bugs Project: chromium ▼ **Issues** People Development process History Sign in

New issue Search Open issues for Search Advanced search Search tips

Issue 896897

Starred by 60 users

Status: Assigned

Owner: rdevlin....@chromium.org

Cc: rob@robwu.nl
tsteiner@google.com
sime...@chromium.org
jawag@chromium.org
chrome-conops-escalation@…

Components: Platform>Extensions

EstimatedDays: ----

NextAction: ----

OS: Linux , Windows , Chrome , Mac

Pri: 2

Type: Bug

Hotlist-ConOps

Blocked on: View detail
issue 896041
issue 914224

Sign in to add a comment

Extensions: Implement Manifest V3

Project Member Reported by karandeepb@chromium.org, Oct 18

Description #2 (rdevlin....@chromium.org, Nov 17 2018)

This is the tracking bug for extension Manifest V3 implementation.

In Progress Design Doc: https://docs.google.com/document/d/1nPu6Wy4LWR66EFLeYInl3NzzhHzc-qnk4w4PX-0XMw8/edit#

Comment 1 by karandeepb@chromium.org, Oct 18

Blockedon: 896041
Owner: rdevlin....@chromium.org
Status: Assigned (was: Untriaged)

Assigning to you Devlin.

Project Member Comment 2 by bugdroid1@chromium.org, Oct 24

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src.git/+/066cb5ad04248407c0107fbbbf8f910ea7a24255

```
commit 066cb5ad04248407c0107fbbbf8f910ea7a24255
Author: Karan Bhatia <karandeepb@chromium.org>
Date: Wed Oct 24 03:42:22 2018

Extensions: Change restrictions on [min/max]_manifest_version.

- Allow manifest version 2 to be used as "max_manifest_version".
- Allow manifest version 3 to be used as "min_manifest_version".

BUG=896897

Change-Id: I72009c94bb715b7de3606b013d8e4eb5c94f44af
Reviewed-on: https://chromium-review.googlesource.com/c/1292903
Commit-Queue: Karan Bhatia <karandeepb@chromium.org>
Reviewed-by: Devlin <rdevlin.cronin@chromium.org>
Cr-Commit-Position: refs/heads/master@{#602248}
[modify] https://crrev.com/066cb5ad04248407c0107fbbbf8f910ea7a24255/tools/json_schema_compiler/feature_compiler.py
```

Comment 3 by ghuczyn...@gmail.com, Oct 25

Hi rdevlin@.

Is there a document/link to what will be in extension manifest v3?

Thanks

Comment 4 by rdevlin....@chromium.org, Oct 25

Thanks for reaching out! We have internal documents, but I'm working on compiling an external version that I will share on this bug (I'm hoping to have something by next week).

Note that this will be a design doc, and not a concrete guarantee of exactly what manifest v3 will entail - things may change (and if they do, we'll update docs/bugs accordingly).

Comment 5 by ghuczyn...@gmail.com, Oct 25

Looking forward to seeing the doc, and hopefully giving some feedback. Given manifest v3 was trailed in the recent "Trustworthy Chrome Extensions, by default" (https://blog.chromium.org/2018/10/trustworthy-chrome-extensions-by-default.html), it would be nice if extension developers got some insight into this.

Comment 6 by rdevlin....@chromium.org, Oct 26

Cc: rob@robwu.nl jawag@chromium.org

+Rob, who was also asking about this (Rob, see #4)

Comment 7 by rdevlin....@chromium.org, Nov 9

In Progress Design Doc: https://docs.google.com/document/d/1nPu6Wy4LWR66EFLeYInl3NzzhHzc-qnk4w4PX-0XMw8/edit#

Comment 8 by ghuczyn...@gmail.com, Nov 14

Hi rdevlin@

Quick question re manifest v3 document.

Re "Cross-Origin Communication" you say "Extension origins will continue to be able to make cross-origin requests to any sites they have permission to access".

Will it still be possible to request an API permission like: ["http://*/*", "https://*/*"]?

My vote for this would be yes, as it's sometimes not possible for an extension to specify all accessed origins up-front in the manifest file, particularly if it varies on a per-user basis.

For example, I have a webpage bookmarking extension where a user specifies a representative image when bookmarking a web-page: either selected from the webpage ala pinterest, or from webpage structured data. The user can view all bookmarks later from an extension-hosted sidebar iframe which contains the representative images. These images are loaded via xhr from the extension background page, downsized, and passed to the sidebar via messaging. Since the images vary on a per-user basis, I have to request http(s)://*/* permissions to cover the possibilities. There's also a precedent for loading cross-origin images with: https://developer.mozilla.org/en-US/docs/Web/HTML/CORS_enabled_image.

Other extensions that would also use such https(s)://*/* API permissions are ones that are configured on a per-user basis, such as an email-account-checker (user specifies email-server), or an rss-checker (user specifies different feeds).

Comment 9 by woxxom@gmail.com, Nov 16

re Main World injection from a content script, the design doc says it's bad for users and web developers.
I think it's a biased and skewed point of view, probably based on some malicious extensions.

* Accessing page variables is an important feature that allows extensions to augment sites thus enriching UX,
  Firefox even provides a simple direct access via window.wrappedJSObject
  https://developer.mozilla.org/docs/Mozilla/Add-ons/WebExtensions/Sharing_objects_with_page_scripts

* Extensions should be able to augment/extend/hide/override/limit web APIs of a site,
  this is a crucial feature for users who want to have more

control over security and privacy

Comment 10 by rdevlin....@chromium.org, Nov 17

Description: Show this description

Comment 11 by rdevlin....@chromium.org, Nov 17

Thanks for the input, folks!

ghuczynski@: Yes, extensions will still be able to request wildcard hosts for fetch()/XHR. They will simply have to make the request from the background page, rather than a content script. (Note that with runtime host permissions ["Restricting Origin Access" section]), the user may limit which permissions are granted. But as long as the user has approved the permission, these requests are still possible.

woxxom@: When the design doc says "This type of mutation is bad for web developers (who have to deal with it) and bad for users (because developers have to find workarounds, which often come with performance costs, or don't find workarounds, and websites are broken)", it's specifically referring to the extreme cases of e.g. an extension overriding the Array.prototype (which we have seen before, and is something that no one writing JS should ever have to worry about :)). I absolutely agree that there *are* valid reasons to inject in the main world. If we were to make a change there, it would mostly be targeting reducing the likelihood of truly destructive interaction, rather than targeting removing all interaction. Note also that we aren't currently planning on pursuing that (it's in the "Declined Changes" section). I'll also think about changing the phrasing there to make it more clear.

Comment 12 by blazetod...@gmail.com, Nov 22

One gap in service workers vs the existing background page paradigm is that background pages have access to a full DOM they can manipulate while service workers do not.

We use our background script's webpage in order to copy/paste plain and styled text to the clipboard in our extension. I am sure there are probably other use cases for needing access a DOM that other extensions might have.

If this background page DOM went away, I am not sure how we would be able to recreate that functionality. We would probably need to inject elements via the content script into the users page in order to orchestrate the copy/pasting to/from the clipboard. This would be more complicated and increase risks of conflicts. Or maybe we would just open up a new page when we wanted to copy and paste, manipulate it and then close it quickly. But that could lead to flickering and a poor experience.

Related to this I notice that the the clipboard read/write permissions are also potentially on the chopping block. I will look at the current web capabilities on this, but I am not sure that they would be sufficient to replicate what Chrome extensions can currently do.

Comment 13 by aaron.qu...@usaa.com, Dec 9

The extension that we have heavily uses the webrequest API to add and remove cookies for many internal applications. How is this going to be impacted going forward? The suggestion to use declarativeNetRequest is not applicable to our situation.

Also, what is the plan for having Chrome run in the background? We use the persistence flag to keep the extension running even when no Chrome window is up. Will that functionality still exist?

Comment 14 by karandeepb@chromium.org, Dec 12

Blockedon: 914224

Comment 15 by tsteiner@google.com, Dec 12

Cc: tsteiner@google.com

Comment 16 by sime...@chromium.org, Dec 20

Cc: sime...@chromium.org

Comment 17 by woxxom@gmail.com, Jan 12

The current V3 plan for webRequest API and its replacements will totally obliterate the advanced dynamic resource managers like uMatrix and uBlock, as well as many other advanced consumers of these API. Either it was a glaring oversight by those who designed the plan, which hopefully could be fixed, or it was intentional to not care about the "fringe" cases (arguably 99% of Chrome users wouldn't notice the difference) and leave this niche to the competition e.g. Firefox which already provides multiple enhancements of the extensions API compared to Chrome.

Comment 18 by mexmat.s...@gmail.com, Jan 13

My concern with manifest V3 is playing sounds from an extension (e.g. a sound alert).

Current implementation is to create an <audio> element and play that from the background. However, if no DOM is available, this will no longer work.

Is there a workaround for this?

Comment 19 by thomasga...@gmail.com, Jan 14

I also have many concerns about the V3 API as proposed, at least as I understand it. I've attached a document with some of them set out. If I've many any errors in understanding I'd appreciate a correction.

V3ExtensionsManifest.txt
8.2 KB View Download

Comment 20 by sscar...@gmail.com, Yesterday (31 hours ago)

@woxxom, I doubt they would fix this, the way it's written in the doc, this was intentional and they're bringing privacy issues to justify it which is the opposite of what uBlock and uMatrix does.

Comment 21 Deleted

Comment 22 by zombull...@gmail.com, Today (17 hours ago)

This doesn't surprise me in the least, since Edge jumped off ship and now Google (an advertising company) as the sole captain of the ship, this step of "progression" really shouldn't surprise anyone.

At least, for the time being, there's still the "Firefox" option.

Comment 23 by rh...@raymondhill.net, Today (16 hours ago)

In the design document, it is said that the webRequest API will no longer allow to be used in blocking mode:

> In Manifest V3, we will strive to limit the blocking version
> of webRequest, potentially removing blocking options from most
> events (making them observational only). Content blockers should

> instead use declarativeNetRequest (see below). It is unlikely
> this will account for 100% of use cases (e.g., onAuthRequired),
> so we will likely need to retain webRequest functionality in
> some form.

From the description of the declarativeNetRequest API[1], I understand that its purpose is to merely enforce Adblock Plus ("ABP")-compatible filtering capabilities[2]. It shares the same basic filtering syntax: double-pipe to anchor to hostname, single pipe to anchor to start or end of URL, caret as a special placeholder, and so on. The described matching algorithm is exactly that of a ABP-like filtering engine.

If this (quite limited) declarativeNetRequest API ends up being the only way content blockers can accomplish their duty, this essentially means that two content blockers I have maintained for years, uBlock Origin ("uBO") and uMatrix, can no longer exist.

Beside causing uBO and uMatrix to no longer be able to exist, it's really concerning that the proposed declarativeNetRequest API will make it impossible to come up with new and novel filtering engine designs, as the declarativeNetRequest API is no more than the implementation of one specific filtering engine, and a rather limited one (the 30,000 limit is not sufficient to enforce the famous EasyList alone).

Key portions of uBlock Origin[3] and all of uMatrix[4] use a different matching algorithm than that of the declarativeNetRequest API. Block/allow rules are enforced according to their *specificity*, whereas block/allow rules can override each others with no limit. This cannot be translated into a declarativeNetRequest API (assuming a 30,000 entries limit would not be a crippling limitation in itself).

There are other features (which I understand are appreciated by many users) which can't be implemented with the declarativeNetRequest API, for examples, the blocking of media element which are larger than a set size, the disabling of JavaScript execution through the injection of CSP directives, the removal of outgoing Cookie headers, etc. -- and all of these can be set to override a less specific setting, i.e. one could choose to globally block large media elements, but allow them on a few specific sites, and so on still be able to override these rules with ever more specific rules.

Extensions act on behalf of users, they add capabilities to a *user agent*, and deprecating the blocking ability of the webRequest API will essentially decrease the level of user agency in Chromium, to the benefit of web sites which obviously would be happy to have the last word in what resources their pages can fetch/execute/render.

With such a limited declarativeNetRequest API and the deprecation of blocking ability of the webRequest API, I am skeptical "user agent" will still be a proper category to classify Chromium.

---

[1] https://developer.chrome.com/extensions/declarativeNetRequest

[2] https://adblockplus.org/filter-cheatsheet

[3] https://github.com/gorhill/uBlock

[4] https://github.com/gorhill/uMatrix

Comment 24 by craigtumblison@chromium.org, Today (5 hours ago)

Labels: Hotlist-ConOps

Comment 25 by demonsta...@gmail.com, Today (4 hours ago)

I really do not wish to jump ship back to Firefox, please reconsider changes that would end up breaking uBlock. Because I will be forced to jump ship if they break.

Comment 26 by aakash.x...@gmail.com, Today (3 hours ago)

Time to fork chromium

Comment 27 by jackcodi...@gmail.com, Today (3 hours ago)

I have an extension in use by a small number of users (~5700) which modifies the response headers on specific web requests to insert a CORS header. Would this no longer be possible?

If this were to break the extension the majority of users would likely switch to Firefox which isn't something I wish to see happen.

Assuming the above is true then I am a +1 for the please reconsider vote.

Comment 28 Deleted

Comment 29 by kc0...@gmail.com, Today (3 hours ago)

I'd like to add a vote to the "don't break uBlock Origin or other ad blocking extensions" camp. I believe very, very strongly in maintaining my ability to use ad blocking software on my browser, and I will switch myself to another browser to maintain that capability if required.
I will also switch everyone I support on a technical basis, and begin blocking Google's ads on a DNS level for not only my personal network but also the networks I manage at work. Up until now we've mostly turned a blind eye to ads, since it wasn't worth convincing executives that they should greenlight DNS filtering and it helps to pay for the products we all use in our personal time, but if Chromium and Google begin actively working to subvert user choice in this manner, my team will be much more incentivized to figure out a less-targeted solution than an ad blocker.
I urge the Chromium team to reconsider. I know many of the developers working on this team are interested in building a better browser and providing a better user experience; this, however, will not further those goals.

Comment 30 by xopxopx...@gmail.com, Today (3 hours ago)

If you haven't already, please switch your browser.

Comment 31 by pixus...@gmail.com, Today (3 hours ago)

I recommend Google Chrome developers to look into adding a limited virtual machine for filters like eBPF[1] with constrained execution time and resources.

This will address valid problem of browser extensions holding a request for indefinite amount of time, at the same time it will give extensions a flexibility to make filtering by any criteria imaginable.

[1] - https://opensource.com/article/17/9/intro-ebpf

Comment 32 by ay.mesh...@gmail.com, Today (2 hours ago)

Hi, I am another ad blocker developer (AdGuard), and from our perspective, the proposed change will be even more crippling to all ad blockers than what was done by Apple when they introduced their declarative content blocking API.

I agree with the points Raymond made in comment 23, but there's another serious change that needs attention. The proposed change to hosts permissions (either using activeTab or requesting access on every new website) basically means that every time users navigate to a new website, nothing is blocked there. Ok, maybe something is blocked by declarative rules, but blocking web requests is just a tiny part of what ad blockers do. For instance, they need to apply cosmetic rules and that's roughly half of EasyList rules.

Comment 33 by rdevlin....@chromium.org, Today (2 hours ago)

Hi folks!

Thank you very much for the feedback here.

First off, a friendly reminder to keep discussions both respectful and constructive. If this thread gets too noisy with comments not related to this design discussion, I'll have to periodically trim out some comments.

Unfortunately, neither this bug nor comments on the doc are an appropriately scalable place for these discussions. For future comments, feedback, etc, can we move discussions to take place on chromium-extensions@chromium.org? To make them easier to track, consider prefixing with something like "Manifest V3", e.g. "Manifest V3: Web Request Changes". Feel free to cc me directly on messages, and I'll try to keep up with them.

Authors of comments 12, 19, 23, 32, and anyone else that would like to: Sorry for the trouble, but would you mind re-posting your comments there (chromium-extensions@chromium.org), where we can kick off a larger discussion? These all touch on issues that I'd like to address more fully than is feasible here.

Comment 34 by regal...@gmail.com, Today (2 hours ago)

I'm the author of an extension that needs to add an outbound header for it to work. This sounds like it would break my extension, no?

If the declarative request api supports this, can it be changed at runtime? A static file won't fit my needs.

Comment 35 by pixus...@gmail.com, Today (2 hours ago)

Here is link to chromium-extensions mail list Devlin suggested for further discussion - https://groups.google.com/a/chromium.org/forum/#!topic/chromium-extensions/veJy9uAwS00

Comment 36 by entrance...@gmail.com, Today (2 hours ago)

I think it's in everyone's best interest to not do this, or just let a globbing so that extensions are still capable of controlling these sorts of things at the user's peril.

Comment 37 Deleted

Comment 38 Deleted

Comment 39 by netheri...@gmail.com, Today (72 minutes ago)

Safari has introduced a similar API, which I guess inspires this. My personal experience is that extensions written in that API is usable, but far inferior to the full power of uBlock Origin. I don't want to see this API to be the sole future.

By the way, the biggest downside is the limit on number of rules, while I may tolerate the loss of advanced filtering rules. Safari has the limit of 50,000, larger than the one proposed here, and it never suffices for me.

Comment 40 by kelenchi...@gmail.com, Today (65 minutes ago)

The

chrome.declarativeNetRequest.MAX_NUMBER_OF_RULES
shoud be at least Ten times larger(around 300,000 or more).

A similar limit(about 50,000, see #Comment 39) in Apple's Safari has been proven to be insufficient to hold the essential rules.

If content block extentions' performance is going to be restricted according to #Comment 23, I would have to switch to an alternative browser like Firefox at the time.

Comment 41 by fbol...@gmail.com, Today (65 minutes ago)

Concerning the blocking system that allows extensions to veto webRequest, do I understand correctly that the rationale this proposal provides for its deprecation is that, because it exists, therefore it has to run, and thus slows down every requests?

And in order to fix this problem, they will make a new non-blocking system to veto webRequest; but this one will not slow down every request, even though it also exists and has to run?

Or is the rationale only about the fact that a blocking no-op is slower than a non-blocking no-op, and this is purely a judgement based on speed? I hope that is not the case, for I (and many others apparently) don't value speed over control. Besides, Chromium is fast already.

Comment 42 by walde.ch...@gmail.com, Today (64 minutes ago)

@rdevlin: One should only ask for respect when one gives respect. Someone who virtually declares war on the entirety of the world for the sake of one's wallet should be ready to handle consequences.

Comment 43 Deleted

Comment 44 by dreadhaw...@gmail.com, Today (55 minutes ago)

If adblocking becomes infeasible, I'm afraid I'd have to switch browsers.

Comment 45 by slayerof...@gmail.com, Today (43 minutes ago)

@rdevlin

I suspect people are going to want to comment here in public view rather than to an email address, and given this change and this thread in particular are now highlighted and linked on sites like SlashDot and The Register I'd expect you are going to see an influx of unhappy people.

My 2 cents, if you don't delete this post, is that this is a terrible idea. Speed is good, but speed at the cost of breaking essential plugins isn't okay.

▶ Sign in to add a comment