

Disabling hardware noise suppression



By Oskar Sundbom

Software Engineer working on WebRTC and Audio in Chromium

In Chrome 64 we're trying a new behavior for `getUserMedia` audio streams that have the `echoCancellation` constraint enabled. What's new is that such streams will temporarily disable hardware noise suppression for the duration of the stream. We anticipate this will make the echo canceller perform better. As this functionality is experimental, it needs to be explicitly turned on; [see below](#).

At this point, this behavior is only supported for certain input devices and only on macOS. Support is limited to devices which have toggleable “ambient noise reduction” in the *Sound* panel of *System Preferences*.

Background

An echo canceller tries to remove any sound played out on the speakers from the audio signal that's picked up by the microphone. Without this, what you're saying as one party of a call, will be picked up by the microphone of the other parties and then sent back to you. You'll hear an echo of yourself!

To be successful in removing echo, WebRTC's echo canceller (which is used in Chrome) needs to get as clean an audio signal as possible from the microphone. Processing that's applied *before* the audio reaches the echo canceller, such as hardware noise suppression, will normally impede its performance. Moreover, there is already software noise suppression in place, but only *after* the echo canceller has done its processing.

Details of the new behavior

Web developers can enable the new behavior on their sites by opting in to an [Origin Trial](#) [↗](#). End users can enable it globally by passing a command-line flag when starting Chrome. For more information, [see below](#).

When this is enabled, and a web page calls `getUserMedia` to get audio from an input device, the following happens:


- If the `echoCancellation` constraint is enabled, hardware noise suppression will be turned off for the duration of the newly created audio stream.
- Since this setting is system-wide this will apply to all audio input streams from the same device (i.e. the same microphone).
- Once the last stream that wants hardware noise suppression turned off closes, hardware noise suppression is turned back on.
- If hardware noise suppression was already disabled beforehand, Chrome will not change its state.
- If `getUserMedia` is called without `echoCancellation` enabled, Chrome will not touch hardware noise suppression.

As this setting is also user-controllable, there are some specific interactions with the user:

- If Chrome has turned hardware noise suppression off, and the user turns it back on, Chrome will not attempt to disable it again for that stream.
- If Chrome has turned hardware noise suppression off, and the user turns it back on, then off again, Chrome will still re-enable it once the stream ends.

The behavior takes effect by simply enabling the experiment. There are no API changes necessary.

How to enable the experiment

To get this new behavior on your site, you need to be [signed up](#)  for the "Disable Hardware Noise Suppression" [Origin Trial](#). If you just want to try it out locally, it can also be enabled on the command line:


```
chrome --enable-blink-features=DisableHardwareNoiseSuppression
```



Passing this flag on the command-line enables the feature globally for the current session.

There are a couple of aspects we wish to evaluate with this experiment:

- Qualitative differences, in the field, between having hardware noise suppression turned on vs. off.
- How does changing this setting from within Chrome affect the end user and other software they may be running?

We are interested in feedback on both of these aspects. Are calls better or worse with this feature turned on? Are there problems with the implementation that causes unexpected behaviors? In any case, if you're trying this out, please file feedback on [this bug](#) . If possible, include what microphone / headset / etc. was used and if it supports ambient noise reduction. If doing more large-scale experiments, links to comparative statistics on audio call quality are appreciated.

Except as otherwise noted, the content of this page is licensed under the [Creative Commons Attribution 3.0 License](#), and code samples are licensed under the [Apache 2.0 License](#). For details, see our [Site Policies](#). Java is a registered trademark of Oracle and/or its affiliates.

Last updated July 2, 2018.