Reexamining Consent

Mozilla position paper for the W3C Workshop on Permissions and User Consent.

Contact: Martin Thomson, Marcos Caceres.

Introduction

The strength of the web lies in its ability to form casual interactions. The freedom of movement that the hyperlink enables is only realized because browsers make following links safe. That it is safe to follow any link is the foundation of trust in the web as an application and communications platform.

The role of informed consent in ensuring that the web remains safe is complex. The consequences of a choice can be complex. In seeking to make powerful capabilities available to the web, consent is often a poor means of controlling access.

A browser typically requests consent because of the potential for negative consequences. In asking a question, the goal should be to ensure that a person understands and acknowledges all potential negative consequences. That is, should a negative outcome arise, an individual would recognize that they acceded to this possibility, even if this represents a violation of trust by those involved.

This paper explores central concepts in consent and proposes four concepts that we believe should govern the development of new permissions interactions. Then we explore the use of permissions using those principles. Finally, we explore some trends in the use of permissions and their relation to these central concepts.

Concepts

The notion of informed consent has a rich history in academic literature, being critical to the both the practice and research of medicine. Though intended for use in medical research, the Belmont Report describes a useful three-part taxonomy for **informed consent**:

- **Information**: The disclosure of information about what is being consented to. The information provided needs to be complete.
- **Comprehension**: An assessment of whether the disclosed information has been understood.
- **Voluntariness**: This concept encompasses both competency (the right to make the decision) and freedom from coercion.

We'll examine these concepts in relation to the role of consent on the web. Web features are used to illustrate the limitations and shortcomings of informed consent.

We also argue that informed consent, though a necessary part of any permissions interaction, is not sufficient to ensure continuing trust in the web. We propose that—in addition to informed consent—**accountability** is a necessary additional condition for enabling features.

Sites with access to capabilities beyond those granted to other sites need to remain accountable for the actions they take. With a permissions model that assumes a persistent grant of capabilities, accountability means two things: providing ways in which access can be monitored, and providing effective means of revocation.

Both informed consent and accountability need to be considered in assessing, designing, and standardizing new features.

Information and Comprehension

The ability to present complete and accurate information can be challenging. What is often simple at face value can hold complexity that is hard to communicate effectively.

For instance, APIs that enable access to a microphone or screen sharing seem fairly understandable at face value. The obvious risks can mask other more subtle problems. For instance, the capture of ultrasonics can be used as a covert channel between machines in the same room. Or maybe the audio from fans can be recovered and used to measure the room temperature and CPU load. Screen sharing potentially allows sites to access information on other sites that would not otherwise be available to them and in ways that those sites might not be prepared to defend against.

The same story plays out in many ways. The use of a battery status API for tracking, the recovery of passwords using an accelerometer API, the use of an ambient light sensors to recover cross-origin content. In some of these cases, the risks were only discovered after independent research on the subject. Expecting that information is both available and understood has been shown to be a particularly difficult problem. When it is hard for browser vendors to understand the full ramifications of features, expecting non-experts to understand and make the right decisions might be unreasonable.

Voluntariness

The role of a user agent in ensuring that permissions are voluntary is a difficult problem for user experience designers.

The Norwegian Consumer Council (Forbrukerrådet) report on consent as it pertains to use on a number of major sites highlights the risks inherent in obtaining informed consent. The report highlights several ways in which subtle coercion might be used to influence outcomes. Special care is needed in framing of questions, the language used, and the default option.

In many cases, permission to access a capability is not explicit. For instance, the ability to open a new window or access a fullscreen display is often implicitly granted in response to a user gesture (a mouse click, tap, or keypress). This sort of inference is critical to the functioning of a modern browser. Browsers limit the capabilities that are implicitly granted in this fashion; this sort of access is often limited to capabilities that are short-lived or easily revoked.

Accountability

The web overall has excellent accountability characteristics. Sites gain access to a window on which they are able to render literally anything. Sites can execute script within a sandbox, access the network under certain restrictions, and accept user input that is directed at them. When that window is closed, all these capabilities are revoked and the site no longer has any further ability to operate.

Of course, this is a gross simplification. Even though access to capabilities beyond this are now generally carefully circumscribed. A more thorough inventory of the capabilities of the platform and their access control mechanisms is maintained by Jeffrey Yaskin.

Ensuring that a site remains accountable for the features that it uses can be difficult. For instance, Service Workers can operate beyond the time that a site remains open.

Case Studies

Geolocation

Geolocation was one of the first features to be gated behind a permissions prompt, if not the first. Here, the information needed to make a good decision is clear and concise, removing concerns about comprehension. The non-modal interaction model (on desktop at least) and UX paradigms chosen by various browsers avoids the worst questions about coercion.

In terms of accountability, browsers differ in how they display active access to the API (e.g., Privacy of Geolocation Implementations). Whether an indicator is shown for sites that access the API varies. However, all tested browsers provide a means of revoking access in the site control panel (the dialog that is usually accessed by clicking on the lock icon).

Thus, geolocation might be held up as the gold standard in terms of informed consent and accountability. However, this is not necessarily the entire story. Acceptance rates for geolocation are dramatically lower than for other APIs. This is partly reflective of widespread poor practice on the part of sites that prompt immediately on first impression. More research will be necessary to understand the underlying reasons.

Opening Windows

The ability to open a new window was one of the first web capabilities to see abuse and restriction. The misuse of the `window.open()` API for various purposes had a serious impact on the trust in the web. Without constraints on its use, this API could rapidly render a machine unusable.

Browsers introduced "popup blocking", which is generally a simple requirement to restrict calls of window.open to one call for each engagement gesture. This can produce an experience similar to the permissions prompt if a popup is blocked.

Audio Playback

The story of audio playback is similar to that of popups. Abuse of the capability has lead to browsers producing better accountability in the form of indicators, but also with the ability to mute a page.

This followed the "ask for forgiveness" approach, where permissions are granted by default, but with effective means of revoking that access. However, that was not proven to be effective and so we have seen further restrictions on playback, particularly at the time that a page loads.

Recent Trends

This area continues to evolve rapidly. Here are a few developments that should be considered.

Feature Policy

The introduction of Feature Policy represents a simplification of the permissions model. Today, the browsing context that requests a permission could be any origin. Though the top-level browsing context shows one origin, the origin that requests access to an API that requires consent might be different. This presents a challenge for browsers in effectively communicating what is going on, and a pretty significant risk to comprehension.

Feature Policy makes the top-level browsing context responsible for all permissions. If framed contexts want access to capabilities, the top-level context is the one that acquires permission. The top-level context is then responsible for passing that capability to the framed context.

There will be challenges for capabilities that are currently available in framed contexts, but this change is a generally positive one.

(Feature Policy has a CSP-style policy language that is currently coupled to this important change. That aspect of the proposal is less clearly positive.)

Permission Bundling

Some features that depend on consent, like geolocation, are often clear and crisp. However, new features are emerging that are challenging to explain.

Notifications permissions are commonly interpreted as allowing access to the Push API. On face value, this seems OK, but it means that sites are able to send push notifications to idle browsers without any visible effect. Browsers are still experimenting with the right way to ensure accountability for this API. The more promising ideas concentrate on having access expire in different ways with a visit to the site refreshing the access.

WebRTC decided that it was essentially impossible to explain the consequences of its use of IP addresses and so they decided to enable use of the more privacy-sensitive addresses when getUserMedia permissions are granted. The operating principle seemingly that the incremental damage to privacy is negligible.

The consequence of this sort of bundling is that it can violate expectations. It is surprising when the capabilities that a site has access to change in non-obvious ways. These expectations are built around the permissions that are requested, how those permissions are used in practice, and the feedback provided about use of permissions. For instance, it might be reasonable to assume that geolocation permissions are only available while a site is active; extending that permission to include access to that information from a Service Worker would change what is understood to be the scope of that permission.

Accountability for CPU Access

Browsers are lacking accountability measures for when sites abuse access to computational resources. Historically, this was limited to sites that ran infinite loops, which could cause a browser to become completely unresponsive. However, this is really only effective in exposing bugs. Consistent and sustained CPU (or GPU) access can be hard to attribute to a particular site.

Recent trends to limit and reduce the priority of access to the CPU for background and framed browsing contexts is a positive development. As browsers move to more granular process isolation techniques, it might become easier to provide more detailed information on badly behaving sites.

Conclusions

To make the web more capable risks that which is the web's most significant advantage: trust.

We cannot assume that the web needs every new feature - or every feature found in competing platforms. Using consent as a means of offloading responsibility for the safety of features is irresponsible, and should be considered a bad practice. In deploying a feature

to the web, browsers need to do everything they can to guarantee that the consequences of a bad choice are limited and can be observed and stopped - or reversed if possible.

This doesn't mean that it isn't possible to ship new, powerful capabilities, but it means that the bar is higher. Where consent is used as a way to control access to these features, it is necessary to take care to ensure that consent is informed.

Consent is a subtle tool, and we would be wise not to overuse it or rely on it too heavily.

For Discussion and Investigation

This paper covers a number of topics, but does not address the entire spectrum of subjects. We hope to continue to have discussions on other questions, such as:

- How do browsers store and manage permissions? The duration and scope over which a permission grant is given is not well understood. A comprehensive description of how permissions are managed in browsers might be valuable. This is not meant to imply that browsers would be required to apply consistent policies.
- Does it matter if permissions are deferred to a separate API (i.e., what's the scope of the Permissions API - why things like .revoke() might be problematic?) - or should permissions be requested by each individual feature?
- The safeguards user agents apply to different platform features differs based on the capability. Consistent use of recognizably similar safeguards has value, if only because a consistent design language reduces the potential for problems that misunderstanding might produce.