Dear Chrome team,

Thank you for requesting feedback on the proposed changes for Manifest V3. Below, we've listed a set of changes to the proposal that we believe are necessary to keep Privacy Badger functioning as it does now and to allow us to implement planned changes in the near future. However, we still believe that the proposed changes in Manifest V3 will hamper extension developers' ability to innovate and adapt to a changing tracking landscape in the future, and continue to urge you to reconsider deprecating the blocking webRequest API.

Our comments are based on the Manifest V3 design document (as of April 11, 2019), the declarativeNetRequest API documentation (also as of April 11), and public comments made by members of the team on the mailing list (e.g. https://groups.google.com/a/chromium.org/forum/#!msg/chromium-extensions/WcZ42Iqon_M/lT-Y5aZiAwAJ). We apologize for the gaps in our knowledge of the state of the proposal. If we have misinterpreted anything, or if portions of our comments have already been addressed on the mailing list or elsewhere, please let us know.

Permissions

The design document states that it is the team's intent to deprecate the <all_urls> permission and similar blanket permissions. It elaborates with the following:

> In Manifest V3, we want activeTab-style host permissions to be the default, with a number of extra options. Instead of being granted access to all URLs on installation, extensions will be unable to request <all_urls>, and instead the user can choose to invoke the extension on certain websites, like they would with activeTab. Additional settings will be available to the user post-installation, to allow them to tweak behavior if they so desire.

Privacy Badger, and other ad- and tracker-blockers, depend on blanket permissions like <all_urls> to function. There is no practical way to build a blocking extension in the activeTab model; the burden on users to grant consent for each and every site they visit would be far too great, and all but the most privacy-conscious users would uninstall the extensions.

The document mentions that users will be able to tweak "additional settings" affecting this behavior "post-installation." We interpret this to mean that Chrome will provide some kind of option for users to grant <all_urls>-like permissions to extensions after they've been installed. This would be preferable to an activeTab-only model, but still far from optimal.

The document pitches the benefits of this model as follows:

> Finally, this avoids giving the users an ultimatum. Currently, we force users to accept all permissions and install the extension, or accept none and refuse the extension.

However, there is simply no way for Privacy Badger or other blocking extensions to operate *without* <all_urls> or similar permissions, so treating <all_urls> as an “optional” permission would be disingenuous. Furthermore, defaults matter. Every time a user needs to jump through an additional hoop, find a non-default setting, or click though a consent dialog to enable a piece of software’s intended functionality, the probability that the user will just give up (and uninstall the software) increases dramatically. Privacy Badger and other privacy-conscious extensions seek to make privacy the default to the user. This change removes the choice of privacy by default for all sites.

Recently, EFF experienced this phenomenon first-hand when we made a small change to the permissions required by HTTPS Everywhere in Firefox. The extension already had permissions to access and modify requests to “http://*” and “https://*”, which prompted users to grant it access to “all data on all sites” when they installed it. Last fall, we updated the extension to require permissions for “ftp://*” as well. This forced all current users of HTTPS Everywhere to grant the extension access to “all data on all sites” once again in order to receive the update. Unfortunately, the permissions dialog did not provide any additional context for the request; as a result, rather than agree to re-grant permissions they had already granted, millions of users chose not to update the extension at all. At the same time, the extension received several negative reviews in the Firefox extension marketplace accusing it of harvesting user data for nefarious purposes.

Our point is not that consent dialogs should try to avoid scaring users. Privacy Badger does, indeed, require permissions for all user data on all sites, and users should be informed of the associated risks. However, they should also have the chance to give full consent at the moment they decide to install the extension. When a dramatic permissions dialog is surfaced outside of the context of a user’s original decision, it leads to confusion and frustration. Privacy Badger requires <all_urls> to operate; it is impossible to build an extension that achieves its goals otherwise. Users should have the chance to opt in to its full functionality in one fell swoop.

Observation

Privacy Badger is a heuristic tracker blocker. It observes network traffic from a user’s browser to learn which third-party domains are likely to be tracking the user, then blocks them. Therefore, the ability to monitor network traffic is absolutely essential for Privacy Badger to function. In a developer update, the team wrote that “there are no planned changes to the observational capabilities of webRequest.” We interpret this to mean that under Manifest V3, extensions will still be able to inspect URLs, GET and POST parameters, and headers (including cookies) for all requests and responses. If this isn’t the case, Privacy Badger will be unable to perform its primary function.

Improvements to the declarativeNetRequest API

We continue to believe that the only way to ensure heuristic blocking extensions remain viable in the long term is with the ability to intercept, inspect, and modify requests with arbitrary code at runtime, as the webRequest API currently allows. However, here we suggest a set of improvements to the declarativeNetRequest (DNR) API that we believe will allow Privacy Badger to continue protecting users from tracking in the short term.

Header modification: Privacy Badger modifies headers in a number of ways to mitigate tracking. Most simply, it sets the `DNT=1` header on every request to indicate its user's intent to opt out of tracking. It removes cookies and obfuscates the `Referer` header from requests to any domain that it is configured to "cookieblock." In the future, we may choose to modify other headers in order to protect privacy, including `User-Agent` and `ETag`.

Currently, DNR has no way to modify headers of any kind, making all of the aforementioned actions impossible. As headers are a potent channel for information of all kinds, including various kinds of tracking mechanisms, we ask that declarativeNetRequest allow modification of all headers, not just a predefined subset. This should include the addition of headers not already present in a request (like `DNT`). Furthermore, DNR should allow extensions to use other information about a request in order to modify headers. For example, we may want to rewrite the `Referer` header to set it equal to the destination domain of a third-party request, in order to protect user privacy without breaking plugins that rely on a valid Referer.

URL parameter modification: In addition to blocking and cookieblocking third-party requests, Privacy Badger sometimes modifies URL parameters in order to mitigate tracking. In the simplest case, this may involve removing `fbclid`, `gclid`, and `utm_*` URL parameters. In more complex cases, it may involve identifying and removing sensitive strings, such as phone numbers, social security numbers, or identifying cookies, from the parameters. DNR should support rules which match, modify, and delete URL parameters using regular expressions. Furthermore, declarativeNetRequest should allow the modification and deletion of POST parameters as well as GET parameters.

Contextual redirects: Privacy Badger may also need to perform more sophisticated rewrites (which we have not implemented, but plan to in the future). One possible use-case involves removing unnecessary intermediary redirects, such as rewriting "https://www.google.com/url?q=https://example.com" to "https://example.com". See https://groups.google.com/a/chromium.org/d/msg/chromium-extensions/MH9Yj9nHtvc/-ZrBAxUFBQAJ for other examples. Currently, this style of redirection is impossible with static DNR redirect rules. DNR should allow regular-expression based redirects.

We hope this is a useful starting point. Please let us know if you would like any clarifications or more detail about any of these requests.

Best,

Bennett Cyphers, Staff Technologist

Andrés Arrieta, Director of Consumer Privacy Engineering