Starred by 5 users

Owner:	iclel...@chromium.org
CC:	a...@google.com mkwst@chromium.org engedy@chromium.org iclel...@chromium.org ekaramad@chromium.org hkamila@chromium.org
Status:	Fixed *(Closed)*
Components:	Blink>FeaturePolicy Blink>SecurityFeature>IFrameSandbox
Modified:	Feb 9, 2021
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	----
OS:	Linux, Android, Windows, Chrome, Mac
Pri:	2
Type:	Bug-Security

reward-ineligible
Security_Severity-Low
Security_Impact-Stable
Arch-x86_64
Arch-All
allpublic
Via-Wizard-Security
CVE_description-submitted
Target-77
Target-78
Target-79
Target-80
Target-81
Target-84
M-86
Target-83
Target-85
Target-86
Release-0-M88
CVE-2021-21139

Issue 937131: Feature Policy 'allow' attribute can override top-level policy in frames

Reported by jun.k...@microsoft.com on Thu, Feb 28, 2019, 8:24 PM EST Project Member

Code

UserAgent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3709.0 Safari/537.36 Edg/74.1.85.0

Steps to reproduce the problem:
1. Go to https://test.shhnjk.com/FP.php?allow=geolocation%20%27self%27&xss=%3Ciframe%20src=%22https://shhnjk.azurewebsites.net/geo.html%22%20allow=%22geolocation%22%3E
2. Allow location prompt

What is the expected behavior?
Cross-origin iframe does not get geolocation permission because top-level window configured feature policy to be "geolocation 'self'"

What went wrong?
Cross-origin iframe can get permission as long as iframe has allow attribute with right permission. Top-level window's feature policy is basically ignored for all the cases except, when policy is set to 'none'.

Did this work before? N/A

Chrome version: 72 Channel: stable
OS Version: 10.0
Flash Version:

Comment 1 by tkent@chromium.org on Fri, Mar 1, 2019, 3:53 AM EST Project Member

Components: Blink>FeaturePolicy

Comment 2 by jun.k...@microsoft.com on Fri, Mar 1, 2019, 12:43 PM EST Project Member

While this is documented (https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy/geolocation#With_an_%3Ciframe%3E_element), I'm not sure how website will be able to protect against HTML injection.

For example, following website has HTML injection (attached source), but XSS is mitigated by CSP. This websites also sets geolocation to 'self' so that only same-origin frame will be able to access geolocation. But basically, if you allow any website in CSP frame-src, that website will be able to inherit permission from top-level browsing context (due to Permission Delegation).

https://test.shhnjk.com/self.php?xss=%3Ciframe%20src=%22//shhnjk.azurewebsites.net/xssable.php?xss=%3Cscript%3Enavigator.geolocation.getCurrentPosition(function(position)%20{%20alert(position.coords.latitude);%20});%3C/script%3E%22%20allow=%22geolocation%22%3E

So feature policy implementation suggests:
"If you have third party website in CSP frame-src, you are responsible for XSS in their website". And this doesn't make sense to me.

self.php
219 bytes View Download

Comment 3 by rsesek@chromium.org on Fri, Mar 1, 2019, 7:38 PM EST Project Member

Status: Assigned (was: Unconfirmed)
Owner: ekaramad@chromium.org
Cc: iclel...@chromium.org
Labels: OS-Android OS-Chrome OS-Linux OS-Mac
Components: Blink>SecurityFeature>IFrameSandbox

ekaramad: Can you take a look?

Comment 4 by ekaramad@chromium.org on Fri, Mar 1, 2019, 8:19 PM EST Project Member

I am trying to understand the problem. Is the concern the fact that in example above the <iframe>'s document would get geolocation regardless of its origin? If so then I would say the reason is that providing allow="geolocation" is possibly the same as allow="geolocation *".

If you'd like to filter the permission a more detailed allowlist is required:

<iframe src="URL" allow="geolocation http://some-domain.com"></iframe>

would only give permission to the <iframe> document if URL matches http://some-domain.com.

Also note that the policy is only inherited to a nested document if it is already enabled in the parent document. So if the header does not include 'self' for geolocation then no matter what the allowlist, the <iframe> should not get the policy.

Comment 5 by jun.k...@microsoft.com on Sat, Mar 2, 2019, 2:20 AM EST Project Member

Problem here is, let's assume that the top-level website `https://top.tld` wants to use a feature (e.g. geolocation), and they would also want to iframe some other website `https://other.tld`. Given this requirement, how could `https://top.tld` set a policy that doesn't allow `https://other.tld` to inherit geolocation permission even if there is an HTML injection in `https://top.tld`?

This wasn't a problem prior to Permission Delegation because permission wouldn't be inherited to `https://other.tld` (and even if they request one, the prompt would show `https://other.tld` as a requesting origin).

Comment 6 by ekaramad@chromium.org on Sat, Mar 2, 2019, 2:57 AM EST Project Member

Owner: iclel...@chromium.org

Thanks. So as I understand we are talking about some https://top.tld which is already vulnerable due to html injection correct? As in the allow attribute above could be modified and abused? If this is the case I don't quite see how allow is making things worst since the injected HTML could be anything including <script> perhaps?

Back to your point I think we could perhaps summarize this bug to: "allow" should be ignored for certain/all policies right? I think this is more of a spec issue and should perhaps be filed in the github page. I think this thread is relevant (and still open):
https://github.com/w3c/webappsec-feature-policy/issues/252

There is good arguments there as to why having each nested context request its own permission is problematic in other ways.

I am assigning this bug to iclelland@ who is more familiar with permissions and also is working on different paradigms for feature policy inheritance.

Comment 7 by jun.k...@microsoft.com on Sat, Mar 2, 2019, 3:31 AM EST Project Member

> If this is the case I don't quite see how allow is making things worst since the injected HTML could be anything including <script> perhaps?

As stated in the comment 2, injecting <script> is mitigated by CSP.

Comment 8 by rsesek@chromium.org on Sun, Mar 3, 2019, 10:33 AM EST Project Member

Cc: ekaramad@chromium.org

Comment 9 by och...@chromium.org on Thu, Mar 7, 2019, 5:27 AM EST Project Member

Labels: Security_Impact-Stable Security_Severity-Low

Comment 10 by iclel...@chromium.org on Thu, Mar 7, 2019, 11:58 AM EST Project Member

I suspect that there are going to be other exfiltration methods available, given such an XSS vector, but I'm not certain. It seems like a dangerous position for the top-level page to be in. This an issue with HTML injection controlling features, and is already a problem with allowfullscreen / allowusermedia / allowpaymentrequest, right?

Jun, are you able to bring this up on the Feature-Policy repo? (https://github.com/w3c/webappsec-feature-policy/issues) I'd be very interested in discussing this with WebAppSec folks. If you think it's sensitive enough that we should use other channels, I'm open to that as well.

It may be that we want the ability to lock policies, on a page-by-page basis, such that the allow attribute can't override it, but I don't think we want to abandon the allow attribute altogether. It's definitely useful for frame-by-frame permission delegation, and we shouldn't optimize for the "top-level-page-has-an-open-XSS-attack" scenario.

Comment 11 by iclel...@chromium.org on Thu, Mar 7, 2019, 11:59 AM EST Project Member

Cc: a...@google.com mkwst@chromium.org

cc'ing some CSP/Security folks for their opinions as well

Comment 12 by jun.k...@microsoft.com on Thu, Mar 7, 2019, 1:02 PM EST Project Member

I don't think this is something we should talk in Feature Policy repo (as commented in #c5 this became an issue due to Permission Delegation). If Permission Delegation is something that will be standarized or will be implemented by other vendors, then we can discuss in the repo.

Comment 13 by jun.k...@microsoft.com on Thu, Mar 7, 2019, 1:38 PM EST Project Member

For example, repro url would show prompt as frame origin in Firefox. And permission inheritance wouldn't occuer in Firefox either. This is an acceptable implementation that user have to allow permission for requesting origin before they can get access to powerful API.

Where Chrome's implementation would just allow inheritance of permission if top-level page has a permission already, or it would allow requesting permission with top-level page's origin. And there is a case where site owner has no way to prevent it.

Probably title of this bug is misleading, but this is an issue when Permission Delegation is enabled. And this is a result of poor interaction between Permission Delegation and Feature Policy.

Comment 14 by a...@google.com on Mon, Mar 11, 2019, 6:18 PM EDT Project Member

I thought about this a bit and I'm inclined to agree with Jun: the current shape of permission delegation (without requiring user consent to grant access to the embeddee) is dangerous in the face of markup injections. It allows injections in sensitive domains which would otherwise be blocked by CSP to grant access to the origin's permissions to the attacker, reducing the protections offered by CSP. For example, if we prevent the exploitation of XSS in Google Hangouts but the attacker can still get access to the camera, the injection remains very dangerous.

A better model seems to be to require explicit allowlisting of origins permitted to inherit permissions in Feature Policy; that is, the developer would specify the list of expected grantees and the frame with the `allow` attribute would have to be on the list (i.e. make the relationship an AND, not an OR).

Comment 15 by mkwst@chromium.org on Tue, Mar 12, 2019, 4:06 AM EDT Project Member

Cc: hkamila@chromium.org engedy@chromium.org

+engedy@, hkamila@ for permissions generally.

I don't have enough context to have super strong opinions here, but two thoughts:

1. My understanding of delegation is that we only allow delegation from a page that itself has access to the permission. That is, the theoretical Hangouts issue Artur notes would be a problem, but a different page on the same origin that denied itself access to the camera via Feature Policy would not be able to delegate it further. Is that accurate?

2. I think it would be valuable to give pages the ability to deny access to frames in a way that's not overridable by `allow` attributes. Artur's suggestion of requiring the page's FP to be a superset of the individual frames' policy decisions seems pretty reasonable, and in-line with my vague understanding in #1 above.

Comment 16 by mmoroz@chromium.org on Mon, Apr 29, 2019, 4:30 PM EDT Project Member

Labels: M-76

Comment 17 by mkwst@chromium.org on Mon, Apr 29, 2019, 4:35 PM EDT Project Member

Ping iclelland@ and engedy@. Can y'all go talk about this from the FP and Permissions perspective, and make a decision about the path we should take?

Comment 18 by sheriffbot@chromium.org on Wed, Sep 11, 2019, 9:03 AM EDT Project Member

Labels: -M-76 M-77 Target-77

Comment 19 by jun.k...@microsoft.com on Wed, Oct 2, 2019, 3:57 PM EDT Project Member

Hi, is anyone wants to fix anything here? Otherwise, please make it public so that website owners knows the risk.

Comment 20 by sheriffbot@chromium.org on Wed, Oct 23, 2019, 9:12 AM EDT Project Member

Labels: -M-77 Target-78 M-78

Comment 21 by jun.k...@microsoft.com on Fri, Dec 6, 2019, 1:22 PM EST Project Member

Labels: -Restrict-View-SecurityTeam

Making this public, as the problem isn't solved and other browser is considering to implement Permission Delegation without any mitigation.

Comment 22 by wfh@chromium.org on Fri, Dec 6, 2019, 1:41 PM EST Project Member

Labels: Restrict-View-SecurityTeam Arch-All

Hi - was an issue raised in the public webappsec-feature-policy tracker for this? If the issue is public, either by a public tracking bug or e.g. a blog/tweet/presentation then we can open this crbug up to reflect that reality, but I don't think we should be opening it just because it's unfixed.

Comment 23 by jun.k...@microsoft.com on Fri, Dec 6, 2019, 1:46 PM EST Project Member

Filed here:
https://github.com/w3c/webappsec-feature-policy/issues/357

Comment 24 by wfh@chromium.org on Fri, Dec 6, 2019, 2:19 PM EST Project Member

Labels: -Restrict-View-SecurityTeam allpublic reward-ineligible

This issue is now public as per #23 so opening the bug up.

Comment 25 by sheriffbot@chromium.org on Wed, Dec 11, 2019, 9:13 AM EST Project Member

Labels: -M-78 Target-79 M-79

Comment 26 by cha...@chromium.org on Wed, Jan 15, 2020, 9:53 AM EST Project Member

Summary: Feature Policy 'allow' attribute can override top-level policy in frames (was: Feature Policy does NOT make sense)

Changing the title to more accurately reflect the nature of the issue.

Comment 27 by bugdroid on Sat, Feb 1, 2020, 11:48 PM EST Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src.git/+/8165a2a32efb5933ad46f5435b207c501478ab5b

commit 8165a2a32efb5933ad46f5435b207c501478ab5b
Author: Ian Clelland <iclelland@chromium.org>
Date: Sun Feb 02 04:46:51 2020

Add use counter to feature policy to gauge web-compatibility of change.

A proposed change to the semantics of the Feature Policy header could
change the availability of features in some frames, depending on how the
Feature-Policy header and allow attribute have been specified. This CL
adds enough of the proposed logic to be able to tell whether the output
would change for any given call to IsFeatureEnabled, and counts how
often that situation occurs.

~~Bug: 937131~~
Change-Id: I2b587546f8d08a0c9d5e0c2bab60b77d771751c5
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2028630
Commit-Queue: Ian Clelland <iclelland@chromium.org>
Reviewed-by: Alexei Svitkine <asvitkine@chromium.org>
Reviewed-by: Jeremy Roman <jbroman@chromium.org>
Cr-Commit-Position: refs/heads/master@{#737617}

[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/third_party/blink/common/feature_policy/feature_policy.cc
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/third_party/blink/common/feature_policy/feature_policy_unittest.cc
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/third_party/blink/public/common/feature_policy/feature_policy.h
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/third_party/blink/public/mojom/web_feature/web_feature.mojom
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/third_party/blink/renderer/core/execution_context/execution_context.cc
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/third_party/blink/renderer/core/execution_context/execution_context.h
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/tools/metrics/histograms/enums.xml
[modify] https://crrev.com/8165a2a32efb5933ad46f5435b207c501478ab5b/tools/metrics/histograms/histograms.xml

Comment 28 by sheriffbot@chromium.org on Wed, Feb 5, 2020, 10:49 AM EST Project Member

Labels: -M-79 M-80 Target-80

Comment 29 by sheriffbot on Thu, Apr 9, 2020, 12:30 PM EDT Project Member

Labels: -M-80 Target-81 M-81

Comment 30 by sheriffbot on Wed, May 20, 2020, 1:31 PM EDT Project Member

Labels: -M-81 M-83 Target-83

Comment 31 by sheriffbot on Thu, Jul 16, 2020, 1:34 PM EDT Project Member

Labels: -M-83 Target-84 M-84

Comment 32 by sheriffbot on Wed, Aug 26, 2020, 1:41 PM EDT Project Member

Labels: -M-84 Target-85 M-85

Comment 33 by sheriffbot on Wed, Oct 7, 2020, 1:41 PM EDT Project Member

Labels: -M-85 M-86 Target-86

Comment 34 by bugdroid on Wed, Nov 11, 2020, 3:00 PM EST Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src.git/+/d5a2e2f0c728e4a0adc5d3193421900495ef0820

commit d5a2e2f0c728e4a0adc5d3193421900495ef0820
Author: Ian Clelland <iclelland@chromium.org>
Date: Wed Nov 11 19:59:20 2020

Change Feature-Policy header semantics

This change implements the algorithmic changes for a recent change
to the Feature/Permissions policy spec:
https://github.com/w3c/webappsec-permissions-policy/pull/378

With this change, the Feature-Policy or Permissions-Policy headers by
themselves cannot be used to delegate powerful features to cross-origin
iframes; the allow attribute must be used as well. To allow this to
still be ergonomic, the default value for the header for powerful
features is effectively '*', so that delegation is allowed by the header
implicitly. The header can now be used effectively to completely block
access to a feature, as any origins not present in the header allowlist
cannot be granted the feature through the allow attribute.

This also removes some code which previously only existed to track the
cases where this change would affect the output of an IsFeatureEnabled
call.

Several tests will have been modified or rewritten prior to landing
this change; this CL depends on the following (though they are all
independent, so they are not chained together):
- https://crrev.com/c/2424633
- https://crrev.com/c/2424634
- https://crrev.com/c/2424635
- https://crrev.com/c/2424654
- https://crrev.com/c/2424655
- https://crrev.com/c/2424657
- https://crrev.com/c/2425003
- https://crrev.com/c/2425004
(See Patchset 8 for a version with the changes from all of those CLs
included.)

This CL, while large, can best be understood as the union of the
following changes:

- Algorithm changes, including the removal of previous "what-if" code
  and metrics:
  feature_policy.cc
  feature_policy.h
  execution_context.cc

- Unit tests to cover those changes:
  feature_policy_unittest.cc
  render_frame_host_feature_policy_unittest.cc

- Update WPT test expectations to account for the change in behaviour
  when only the header is used:
  3p/b/web_tests/external/wpt/feature-policy/feature-policy-*
  3p/b/web_tests/external/wpt/permissions-policy/permissions-policy-*

- Update Blink web tests for fullscreen and payment request to validate
  that both are now working correctly with the new header semantics:
  3p/b/web_tests/http/tests/feature-policy/fullscreen*
  3p/b/web_tests/http/tests/feature-policy/payment*

- Update Blink web tests for the iframe policy JS interface because of
  new test expectations when features are allowed/disallowed by header:
  3p/b/renderer/core/feature_policy/policy_test.cc
  3p/b/web_tests/http/tests/feature-policy/policy_iframes.php

~~Bug: 1095641~~, ~~837124~~
Change-Id: Iecbb0950c27a4565998ee5192590d6691a03b4a3
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2363169
Reviewed-by: Yoav Weiss <yoavweiss@chromium.org>
Reviewed-by: Charlie Hu <chenleihu@google.com>
Reviewed-by: Ken Buchanan <kenrb@chromium.org>
Commit-Queue: Ian Clelland <iclelland@chromium.org>
Cr-Commit-Position: refs/heads/master@{#826408}

[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/content/browser/renderer_host/render_frame_host_feature_policy_unittest.cc
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/common/feature_policy/feature_policy.cc
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/common/feature_policy/feature_policy_unittest.cc
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/public/common/feature_policy/feature_policy.h
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/renderer/core/execution_context/execution_context.cc
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/renderer/core/execution_context/execution_context.h
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/renderer/core/feature_policy/policy_test.cc
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-all.https.sub.html
[delete] https://crrev.com/9072af77afae87bd2b7bc266a3af9ef50c465a72/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-self.https.sub-expected.txt
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-self.https.sub.html
[delete] https://crrev.com/9072af77afae87bd2b7bc266a3af9ef50c465a72/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-some-override.https.sub-expected.txt
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-some-override.https.sub.html
[delete] https://crrev.com/9072af77afae87bd2b7bc266a3af9ef50c465a72/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-

some.https.sub-expected.txt
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-frame-policy-allowed-for-some.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-header-policy-allowed-for-all.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-header-policy-allowed-for-some.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-header-policy-declined.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-nested-header-policy-allowed-for-all.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/feature-policy-nested-header-policy-allowed-for-self.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/feature-policy/payment-allowed-by-feature-policy.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/html/cross-origin-embedder-policy/cross-origin-isolated-permission.https.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/permissions-policy/payment-allowed-by-permissions-policy.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/permissions-policy/permissions-policy-header-policy-allowed-for-all.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/permissions-policy/permissions-policy-header-policy-allowed-for-some.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/permissions-policy/permissions-policy-header-policy-declined.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/permissions-policy/permissions-policy-nested-header-policy-allowed-for-all.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/external/wpt/permissions-policy/permissions-policy-nested-header-policy-allowed-for-self.https.sub.html
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/http/tests/feature-policy/fullscreen-enabledforall.php
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/http/tests/feature-policy/fullscreen-enabledforself.php
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/http/tests/feature-policy/payment-enabledforall.php
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/http/tests/feature-policy/payment-enabledforself.php
[modify] https://crrev.com/d5a2e2f0c728e4a0adc5d3193421900495ef0820/third_party/blink/web_tests/http/tests/feature-policy/policy_iframes.php

Comment 35 by iclel...@chromium.org on Wed, Nov 11, 2020, 3:34 PM EST Project Member

Status: Fixed (was: Assigned)

This should now be fixed; in line with the spec change, neither the Feature-Policy nor Permissions-Policy header can be overridden by an allow attribute to grant access to a frame whose origin was not listed.

That also implies that returning a header of:

```
Feature-Policy: geolocation 'self'
```

or it's Permissions-Policy equivalent:

```
Permissions-Policy: geolocation=self
```

is different than not returning any header at all. With such a header, *only* same-origin frames could possibly be granted access to geolocation; without a header, *any* frame could potentially be granted such access, but only same-origin frame would be granted access by default.

Comment 36 by jun.k...@microsoft.com on Wed, Nov 11, 2020, 5:58 PM EST Project Member

Great, thanks!

Comment 37 by adetaylor@google.com on Wed, Jan 13, 2021, 5:48 PM EST Project Member

Labels: Release-0-M88

Comment 38 by amyressler@google.com on Tue, Jan 19, 2021, 1:57 PM EST Project Member

Labels: CVE-2021-21139 CVE_description-missing

Comment 39 by amyressler@google.com on Tue, Feb 9, 2021, 9:27 AM EST Project Member

Labels: -CVE_description-missing CVE_description-submitted