText;
}
};
req.send(null);
</script>
呼び出しカウンター:<span id="counter"></span>
</body>
```

【サーバ: 33/33-006.php】

```
/var/www/html/33/33-006.php - wasbook@192.168.56.101 - エディタ - WinSCP
k?php
session_start();
header('Content-Type: application/json');
header('Access-Control-Allow-Origin: http://example.jp');
if (empty($_SESSION['counter'])) {
$_SESSION['counter'] = 1;
} else {
$_SESSION['counter']++;
}
echo json_encode(array('count' => $_SESSION['counter']));
```

【ブラウザ→サーバ: リクエスト 33/33-005a.php → レスポンス】

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

- Request (Left Panel):**
GET http://example.jp/33/33-005a.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
- Response (Right Panel):**
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sun, 16 Dec 2018 01:08:09 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 438
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "1b6-56c2a2decddb-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge
- Response Body (HTML):**

```
<body>  
<script>  
var req = new XMLHttpRequest();  
req.open("GET", "http://api.example.net/33/33-006.php");  
req.withCredentials = true;  
req.onreadystatechange = function() {  
  if (req.readyState == 4 && req.status == 200) {  
    var span = document.getElementById("counter");  
    span.textContent = req.responseText;  
  }  
};  
req.send(null);  
</script>  
呼び出しカウンター:<span id="counter"></span>  
</body>
```
- Request Log (Bottom Table):**

| ID | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|--------------------------------------|----------|------------|-------------|-------------|--------|-----|-----------------|
| 15 | 18/12/16 10:08:08 | GET | http://example.jp/33/33-005a.html | 200 | OK | 7 ms | 438 bytes | Medium | | Script |
| 16 | 18/12/16 10:08:08 | GET | http://api.example.net/33/33-006.php | 200 | OK | 6 ms | 11 bytes | Low | | SetCookie, JSON |

アラート: 0 Critical, 1 High, 3 Medium, 0 Low

現在のスキャン: 0 Errors, 0 Warnings, 0 Info, 0 Debug, 0 Disabled

【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-006.php → レスポンス】

The screenshot shows the network tab of a browser's developer tools. The left pane displays the request details for a GET request to `http://api.example.net/33/33-006.php`. The right pane displays the response details, which is an HTTP 200 OK with a JSON body `{'count':1}`. The response headers include `Server: nginx/1.10.3`, `Date: Sun, 16 Dec 2018 01:08:09 GMT`, `Content-Type: application/json`, `Content-Length: 11`, `Connection: keep-alive`, `Set-Cookie: PHPSESSID=n2qt7i608dt1foock0hqpgad1; path=/`, `Expires: Thu, 19 Nov 1981 08:52:00 GMT`, `Cache-Control: no-store, no-cache, must-revalidate`, `Pragma: no-cache`, and `Access-Control-Allow-Origin: http://example.jp`.

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|--------------------------------------|----------|------------|-------------|-------------|--------|-----|-----------------|
| 15 | 18/12/16 10:08:08 | GET | http://example.jp/33/33-005a.html | 200 | OK | 7 ms | 438 bytes | Medium | | Script |
| 16 | 18/12/16 10:08:08 | GET | http://api.example.net/33/33-006.php | 200 | OK | 6 ms | 11 bytes | Low | | SetCookie, JSON |

【ブラウザ】

The screenshot shows a browser window with the address bar containing `example.jp/33/33-005a.html`. The page content displays the text `呼び出しカウンター:`.

The screenshot shows the source code of the page. The code is as follows:

```
1 <body>
2 <script>
3   var req = new XMLHttpRequest();
4   req.open('GET', 'http://api.example.net/33/33-006.php');
5   req.withCredentials = true;
6   req.onreadystatechange = function() {
7     if (req.readyState == 4 && req.status == 200) {
8       var span = document.getElementById('counter');
9       span.textContent = req.responseText;
10    }
11  };
12  req.send(null);
13 </script>
14 呼び出しカウンター:<span id="counter"></span>
15 </body>
16
```

33-005b:アクセスカウンタ(Access-Control-Allow-Credentialsあり)

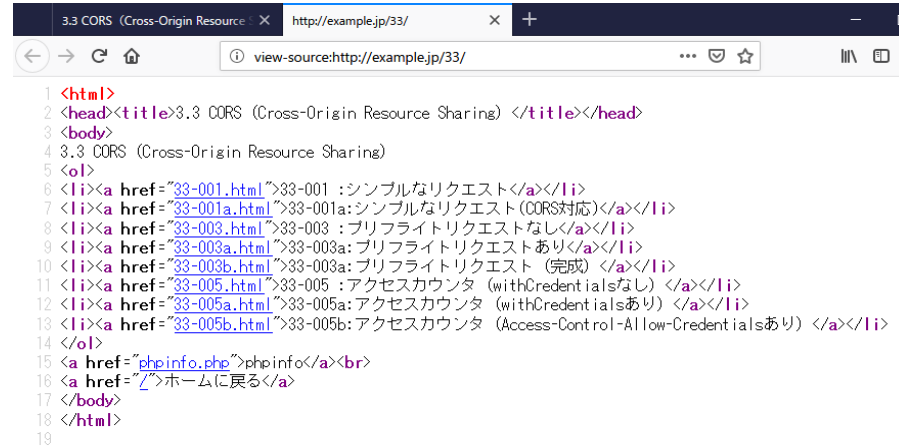
【ブラウザ】



3.3 CORS (Cross-Origin Resource Sharing)

1. 33-001 : シンプルなリクエスト
2. 33-001a : シンプルなリクエスト(CORS対応)
3. 33-003 : プリフライトリクエストなし
4. 33-003a : プリフライトリクエストあり
5. 33-003a : プリフライトリクエスト (完成)
6. 33-005 : アクセスカウンタ (withCredentialsなし)
7. 33-005a : アクセスカウンタ (withCredentialsあり)
8. 33-005b : アクセスカウンタ (Access-Control-Allow-Credentialsあり)

[phpinfo](#)
[ホームに戻る](#)



```
1 <html>
2 <head><title>3.3 CORS (Cross-Origin Resource Sharing) </title></head>
3 <body>
4 3.3 CORS (Cross-Origin Resource Sharing)
5 <ol>
6 <li><a href="33-001.html">33-001 : シンプルなリクエスト</a></li>
7 <li><a href="33-001a.html">33-001a : シンプルなリクエスト(CORS対応)</a></li>
8 <li><a href="33-003.html">33-003 : プリフライトリクエストなし</a></li>
9 <li><a href="33-003a.html">33-003a : プリフライトリクエストあり</a></li>
10 <li><a href="33-003b.html">33-003a : プリフライトリクエスト (完成) </a></li>
11 <li><a href="33-005.html">33-005 : アクセスカウンタ (withCredentialsなし) </a></li>
12 <li><a href="33-005a.html">33-005a : アクセスカウンタ (withCredentialsあり) </a></li>
13 <li><a href="33-005b.html">33-005b : アクセスカウンタ (Access-Control-Allow-Credentialsあり) </a></li>
14 </ol>
15 <a href="phpinfo.php">phpinfo</a><br>
16 <a href="/">ホームに戻る</a>
17 </body>
18 </html>
19
```

【サーバ: 33/33-005b.php】

```
/var/www/html/33/33-005b.html - wasbook@192.168.56.101 - エディタ - WinSCP
kbody>
<script>
var req = new XMLHttpRequest();
req.open('GET', 'http://api.example.net/33/33-006b.php');
req.withCredentials = true;
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
var span = document.getElementById('counter');
span.textContent = req.responseText;
}
};
req.send(null);
</script>
呼び出しカウンター:<span id="counter"></span>
</body>
```

【サーバ: 33/33-006b.php】

```
/var/www/html/33/33-006b.php - wasbook@192.168.56.101 - エディタ - WinSCP
k?php
session_start();
header('Content-Type: application/json');
header('Access-Control-Allow-Origin: http://example.jp');
header('Access-Control-Allow-Credentials: true');
if (empty($_SESSION['counter'])) {
$_SESSION['counter'] = 1;
} else {
$_SESSION['counter']++;
}
echo json_encode(array('count' => $_SESSION['counter']));
```

【ブラウザ→サーバ: リクエスト 33/33-005b.php → レスポンス】

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

- Request (Left Panel):**

```

GET http://example.jp/33/33-005b.html HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
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/33/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Host: example.jp
            
```
- Response (Right Panel):**

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sun, 16 Dec 2018 01:37:01 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 439
Connection: keep-alive
Last-Modified: Mon, 14 May 2018 13:08:18 GMT
ETag: "1b7-56c2a2decafa-gzip"
Accept-Ranges: bytes
Vary: Accept-Encoding
X-UA-Compatible: IE=edge

<body>
<script>
var req = new XMLHttpRequest();
req.open("GET", "http://api.example.net/33/33-006b.php");
req.withCredentials = true;
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    var span = document.getElementById("counter");
    span.textContent = req.responseText;
  }
};
req.send(null);
</script>
呼び出しカウンター:<span id="counter"></span>
</body>
            
```
- Log Table (Bottom):**

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|---------------------------------------|----------|------------|-------------|-------------|--------|-----|--------|
| 17 | 18/12/16 10:37:01 | GET | http://example.jp/33/33-005b.html | 200 | OK | 4 ms | 439 bytes | Medium | | Script |
| 18 | 18/12/16 10:37:01 | GET | http://api.example.net/33/33-006b.php | 200 | OK | 5 ms | 11 bytes | Low | | JSON |

【ブラウザ→APIサーバ: リクエスト api.example.net/33/33-006b.php → レスポンス】

デフォルトビュー

```

GET http://api.example.net/33/33-006b.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
Accept: */*
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://example.jp/33/33-005b.html
Origin: http://example.jp
DNT: 1
Connection: keep-alive
Cookie: PHPSESSID=n2qt7i608dt1foock0hpggad1
Host: api.example.net
    
```

```

HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Sun, 16 Dec 2018 01:37:01 GMT
Content-Type: application/json
Content-Length: 11
Connection: keep-alive
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Access-Control-Allow-Origin: http://example.jp
Access-Control-Allow-Credentials: true
X-UA-Compatible: IE=edge

{"count":2}
    
```

Access-Control-Allow-Credentials ヘッダを true に指定

| Id | リクエスト日時 | メソッド | URL | ステータスコード | ステータスコード説明 | ラウンドトリップタイム | レスポンスボディサイズ | 検出アラート | ノート | タグ |
|----|-------------------|------|---------------------------------------|----------|------------|-------------|-------------|--------|--------|----|
| 17 | 18/12/16 10:37:01 | GET | http://example.jp/33/33-005b.html | 200 | OK | 4 ms | 439 bytes | Medium | Script | |
| 18 | 18/12/16 10:37:01 | GET | http://api.example.net/33/33-006b.php | 200 | OK | 5 ms | 11 bytes | Low | JSON | |

アラート 0 1 3 0 現在のスキャン 0 0 0 0 0 0 0 0

【ブラウザ】

example.jp/33/33-005b.html

呼び出しカウンター:{"count":2}

カウントアップされた

```

1 <body>
2 <script>
3   var req = new XMLHttpRequest();
4   req.open('GET', 'http://api.example.net/33/33-006b.php');
5   req.withCredentials = true;
6   req.onreadystatechange = function() {
7     if (req.readyState == 4 && req.status == 200) {
8       var span = document.getElementById('counter');
9       span.textContent = req.responseText;
10    }
11  };
12  req.send(null);
13 </script>
14 呼び出しカウンター:<span id="counter"></span>
15 </body>
16
    
```