How do HttpOnly cookies work with AJAX requests?

Asked 16 years, 4 months ago Modified 8 years, 5 months ago Viewed 85k times

208

JavaScript needs access to cookies if AJAX is used on a site with access restrictions based on cookies. Will HttpOnly cookies work on an AJAX site?

Edit: Microsoft created a way to prevent XSS attacks by disallowing JavaScript access to cookies if HttpOnly is specified. FireFox later adopted this. So my question is: If you are using AJAX on a site, like StackOverflow, are Http-Only cookies an option?

Edit 2: Question 2. If the purpose of HttpOnly is to prevent JavaScript access to cookies, and you can still retrieve the cookies via JavaScript through the XmlHttpRequest Object, **what is the point of HttpOnly**?

Edit 3: Here is a quote from Wikipedia:

> When the browser receives such a cookie, it is supposed to use it as usual in the following HTTP exchanges, but not to make it visible to client-side scripts. [32] The `HttpOnly` flag is not part of any standard, and is not implemented in all browsers. Note that there is currently no prevention of reading or writing the session cookie via a XMLHTTPRequest. [33].

I understand that `document.cookie` is blocked when you use HttpOnly. But it seems that you can still read cookie values in the XMLHttpRequest object, allowing for XSS. How does HttpOnly make you any safer than? By making cookies essentially read only?

In your example, I cannot write to your `document.cookie`, but I can still steal your cookie and post it to my domain using the XMLHttpRequest object.

```
<script type="text/javascript">
    var req = null;
    try { req = new XMLHttpRequest(); } catch(e) {}
    if (!req) try { req = new ActiveXObject("Msxml2.XMLHTTP"); } catch(e) {}
    if (!req) try { req = new ActiveXObject("Microsoft.XMLHTTP"); } catch(e) {}
    req.open('GET', 'http://stackoverflow.com/', false);
    req.send(null);
    alert(req.getAllResponseHeaders());
</script>
```

Edit 4: Sorry, I meant that you could send the XMLHttpRequest to the StackOverflow domain, and then save the result of getAllResponseHeaders() to a string, regex out

the cookie, and then post that to an external domain. It appears that Wikipedia and ha.ckers concur with me on this one, but I would love be re-educated...

Final Edit: Ahh, apparently both sites are wrong, this is actually a bug in FireFox. IE6 & 7 are actually the only browsers that currently fully support HttpOnly.

To reiterate everything I've learned:

- HttpOnly restricts all access to document.cookie in IE7 & and FireFox (not sure about other browsers)
- HttpOnly removes cookie information from the response headers in XMLHttpObject.getAllResponseHeaders() in IE7.
- XMLHttpObjects may only be submitted to the domain they originated from, so there is no cross-domain posting of the cookies.

edit: This information is likely no longer up to date.

ajax cookies httponly

Share
Improve this question
Follow

edited Apr 9, 2010 at 13:11

asked Aug 26, 2008 at 13:06
Shawn
19.8k 20 100 153

I threw your example in a greasemonkey script and it looks like FF no longer displays cookies. Excellent research and example. – user34537 Jul 14, 2010 at 22:09

Maybe with Same Origin Policy you can not make an http request to a domain that is not the same the script is running in; however, I believe that you could easily pass the cookies by redirecting the user to a page using window.location and pass al the information through query string parameters. – Luca Marzi Jun 9, 2016 at 8:38

@LucaMarzi "*you can not make an http request to a domain that is not the same the script is running in*" Are you saying that a site X cannot include an image from host Y? (a feature that has been supported by all browsers since Mosaic?) – curiousguy Nov 23, 2018 at 3:46

9 Answers

Sorted by: Highest score (default)

69

Yes, HTTP-Only cookies would be fine for this functionality. They will still be provided with the XmlHttpRequest's request to the server.

In the case of Stack Overflow, the cookies are automatically provided as part of the XmlHttpRequest request. I don't know the implementation details of the Stack

Overflow authentication provider, but that cookie data is probably automatically used to verify your identity at a lower level than the "vote" controller method.

More generally, cookies are **not** required for AJAX. XmlHttpRequest support (or even iframe remoting, on older browsers) is all that is technically required.

However, if you want to provide security for AJAX enabled functionality, then the same rules apply as with traditional sites. You need some method for identifying the user behind each request, and cookies are almost always the means to that end.

> In your example, I cannot write to your document.cookie, but I can still steal your cookie and post it to my domain using the XMLHttpRequest object.

XmlHttpRequest won't make cross-domain requests (for exactly the sorts of reasons you're touching on).

You could normally inject script to send the cookie to your domain using iframe remoting or JSONP, but then HTTP-Only protects the cookie again since it's inaccessible.

Unless you had compromised StackOverflow.com on the server side, you wouldn't be able to steal my cookie.

> Edit 2: Question 2. If the purpose of Http-Only is to prevent JavaScript access to cookies, and you can still retrieve the cookies via JavaScript through the XmlHttpRequest Object, what is the point of Http-Only?

Consider this scenario:

- I find an avenue to inject JavaScript code into the page.
- Jeff loads the page and my malicious JavaScript modifies his cookie to match mine.
- Jeff submits a stellar answer to your question.
- Because he submits it with my cookie data instead of his, the answer will become mine.
- You vote up "my" stellar answer.
- My real account gets the point.

With HTTP-Only cookies, the second step would be impossible, thereby defeating my XSS attempt.

> Edit 4: Sorry, I meant that you could send the XMLHttpRequest to the StackOverflow domain, and then save the result of getAllResponseHeaders() to a string, regex out the cookie, and then post that to an external domain. It appears that Wikipedia and ha.ckers concur with me on this one, but I would love be re-educated...

That's correct. You can still session hijack that way. It does significantly thin the herd of people who can successfully execute even that XSS hack against you though.

However, if you go back to my example scenario, you can see where HTTP-Only *does* successfully cut off the XSS attacks which rely on modifying the client's cookies (not uncommon).

It boils down to the fact that a) no single improvement will solve *all* vulnerabilities and b) no system will *ever* be completely secure. HTTP-Only **is** a useful tool in shoring up against XSS.

Similarly, even though the cross domain restriction on XmlHttpRequest isn't 100% successful in preventing all XSS exploits, you'd still never dream of removing the restriction.

Share

Improve this answer

Follow

edited Mar 21, 2010 at 22:32

Jeff Atwood

63.9k ● 48 ● 151 ● 153

answered Aug 26, 2008 at 22:24

Dave Ward

60.5k ● 14 ● 118 ● 134

> Many frameworks put `csrf` token in cookies. I suppose AJAX call that needs a `csrf` check won't work unless you put the csrf token in a hidden HTML element for the JS to retrieve it. – user Feb 5, 2015 at 14:26

5

Yes, they are a viable option for an Ajax based site. Authentication cookies aren't for manipulation by scripts, but are simply included by the browser on all HTTP requests made to the server.

Scripts don't need to worry about what the session cookie says - as long as you are authenticated, then any requests to the server initiated by either a user or the script will include the appropriate cookies. The fact that the scripts cannot themselves know the content of the cookies doesn't matter.

For any cookies that are used for purposes other than authentication, these can be set without the HTTP only flag, if you want script to be able to modify or read these. You can pick and choose which cookies should be HTTP only, so for example anything non-sensitive like UI preferences (sort order, collapse left hand pane or not) can be shared in cookies with the scripts.

I really like the HTTP only cookies - it's one of those proprietary browser extensions that was a really neat idea.

Share Improve this answer Follow

answered Feb 26, 2009 at 12:16

thomasrutter

117k • 31 • 152 • 170

4

Not necessarily, it depends what you want to do. Could you elaborate a bit? AJAX doesn't need access to cookies to work, it can make requests on its own to extract information, the page request that the AJAX call makes could access the cookie data & pass that back to the calling script without Javascript having to directly access the cookies

Share Improve this answer Follow

answered Aug 26, 2008 at 13:18

Glenn Slaven

34.2k • 27 • 117 • 166

2

As clarification - from the server's perspective, the page that is requested by an AJAX request is essentially no different to a standard HTTP get request done by the user clicking on a link. All the normal request properties: user-agent, ip, session, cookies, etc. are passed to the server.

Share Improve this answer Follow

answered Aug 26, 2008 at 13:31

Glenn Slaven

34.2k • 27 • 117 • 166

> The "session" isn't an HTTP concept. It's an high level concept built on top of HTTP concepts by a framework. – curiousguy Nov 23, 2018 at 4:28

2

There's a bit more to this.

Ajax doesn't strictly require cookies, but they can be useful as other posters have mentioned. Marking a cookie HTTPOnly to hide it from scripts only partially works, because not all browsers support it, but also because there are common workarounds.

It's odd that the XMLHTTPresponse headers are giving the cookie, technically the server doesn't have to return the cookie with the response. Once it's set on the client, it stays set until it expires. Though there are schemes in which the cookie is changed

with every request to prevent re-use. So you may be able to avoid that workaround by changing the server to not provide the cookie on the XMLHTTP responses.

In general though, I think HTTPOnly should be used with some caution. There are cross site scripting attacks where an attacker arranges for a user to submit an ajax-like request originating from another site, using simple post forms, without the use of XMLHTTP, and your browser's still-active cookie would authenticate the request.

If you want to be sure that an AJAX request is authenticated, the request itself AND the HTTP headers need to contain the cookie. Eg through the use of scripts or unique hidden inputs. HTTPOnly would hinder that.

Usually the interesting reason to want HTTPOnly is to prevent third-party content included on your webpage from stealing cookies. But there are many interesting reasons to be very cautious about including third-party content, and filter it aggressively.

Share Improve this answer Follow

answered Jun 10, 2009 at 14:17

davenpcj

12.7k ● 5 ● 42 ● 38

I should mention, the current best practice is to validate the payload and the request, as I mentioned above, but use HTTPOnly *secure* cookie to validate the request, and a different token with the payload so HTTPOnly isn't a hindrance. – davenpcj May 28, 2021 at 14:55

1

Cookies are automatically handled by the browser when you make an AJAX call, so there's no need for your Javascript to mess around with cookies.

Share Improve this answer Follow

answered Aug 26, 2008 at 13:26

pkchukiss

400 ● 4 ● 14

1

> Therefore I am assuming JavaScript needs access to your cookies.

All HTTP requests from your browser transmit your cookie information for the site in question. JavaScript can both set and read cookies. Cookies are not by definition required for Ajax applications, but they are required for most web applications to maintain user state.

The formal answer to your question as phrased - "Does JavaScript need access to cookies if AJAX is used?" - is therefore "no". Think of enhanced search fields that use Ajax requests to provide auto-suggest options, for example. There is no need of cookie information in that case.

Share Improve this answer Follow

answered Aug 26, 2008 at 13:31

Polsonby

22.9k ● 19 ● 60 ● 74

> XmlHttpRequest needs cookies. The enhanced search you mention might be behind a login page. But whether or not Javascript needs to be able to expose the cookie value to the VM is a different question. – Mr. Shiny and New 安宇 Mar 23, 2009 at 15:54

0

No, the page that the AJAX call requests has access to cookies too & that's what checks whether you're logged in.

You can do other authentication with the Javascript, but I wouldn't trust it, I always prefer putting any sort of authentication checking in the back-end.

Share Improve this answer Follow

answered Aug 26, 2008 at 13:23

Glenn Slaven

34.2k ● 27 ● 117 ● 166

0

Yes, cookies are very useful for Ajax.

Putting the authentication in the request URL is bad practice. There was a news item last week about getting the authentication tokens in the URL's from the google cache.

No, there is no way to prevent attacks. Older browsers still allow trivial access to cookies via javascript. You can bypass http only, etc. Whatever you come up with can be gotten around given enough effort. The trick is to make it too much effort to be worthwhile.

If you want to make your site more secure (there is no perfect security) you could use an authentication cookie that expires. Then, if the cookie is stolen, the attacker must use it before it expires. If they don't then you have a good indication there's suspicious activity on that account. The shorter the time window the better for security but the more load it puts on your server generating and maintaining keys.

Share Improve this answer Follow

answered Mar 23, 2009 at 15:35

Jay

14.4k ● 5 ● 46 ● 74