



**moz://a**

# Web Security in 2017

Johann Hofmann

Me.

**I am** Johann Hofmann

**I work on** Firefox (Security & Privacy)

**Twitter** [@johannh](https://twitter.com/johannh)

**GitHub** [@johannhof](https://github.com/johannhof)

# Secure websites



**Secure Connections**



**Secure Web Content**

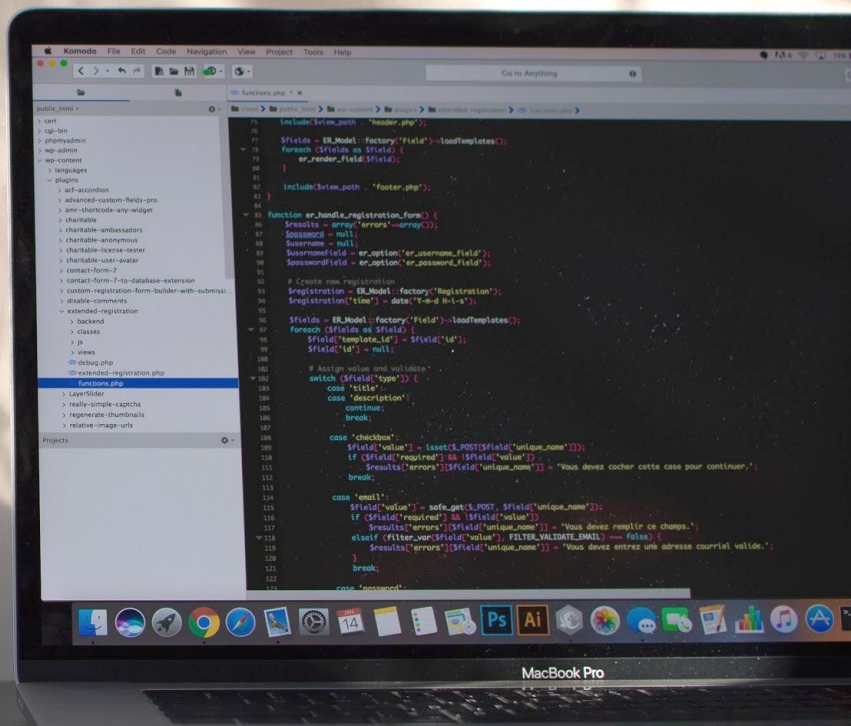


**Secure Access to Device Capabilities**

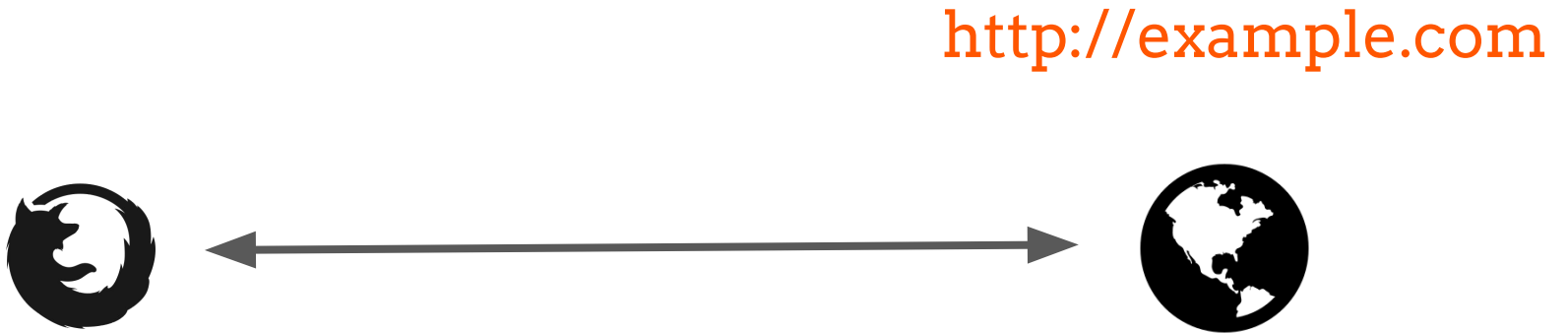
moz://a

# Secure

# Connections

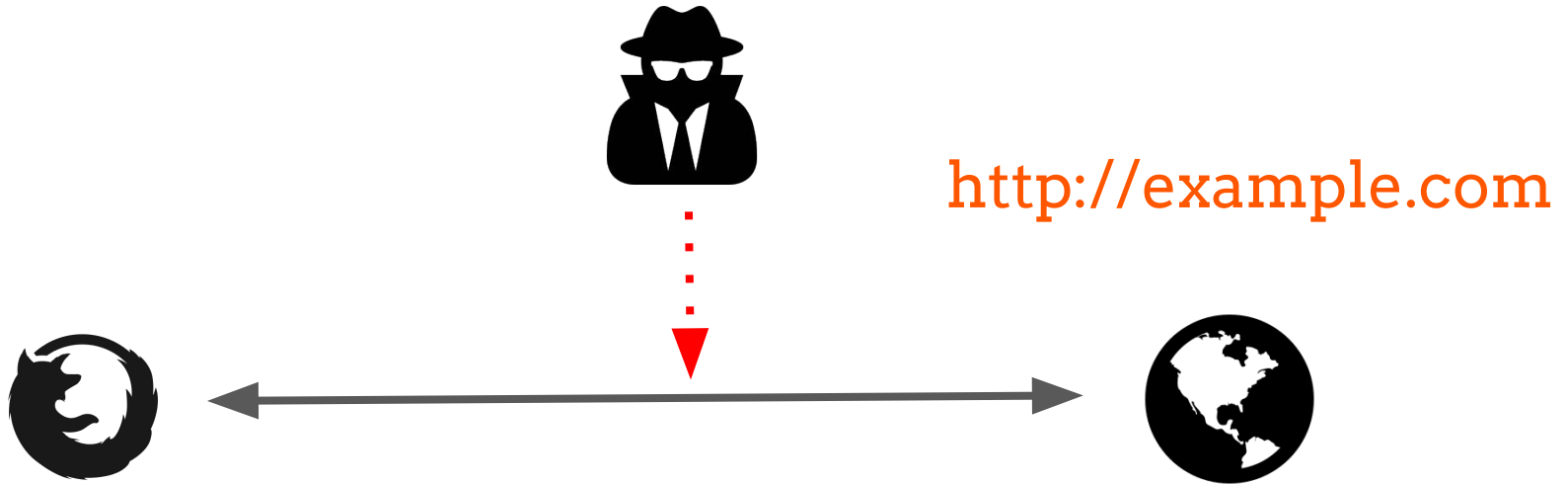


# HTTP



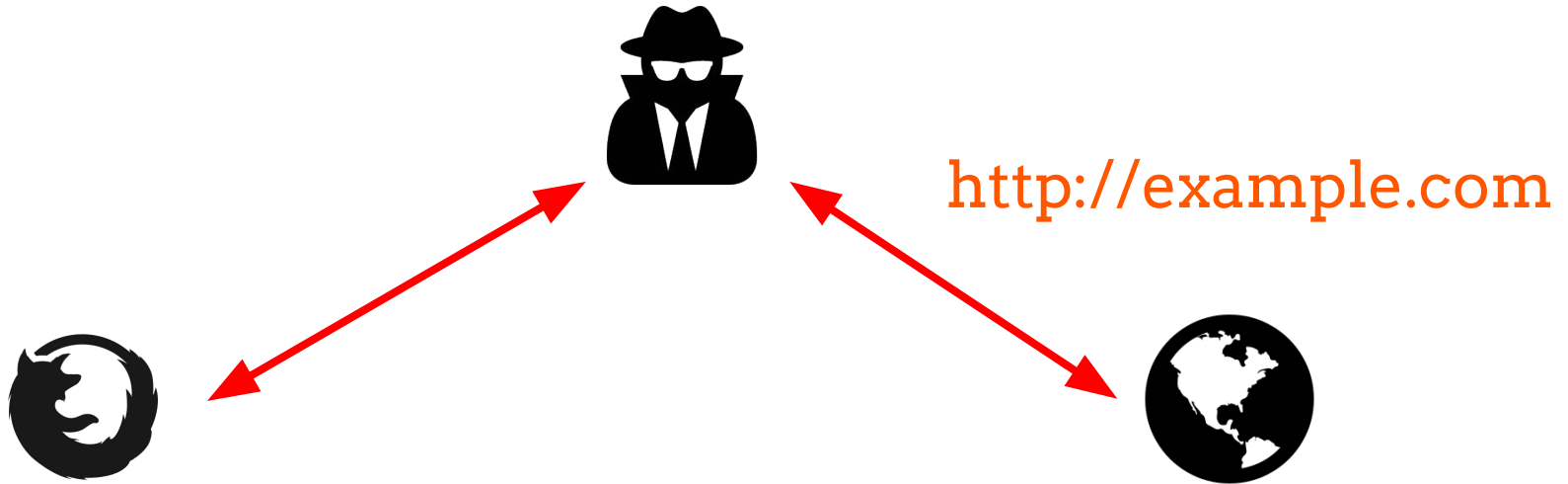
HTTP is the protocol your browser uses to load websites.

# HTTP



Attackers can snoop on HTTP traffic on the same network.

# HTTP



Attackers can intercept and replace (man-in-the-middle) HTTP traffic on the same network.

# HTTP



Attackers can impersonate your entire website.



# Secure Connections

**HTTPS** protects confidentiality and integrity of your and your user's data during transmission.

(Yes it's not perfect, but that's a different story.)

Read more about how protocols on the web work: <https://hpbm.co/>

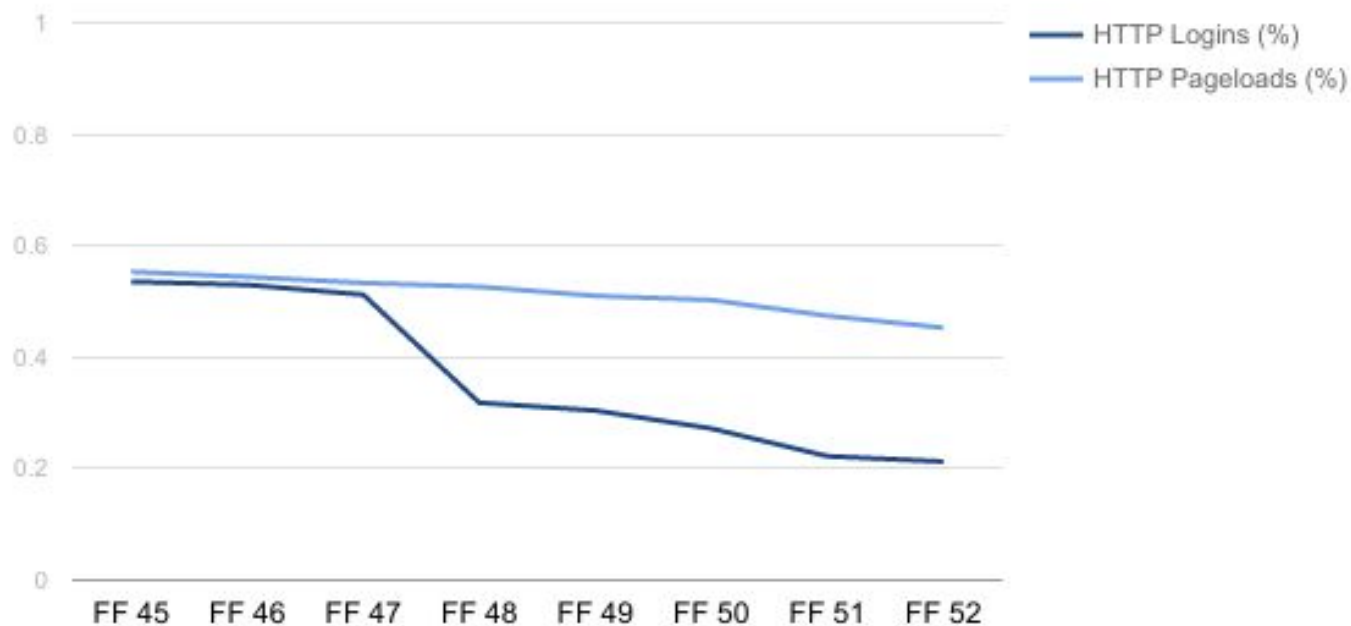
# The bad news

Over 25% of user credentials are transmitted over HTTP.

[https://telemetry.mozilla.org/new-pipeline/dist.html#!cumulative=0&end\\_date=2016-12-09&keys=\\_none !\\_none !\\_none &max\\_channel\\_version=release%252F50&measure=PWMGR\\_LOGIN\\_PAGE\\_SAFETY&min\\_channel\\_version=release%252F46&processType=\\*&product=Firefox&sanitize=1&sort\\_keys=submissions&start\\_date=2016-11-04&table=1&trim=1&use\\_submission\\_date=0](https://telemetry.mozilla.org/new-pipeline/dist.html#!cumulative=0&end_date=2016-12-09&keys=_none !_none !_none &max_channel_version=release%252F50&measure=PWMGR_LOGIN_PAGE_SAFETY&min_channel_version=release%252F46&processType=*&product=Firefox&sanitize=1&sort_keys=submissions&start_date=2016-11-04&table=1&trim=1&use_submission_date=0)

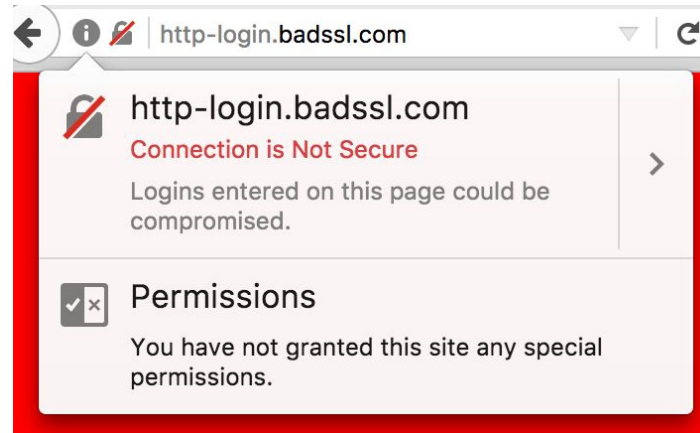
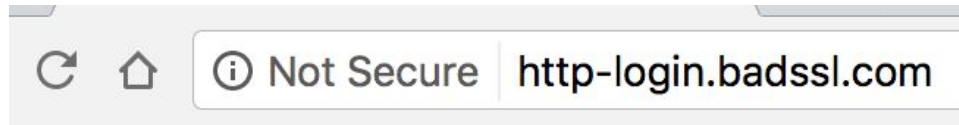
# The good news

## Decline of insecure connections

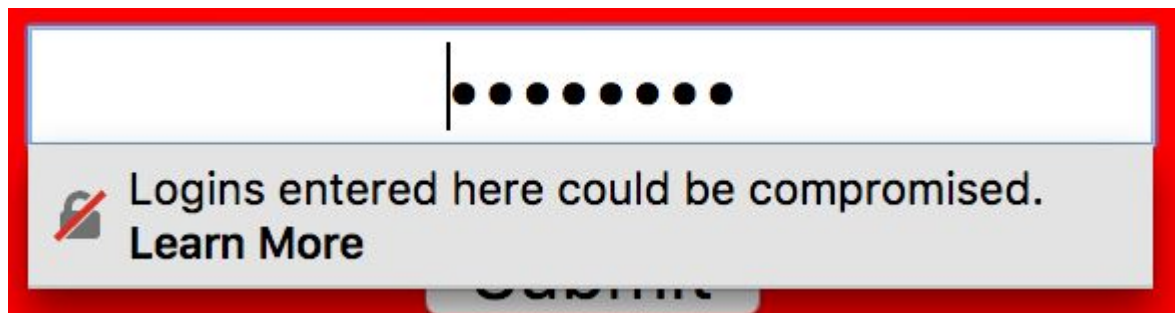


# The switch to HTTPS

Browsers now aggressively flag insecure sites.



# The switch to HTTPS



# The switch to HTTPS

Landing Page ?

**source: firefox-console-errors**

1. [/en-US/docs/Web/Security/Mixed\\_content](/en-US/docs/Web/Security/Mixed_content)
2. [/en-US/docs/Web/Security/Insecure\\_passwords](/en-US/docs/Web/Security/Insecure_passwords)
3. [/en-US/docs/Web/Security/Weak\\_Signature\\_Algorithm](/en-US/docs/Web/Security/Weak_Signature_Algorithm)
4. [/en-US/docs/Web/Security/Mixed\\_content/How\\_to\\_fix\\_website\\_with\\_mixed\\_content](/en-US/docs/Web/Security/Mixed_content/How_to_fix_website_with_mixed_content)
5. [/en-US/docs/Web/Security/Securing\\_your\\_site/Turning\\_off\\_form\\_autocompletion](/en-US/docs/Web/Security/Securing_your_site/Turning_off_form_autocompletion)
6. [/en-US/docs/Web/Security/Same-origin\\_policy](/en-US/docs/Web/Security/Same-origin_policy)

\* MDN Google Analytics: Pageviews in /en-US/docs/Web/Security from utm\_medium=firefox-console-errors

# The switch to HTTPS

It's getting easier to obtain free certificates



<https://certbot.eff.org/>

So I just need to turn on HTTPS?



# Mixed Content

Landing Page ?

source: **firefox-console-errors**

1. [/en-US/docs/Web/Security/Mixed\\_content](/en-US/docs/Web/Security/Mixed_content)
2. [/en-US/docs/Web/Security/Insecure\\_passwords](/en-US/docs/Web/Security/Insecure_passwords)
3. [/en-US/docs/Web/Security/Weak\\_Signature\\_Algorithm](/en-US/docs/Web/Security/Weak_Signature_Algorithm)
4. [/en-US/docs/Web/Security/Mixed\\_content/How\\_to\\_fix\\_website\\_with\\_mixed\\_content](/en-US/docs/Web/Security/Mixed_content/How_to_fix_website_with_mixed_content)
5. [/en-US/docs/Web/Security/Securing\\_your\\_site/Turning\\_off\\_form\\_autocompletion](/en-US/docs/Web/Security/Securing_your_site/Turning_off_form_autocompletion)
6. [/en-US/docs/Web/Security/Same-origin\\_policy](/en-US/docs/Web/Security/Same-origin_policy)

\* MDN Google Analytics: Pageviews in /en-US/docs/Web/Security from utm\_medium=firefox-console-errors

# Mixed Content

## Mixed **Passive** Content (The bad kind)

```

```

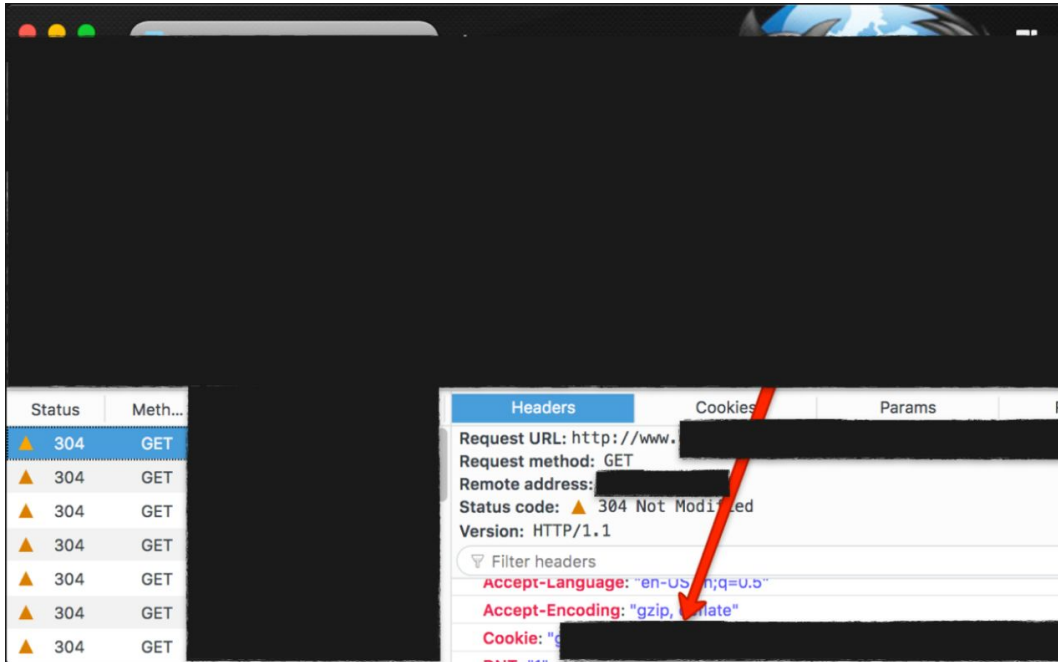
## Mixed **Active** Content (The really bad kind)

```
<script src="http://example.com/script.js"></script>
```

[https://developer.mozilla.org/en-US/docs/Web/Security/Mixed\\_content](https://developer.mozilla.org/en-US/docs/Web/Security/Mixed_content)

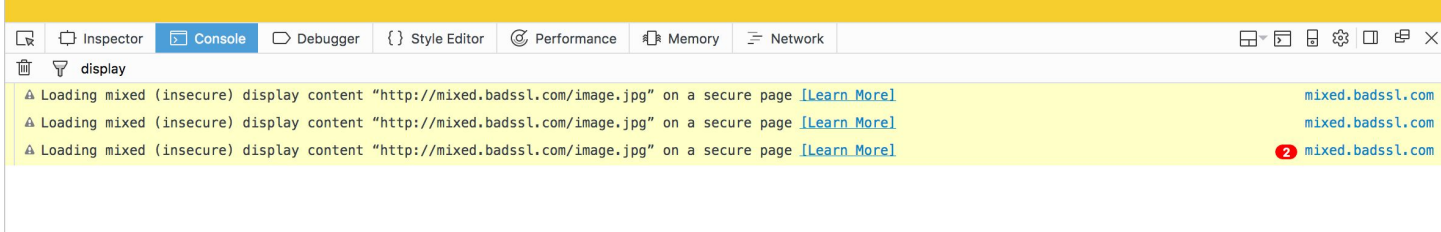
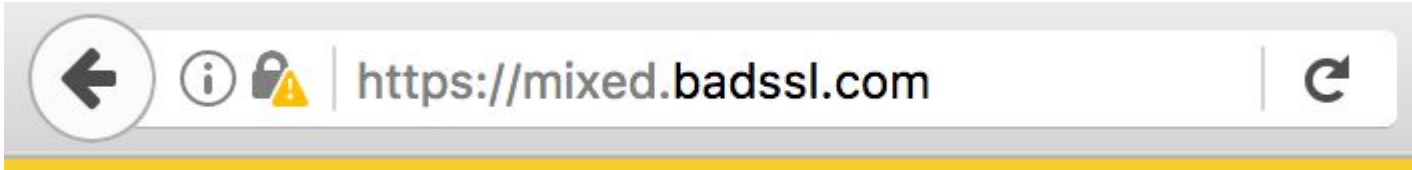
# Mixed Passive Content - Session Snooping

So... mixed passive content is fine, right? Nope.



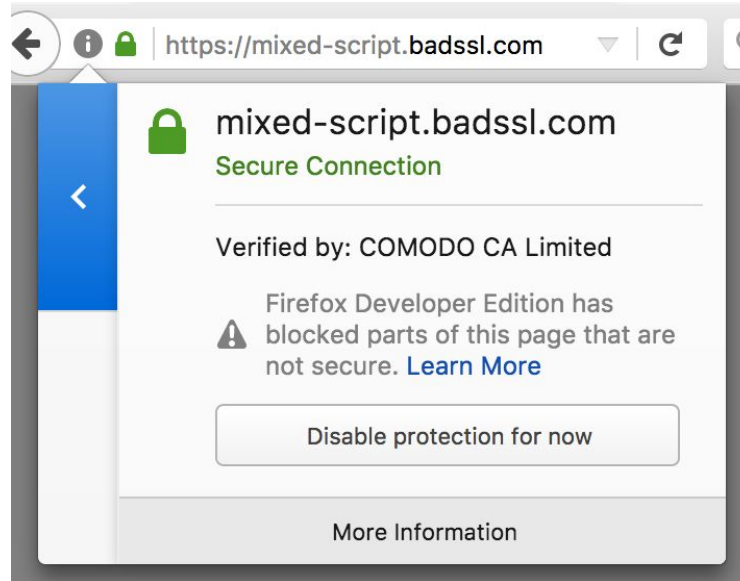
By Luke Crouch (@groovecoder)

# Mixed Passive Content



# Mixed Active Content

## Browsers block mixed active content by default



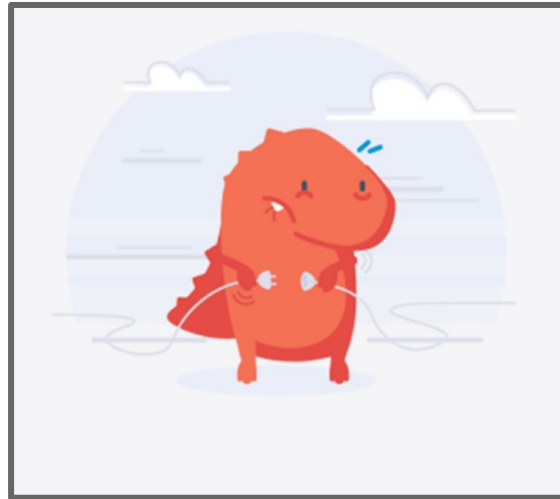
# Mixed Active Content

Can it alter the DOM?

It's Mixed **Active** Content (The really bad kind).

# Secure Connections

The web is moving to HTTPS, don't be a dinosaur!



# Secure Connections

If you've moved to HTTPS (yay),  
consider turning on the HSTS header

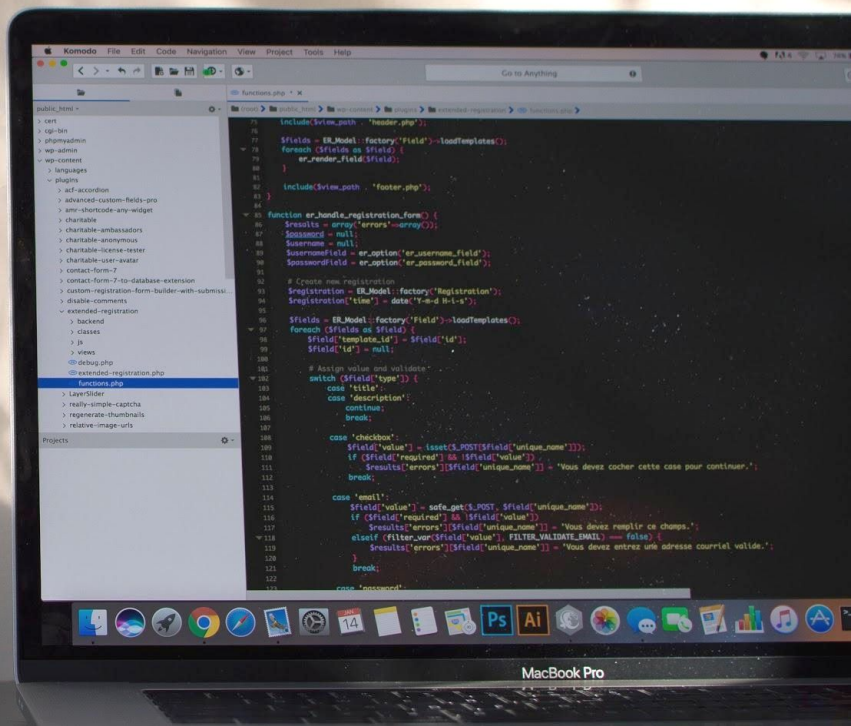
```
Strict-Transport-Security: max-age=31536000;  
includeSubDomains
```

Why? <https://moxie.org/software/sslstrip/>

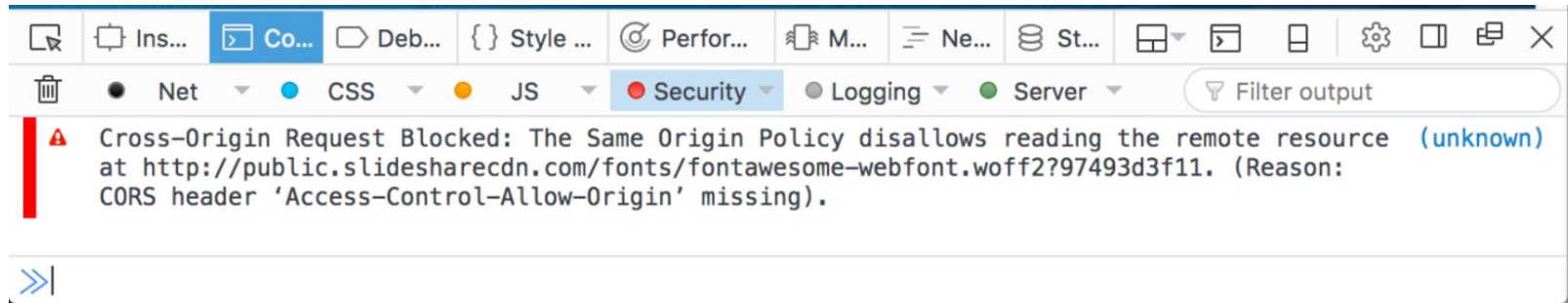


moz://a

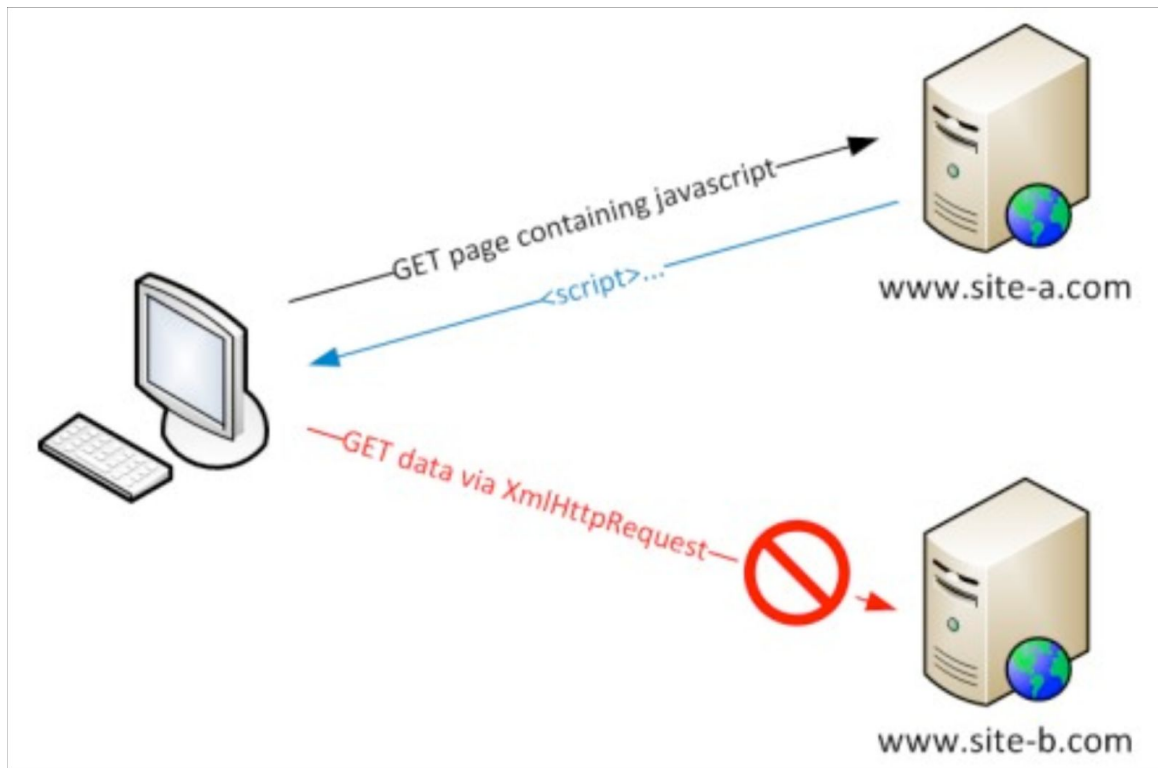
# Secure Web Content



## Same-Origin Policy



# Same-Origin Policy



## What is affected by Same-Origin Policy?

- XHR requests
- `window.opener/window.parent`, etc.
- [https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin\\_policy](https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy)

`<form method="Post">?`

**Not protected!**

Use a CSRF token. Don't rely on cookies.

# Secure Web Content

It's very easy to use

```
Access-Control-Allow-Origin: *
```

to work around this issue. In many cases you should not have to.

How can I make sure that user generated content doesn't break my website security?

**CSP** (+ common sense & sanitization)

<https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP>

# CSP

A set of rules that define how a site may load resources.

```
<meta http-equiv="Content-Security-Policy" content="default-src 'self';  
image-src https:*; child-src 'none';">
```

You can use **Content-Security-Policy-Report-Only** to test it without breaking things



## CSP features recording violations!

```
Content-Security-Policy: default-src https;;  
report-uri https://report.example.com
```

You can set up endpoints at <https://report-uri.io/>

How can I make sure that the third-party content I receive has not changed on the server?

## **Subresource Integrity**

[https://developer.mozilla.org/en-US/docs/Web/Security/Subresource\\_Integrity](https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity)

# Subresource Integrity

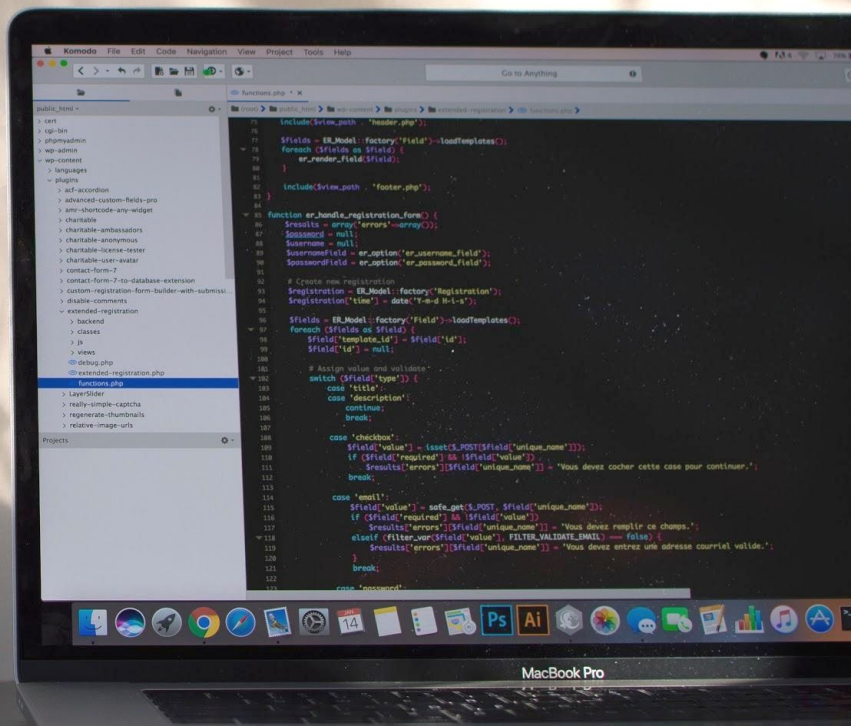
Specify an integrity attribute that is the hash digest of the referenced file.

```
<script src="https://example.com/example-framework.js"  
  integrity="sha384-oqVuAfXRK...234234234"  
  crossorigin="anonymous"></script>
```

<https://www.srihash.org/>

moz://a

# Secure Device Access



# Secure Device Access

The web is getting more native capabilities...



# Secure Device Access

... but we can't let websites access them without permission.



The web platform was not built with a permission system in mind.

*“User agents must acquire permission through a user interface”*

<https://dev.w3.org/geo/api/spec-source.html>

# Secure Device Access

## Requesting permission

```
navigator.mediaDevices.getUserMedia({audio:  
true}).then(stream => ...);
```

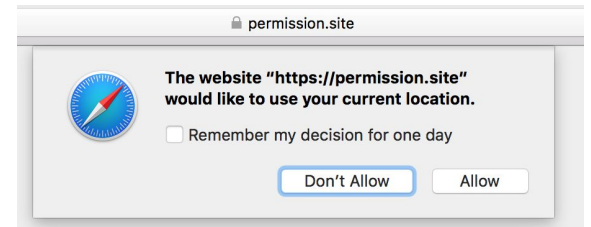
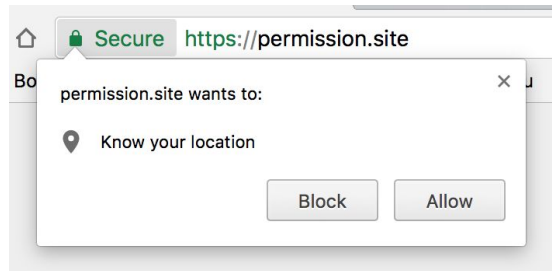
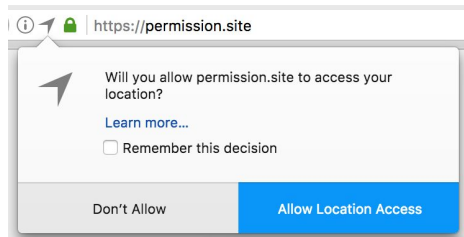
```
Notification.requestPermission().then(result => ...);
```

```
navigator.geolocation.getCurrentPosition(result => ...);
```



# Secure Device Access

## Different prompts from browsers.



Browsers will start to penalize intrusive and surprising permission prompts.

[https://bugzilla.mozilla.org/show\\_bug.cgi?id=1206232](https://bugzilla.mozilla.org/show_bug.cgi?id=1206232)

<https://groups.google.com/a/chromium.org/forum/#!msg/blink-dev/fHI9GNxPPpA/VwhB6h5cDwAJ>

# Secure Device Access

The screenshot shows a mobile browser window with the URL <https://www.t-mobile.com>. A location access permission dialog is displayed in the foreground, asking: "Will you allow www.t-mobile.com to access your location?". The dialog includes a "Learn more..." link, a checkbox for "Remember this decision", and two buttons: "Don't Allow" and "Allow Location Access".

The background website features a pink and black pixelated design. Navigation links include "DEALS", "PHONES", "PLANS", and "MY T-MOBILE". A search icon and a menu icon are also visible. A "LIMITED-TIME OFFER" banner is present, with the main headline: **2 UNLIMITED LINES. \$100. ALL IN.**

Below the headline, the text reads: "Why pay \$600 more every year with Verizon? Switch to T-Mobile and get 2 lines of unlimited LTE data for \$100/mo. with AutoPay—taxes and fees included."

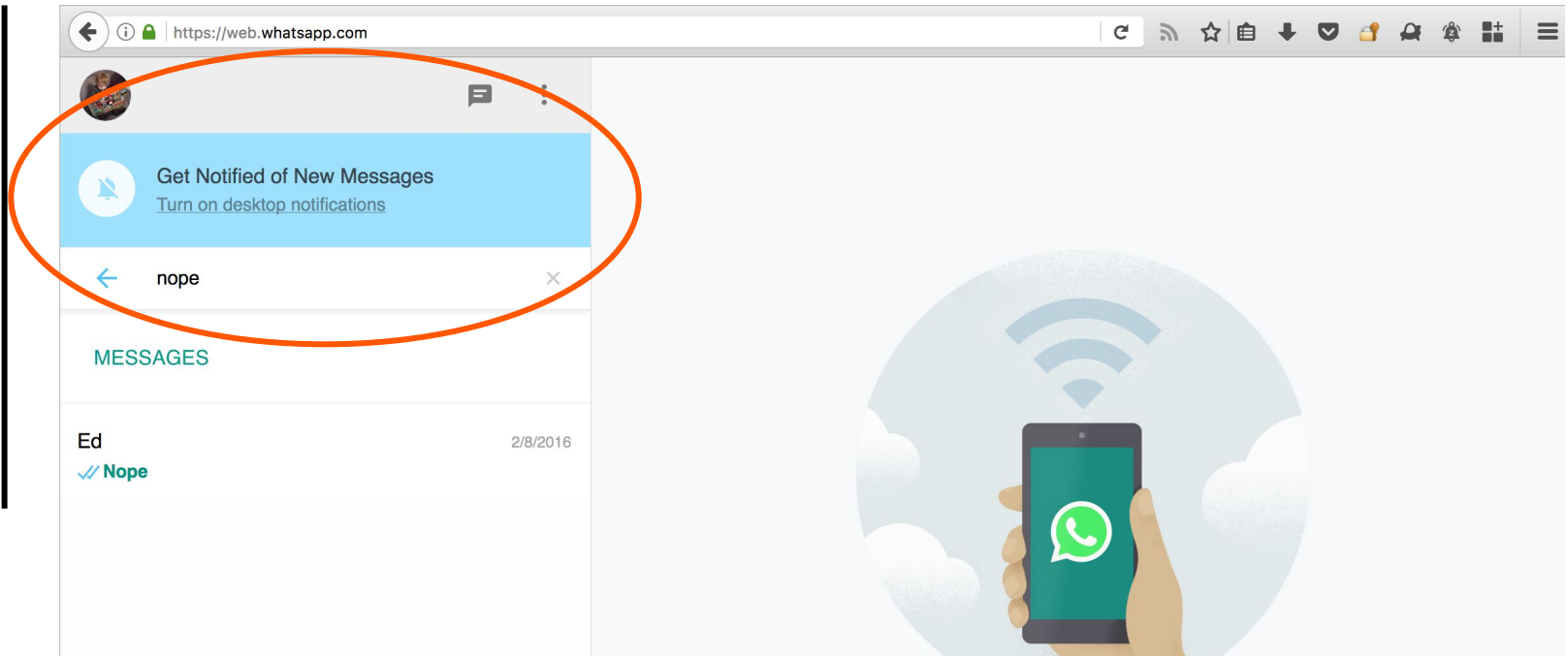
Pre-prompt to give the user context  
and avoid the element of surprise.

(Browsers might actually force you to do that at some point)

# Secure Device Access

The screenshot shows a web browser window with the URL `https://berlinjs.slack.com/messages/@slackbot/`. A notification banner at the top states "Slack needs your permission to enable desktop notifications." The interface includes a sidebar with the "BerlinJS" workspace and a list of channels such as #admin-announcements, #electron, #elm, #events, #hi, #javascript, #job-search, #kiezcoding, #main, and #nodeschool. The main chat area displays a message from "slackbot" with a timestamp of "January 22nd, 2016". The message text is: "You can ask me simple questions about how Slack works, or just type a few keywords. For example: **Can I edit a message I've posted?** Or simply: **edit message.** I'm only a bot, but I'll do my best to answer! If I don't understand, I'll search the [Help Center](#)." Below this message is another from "slackbot" dated "March 2nd, 2016" at "2:34 PM", which says: "If you get lost and want some more help, look at our [Help Center](#) or our [guide to getting started on Slack](#). Otherwise, have a lovely day 🙌".

# Secure Device Access



# Secure Device Access

Soon:

Browsers will start to restrict web permissions from iframes

[https://groups.google.com/a/chromium.org/forum/#!msg/blink-dev/n37ij1E\\_1aY/JA-1R\\_pJDwAJ](https://groups.google.com/a/chromium.org/forum/#!msg/blink-dev/n37ij1E_1aY/JA-1R_pJDwAJ)

# Secure Device Access

With great power...





## Secure Connections

- HTTPS All the Things!
- Take care of mixed content
- Set an HSTS header



## Secure Web Content

- Embrace Cross-Origin restrictions
- Have a CSP
- Use Subresource Integrity



## Secure Access to Device Capabilities

- Don't prompt from iframes or HTTP
- Pre-prompt to give the user context

Thanks!

Johann Hofmann

Twitter [@johannh](https://twitter.com/johannh)

GitHub [@johannhof](https://github.com/johannhof)

Website [johannh.me](https://johannh.me)

# Questions

