Haketilo - Feature #71

[Roadmap 5][Milestone] Make it possible for injected scripts to bypass CORS

08/02/2021 04:09 PM - koszko

Status:	Closed	**Start date:**	08/02/2021
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Cross-Origin Resource Sharing (CORS) is a mechanism through which browsers can decide whether a page should or should not be able to access some third-party resource. Despite being a security feature, this also limits the abilities of our injected scripts. There are at least 2 cases where we would like scripts to be able to bypass CORS:

1. When the original page gets some important data from a third-party script (included as <script src="https://some.third.party.com/some.js"></script>) and the script is served in a way CORS blocks its download if it is requested through AJAX instead of through <script> tag.
2. When we deliberately want to add some features that would not be normally possible.

Scripts running in a privileged context of a WebExtension are allowed to bypass CORS. A page script could communicate with those using messaging. We only need to implement the required API and allow special permissions for its use to be specified on package-by-package basis in the settings.

Roadmap

Related issues:

Blocks Site fixes - Feature #107: [Roadmap 5] A user-controlled reCAPTCHA cli...	**Closed**	**02/24/2022**
Blocks Haketilo - Feature #72: [Roadmap 18][Milestone] Facilitate creation of...	**New**	**08/02/2021**

History

#1 - 08/02/2021 11:49 PM - jahoti

- Parent task set to #72

While they're not the **only** use (as outlined in the description), meta-sites will almost certainly be the main application.

#2 - 08/03/2021 12:50 AM - koszko

BTW, we could also facilitate spoofing of the referer header for similar purposes

EDIT: GreaseMonkey actually has something[1] like what I call for in this issue.

[1] https://wiki.greasespot.net/GM_xmlhttpRequest

#3 - 08/03/2021 11:48 PM - jahoti

> BTW, we could also facilitate spoofing of the referer header for similar purposes

Are extensions allowed to spoof referer? It's definitely *needed*, I fully agree, just (at least for webpages) not *possible*.

> EDIT: GreaseMonkey actually has something[1] like what I call for in this issue.
> [1] https://wiki.greasespot.net/GM_xmlhttpRequest

That looks like a good model- an API object is a good idea in any case, and a more JQuery-like AJAX function would be hugely beneficial no matter what.

As an extra point, do we want to provide a way to limit the domains a script (or package or however the setting may work) can bypass CORS on? Is that even feasible?

#4 - 08/04/2021 10:19 AM - koszko

> BTW, we could also facilitate spoofing of the referer header for similar purposes

Are extensions allowed to spoof referer? It's definitely *needed*, I fully agree, just (at least for webpages) not *possible*.

Does WebRequest not allow rewriting of this header?
Also, perhaps we'd be able to spoof a Referer: https://example.com/ header by opening https://example.com/ in some hidden page or iframe?

> As an extra point, do we want to provide a way to limit the domains a script (or package or however the setting may work) can bypass CORS on? Is that even feasible?

Sure! Once we implement a permissions system, this is probably going to be the main use of it

#5 - 08/05/2021 11:32 AM - jahoti

> Does WebRequest not allow rewriting of [the referer] header?

WebRequest probably does actually; thanks for pointing that out!

> Also, perhaps we'd be able to spoof a Referer: https://example.com/ header by opening https://example.com/ in some hidden page or iframe?

That would definitely work- nevertheless, hopefully it isn't needed and using WebRequest is enough.

#6 - 08/05/2021 12:15 PM - koszko

> Also, perhaps we'd be able to spoof a Referer: https://example.com/ header by opening https://example.com/ in some hidden page or iframe?

> That would definitely work- nevertheless, hopefully it isn't needed and using WebRequest is enough.

We might use it as a workaround for manifest V3 Chromium port

#7 - 10/01/2021 04:28 PM - koszko

In case of important data only being available in external scripts (btw, I think this is the case with reCAPTCHA which, although nasty, is very important for us to support) instead of providing a general facility to bypass CORS (I am thinking of implementing some API similar to XMLHttpRequest), we could add a facility to fetch just the external scripts referenced by the original page. Better yet, we could actually add a general facility to bypass CORS and then make it possible to define payload's permissions in a way it is only allowed to bypass CORS for these particular scripts

#8 - 10/02/2021 04:05 AM - jahoti

> In case of important data only being available in external scripts (btw, I think this is the case with reCAPTCHA which, although nasty, is very important for us to support) instead of providing a general facility to bypass CORS (I am thinking of implementing some API similar to XMLHttpRequest), we could add a facility to fetch just the external scripts referenced by the original page. Better yet, we could actually add a general facility to bypass CORS and then make it possible to define payload's permissions in a way it is only allowed to bypass CORS for these particular scripts

That sounds like a good system- we could even use URL patterns to specify allowed connections, albeit with some way to specify multiple per script.

For reCAPTCHA I think the data that get extracted (maps from challenge code to displayed text) is constant at least within each reCAPTCHA version, which means it's possible (at least in theory) to hardcode them in the script and simply release a new version whenever Google does. Nevertheless, that obviously would not be ideal.

#9 - 10/02/2021 11:59 AM - koszko

> For reCAPTCHA I think the data that get extracted (maps from challenge code to displayed text) is constant at least within each reCAPTCHA version,

That'd explain why instead of something like "select all squares with stairs" I was presented with some JSON... Good that I guessed what to do

#10 - 02/24/2022 12:57 PM - koszko

- Subject changed from Make it possible for injected scripts to bypass CORS to [Roadmap 5][Milestone] Make it possible for injected scripts to bypass CORS

- Description updated

- Parent task deleted (#72)

#11 - 02/24/2022 12:58 PM - koszko

- Blocks Feature #107: [Roadmap 5] A user-controlled reCAPTCHA client library added

#12 - 02/24/2022 12:58 PM - koszko

- Blocks Feature #72: [Roadmap 18][Milestone] Facilitate creation of "meta-sites" added

#13 - 06/21/2022 02:32 PM - koszko

- Status changed from New to Closed

- % Done changed from 0 to 100

done

Here is a Haketilo library that implements a convenient fetch()-compatible function to work with this