bugs chromium New issue Open issues Search chromium issue Sign in

Starred by 2 users

Owner:	dsv@chromium.org
CC:	mkwst@chromium.org dominickn@chromium.org adetaylor@chromium.org bmeu...@chromium.org yangguo@chromium.org rdevl...@chromium.org sawallner@chromium.org hablich@chromium.org creis@chromium.org dsv@google.com
Status:	Fixed *(Closed)*
Components:	Platform>DevTools Platform>Extensions Platform>Apps
Modified:	Jul 29, 2022
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	----
OS:	Linux, Windows, Chrome, Mac, Fuchsia
Pri:	1
Type:	Bug-Security

Hotlist-Merge-Review
M-100
Deadline-Exceeded
Security_Severity-High
allpublic
reward-inprocess
reward-15000
CVE_description-submitted
Target-88
Target-84
Target-85
Target-86
Target-87
Target-89
Target-90

FoundIn-84
Target-91

Issue 1106456: Security: Possible to escape sandbox via devtools_page and Feedback app

Reported by derce...@gmail.com on Thu, Jul 16, 2020, 3:16 PM EDT

Code

VULNERABILITY DETAILS
By specifying a devtools_page entry, an extension can run code within the context of an inspected page once the user opens the devtools. An extension can use this ability to run code within the context of the Feedback app and create an app window. That window can then be used to launch a file that was downloaded, provided that the current browser instance is not the system default browser.

VERSION
Chrome Version: Tested on 84.0.4147.89 (stable) and 86.0.4204.1 (canary)
Operating System: Windows 10, version 1909

REPRODUCTION CASE
1. Ensure that the browser instance you're using to test is not the system default browser.
2. Install the attached extension.
3. Open the devtools on a non-privileged page.
4. Wait about 5 seconds.
5. The target executable (in this case, Process Explorer) should be started.

CREDIT INFORMATION
Reporter credit: David Erceg

background.js
275 bytes View Download

devtools_page.html
132 bytes View Download

devtools_page.js
3.0 KB View Download

local.html
334 bytes View Download

manifest.json
289 bytes View Download

Comment 1 by derce...@gmail.com on Thu, Jul 16, 2020, 3:27 PM EDT

The extension here goes through the following steps:

1. Once the user has opened the devtools, the extension will download two files:

- local.html
- Process Explorer

2. Once both of the above downloads have completed, the extension will navigate the page being debugged to:

view-source:chrome-extension://gfdkimpbcpahaombhbimeihdjnejgicl/js/feedback.js

This is a javascript file contained within the Feedback app.

Note that the navigation to this URL is performed via the chrome.debugger API by sending the Page.navigate command. The reason this is done is that it's not possible to navigate to this location using chrome.tabs.update (an error page will simply be shown).

Additionally, the only reason the above URL is a view-source: URL is that it's not possible to navigate to the file directly. The view-source: URL, however, will load and the page will have access to all of the APIs that the Feedback app has access to.

3. The extension will then use chrome.devtools.inspectedWindow.eval to run code within the context of the page. Note that unlike ~~issue 1101897~~, which specifically relies on a timing issue to run code within the context of a privileged page, here, chrome.devtools.inspectedWindow.eval can simply be called directly. That's because although the devtools attempts to block extensions when debugging a chrome: or devtools: page, chrome-extension: pages aren't currently blocked in any way:

https://source.chromium.org/chromium/chromium/src/+/master:third_party/devtools-frontend/src/front_end/extensions/ExtensionServer.js;l=1030;drc=4576282f49d8129b6374dd0fd389271c0a291b6a

The extension uses chrome.devtools.inspectedWindow.eval to run the following code:

chrome.app.window.create("file:///path/to/local.html")

The chrome.downloads API is used to retrieve the path to this file (which was downloaded in step 1). It would be possible to determine the location of the downloads directory without use of the chrome.downloads API (via the chrome.settingsPrivate.getPref method made available to the Chrome Media Router extension), but using chrome.downloads simplifies the demonstration.

For apps that the user installs, the call above would fail. That's because passing an absolute path to chrome.app.window.create normally triggers an error:

https://source.chromium.org/chromium/chromium/src/+/master:extensions/browser/api/app_window/app_window_api.cc;l=159;drc=12b407642b48f5f29782f09a703021be62d0ccfb

However, apps installed as components (such as the Feedback app) are allowed to pass an absolute URL into chrome.app.window.create, which means the call above will succeed.

4. Once the app window has been created and local.html has been loaded, it will make the following call:

window.open("file:///path/to/processexplorer");

Again, the chrome.downloads API is used to retrieve the path to this file (which was also downloaded in step 1). Also, this call is the reason why local.html is downloaded first - normally, only local files have the ability to call window.open with the path of a local file.

When the current browser instance is not the system default browser, calling window.open from an app window results in the call being passed directly to the OS:

https://source.chromium.org/chromium/chromium/src/+/master:chrome/browser/ui/apps/chrome_app_delegate.cc;l=109;drc=3abb32da2944ffe178dd66f404e7e1bb88a58ed0

For a file: URL, that results in the file being opened. Since the file in this case is an executable, the executable is run.

Comment 2 by derce...@gmail.com on Thu, Jul 16, 2020, 3:40 PM EDT

One other related thing to note is that the _canInspectURL method in ExtensionServer.js blocks attempts to debug the Web Store:

https://source.chromium.org/chromium/chromium/src/+/master:third_party/devtools-frontend/src/front_end/extensions/ExtensionServer.js;l=1033;drc=4576282f49d8129b6374dd0fd389271c0a291b6a

However, because pages with the chrome-extension scheme aren't blocked, this check won't stop an extension from being able to make use of APIs available to the Web Store.

To do that, an extension could navigate the page being debugged to a file within the Web Store app. For example:

chrome-extension://ahfgeienlihckogmohjhadlkjgocpleb/webstore_icon_16.png

That page has access to the full set of APIs provided to the app. An extension embedded within the devtools can then make use of those APIs.

Comment 3 by metzman@chromium.org on Fri, Jul 17, 2020, 4:07 PM EDT Project Member

Cc: caseq@chromium.org rdevl...@chromium.org
Labels: Security_Severity-High Security_Impact-Stable OS-Chrome OS-Fuchsia OS-Linux OS-Mac OS-Windows Pri-1
Components: Platform>DevTools Platform>Extensions

I tried reproing this on trunk linux (replacing procmon with a different executable that would work on Linux) but it didn't seem to work, the download of the executable failed with the message "Failed - No file".

caseq@ or rdevlin.cronin@ could one of you please take a look?

Comment 4 by sheriffbot on Sat, Jul 18, 2020, 1:56 PM EDT Project Member

Labels: Target-84 M-84

Setting milestone and target because of Security_Impact=Stable and high severity.

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 5 by hablich@chromium.org on Mon, Jul 20, 2020, 3:46 AM EDT Project Member

Status: Assigned (was: Unconfirmed)
Owner: bmeu...@chromium.org
Cc: yangguo@chromium.org sawallner@chromium.org hablich@chromium.org

bmeurer@ I am not sure we can do anything about it?

Comment 6 by bmeu...@chromium.org on Mon, Jul 20, 2020, 9:16 AM EDT Project Member

Owner: caseq@chromium.org
Cc: -caseq@chromium.org bmeu...@chromium.org

I'm lacking some historical / technical context on this. From a first glance it looks like this could be WAI.

I'd like to hear from caseq@ on this.

Comment 7 by rsleevi@chromium.org on Tue, Jul 28, 2020, 1:10 PM EDT Project Member

caseq: Could you see Comment #6?

Comment 8 by sheriffbot on Fri, Jul 31, 2020, 1:37 PM EDT Project Member

caseq: Uh oh! This issue still open and hasn't been updated in the last 14 days. This is a serious vulnerability, and we want to ensure that there's progress. Could you please leave an update with the current status and any potential blockers?

If you're not the right owner for this issue, could you please remove yourself as soon as possible or help us find the right one?

If the issue is fixed or you can't reproduce it, please close the bug. If you've started working on a fix, please set the status to Started.

Thanks for your time! To disable nags, add the Disable-Nags label.

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 9 by derce...@gmail.com on Thu, Aug 6, 2020, 3:23 AM EDT

While waiting for caseq to take a look at this, I think I can offer a bit of additional context (based on what I've found when looking back through some of the history of the relevant code), and whether the behavior is intended.

Firstly, since the ability to call chrome.app.window.create with an absolute URL is blocked for regular extensions, it's something that's only relevant for component extensions.

The ability for a component extension to pass an absolute URL was first added in this commit:

https://chromiumcodereview.appspot.com/11048045

Before that, the URL was always treated as relative to the extension:

https://chromiumcodereview.appspot.com/10391199/diff/35003/chrome/browser/extensions/api/app_window/app_window_api.cc#newcode27

It looks like the change was made to allow chrome.app.window.create to be called with the following URL:

chrome://settings-frame/settings

From searching through the Chromium codebase, I've found the following two places where an absolute URL is currently passed to chrome.app.window.create (though it's possible there are other places that I missed or am unaware of):

URL: chrome://histograms
https://source.chromium.org/chromium/chromium/src/+/master:chrome/browser/resources/feedback/js/feedback.js;l=510;drc=d6a8ee4d9d1db6a1453ed94a2214a98d78c7a866

URL: chrome://slow_trace/tracing.zip
https://source.chromium.org/chromium/chromium/src/+/master:chrome/browser/resources/feedback/js/feedback.js;l=167;drc=d6a8ee4d9d1db6a1453ed94a2214a98d78c7a866

Both of these are also chrome: URLs.

So being able to pass a file: URL into chrome.app.window.create and then directly open a local file (particularly an executable file) from that window doesn't seem like it would be part of the intended functionality.

Even if a component app were relying on being able to open a local file using this method, it wouldn't work when the current browser instance was the system default; in that case, the file would be opened within the browser.

I can think of two ways the overall issue raised here might be fixed:

1. The first would be to block embedded extensions within the devtools when debugging a chrome-extension: page.
2. The second would be to block window.open calls in app windows, when the target is a file: URL. Regular apps can't call window.open with a file: URL anyway, so they would be unaffected. Apps installed as components would first have to call chrome.app.window.create with a file: URL, since they can't call window.open with a file: URL either.

It's also possible to take advantage of the behavior described here using the behavior described in ~~issue 1113558~~. That would allow an extension with the debugger permission to open a local executable file, provided that the current browser instance isn't the system default browser. No user interaction would be required after the extension install.

So preventing window.open calls with file: URLs would fix this issue and prevent it from being taken advantage of in other situations.

Comment 10 by sheriffbot on Sat, Aug 15, 2020, 1:36 PM EDT Project Member

caseq: Uh oh! This issue still open and hasn't been updated in the last 29 days. This is a serious vulnerability, and we want to ensure that there's progress. Could you please leave an update with the current status and any potential blockers?

If you're not the right owner for this issue, could you please remove yourself as soon as possible or help us find the right one?

If the issue is fixed or you can't reproduce it, please close the bug. If you've started working on a fix, please set the status to Started.

Thanks for your time! To disable nags, add the Disable-Nags label.

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 11 by sheriffbot on Wed, Aug 26, 2020, 1:37 PM EDT Project Member

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

Comment 12 by caseq@chromium.org on Fri, Aug 28, 2020, 5:10 PM EDT Project Member

Devlin, do you have an insight on why we expose extension bindings to view-source: pages?

Comment 13 by rdevl...@chromium.org on Fri, Aug 28, 2020, 6:02 PM EDT Project Member

Cc: creis@chromium.org

Interesting!

I confess, I have very little knowledge of how view-source works under the hood. But poking around a view-source console for a bit, it looks like it actually "thinks" it's the real page. From the task manager, it also seems like it commits within the same process as the inspected page. If the renderer commits to an origin that has access to extension APIs from within a process that's trusted to have access to those APIs, we would instantiate the bindings on the renderer side and allow access - as far as most systems know, it's an extension page.

What's more, it doesn't look like this is restricted to extension APIs. Opening up view-source for a chrome:// page like settings or extensions also grants access to chrome.send(). Which means there's a lot of equal badness that can happen there, if the extension were able to connect to those.

I think the underlying issue here is that the extension shouldn't be allowed to debug a view-source frame if it can't access the underlying URL. Though, separately, I am curious why view-source pages have JS access at all - it seems like we could make them just plaintext. Charlie, do you have some more context there?

From c#1:

> That's because although the devtools attempts to block extensions when debugging a chrome: or devtools: page, chrome-extension: pages aren't currently blocked in any way:

That's interesting. We don't normally allow extensions to debug other extensions - check out ExtensionCanAttachToURL() [1] and PermissionsData::IsRestrictedURL() [2] - doing so should require a very-scary commandline switch ("extensions-on-chrome-urls", at which point all bets are off). Is this an issue where a) the debugger isn't being properly detached after navigating to a restricted page (in which case, would this be fixed by some of the other bugs we've seen here?), or b) we're not classifying the inspected URL properly as being restricted? (Said differently, is ExtensionCanAttachToURL() checked upon navigation to the view-source page?)

[1]
https://source.chromium.org/chromium/chromium/src/+/master:chrome/browser/extensions/api/debugger/debugger_api.cc;l=91-110;drc=02fc05043bd38d318ba9ce59ccc4c7602284f990
[2]
https://source.chromium.org/chromium/chromium/src/+/master:extensions/common/permissions/permissions_data.cc;l=107-151;drc=02fc05043bd38d318ba9ce59ccc4c7602284f990

Comment 14 by caseq@chromium.org on Fri, Aug 28, 2020, 6:13 PM EDT Project Member

I think ExtensionCanAttachToURL() works as intended here, at least based by David's comments (https://bugs.chromium.org/p/chromium/issues/attachmentText?aid=456756#65) and the fact that view-source: is not in within the list of valid schemes for URLPattern::IsValidSchemeForExtensions() -- the problem is rather in chrome.devtools API (i.e. the front-end, AKA devtools_page extensions) that uses a different check. We'll fix that check, of course, although it would still leave us with ~~http://crbug.com/1101897~~ for now.

That said, exposing bindings to a scheme that isn't supposed to take any legitimate advantage of them looked suspicious to me, so I wonder if this needs to be fixed in more than one place.

Comment 15 by creis@chromium.org on Fri, Aug 28, 2020, 6:13 PM EDT Project Member

> Though, separately, I am curious why view-source pages have JS access at all - it seems like we could make them just plaintext. Charlie, do you have some more context there?

Sorry, that predates me. :) I'm aware that view-source loads the original URL in a different mode, but I'm not sure how the mode works. In general (without having read the rest of the context here), I'd be happy to see it locked down.

Comment 16 by rdevl...@chromium.org on Fri, Aug 28, 2020, 6:21 PM EDT Project Member

> That said, exposing bindings to a scheme that isn't supposed to take any legitimate advantage of them looked suspicious to me, so I wonder if this needs to be fixed in more than one place.

> I'm aware that view-source loads the original URL in a different mode, but I'm not sure how the mode works. In general (without having read the rest of the context here), I'd be happy to see it locked down.

Sounds like we're all on the same page of "it'd be nice to lock this down more, since it shouldn't _really_ need anything". Do we know who the best owner for view-source changes would be?

Comment 17 by sheriffbot on Tue, Sep 15, 2020, 3:15 PM EDT Project Member

Labels: Deadline-Exceeded

We commit ourselves to a 60 day deadline for fixing for high severity vulnerabilities, and have exceeded it here. If you're unable to look into this soon, could you please find another owner or remove yourself so that this gets back into the security triage queue?

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 18 by sheriffbot on Wed, Oct 7, 2020, 1:37 PM EDT Project Member

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

Comment 19 by vakh@chromium.org on Thu, Oct 22, 2020, 12:30 PM EDT Project Member

Friendly ping from the security 👮 for this High severity bug. Any updates?

For high severity vulnerabilities, we aim to deploy the patch to all Chrome
users in under 60 days.

Comment 20 by sheriffbot on Fri, Oct 30, 2020, 6:46 PM EDT Project Member

Labels: reward-potential

Comment 21 by sheriffbot on Wed, Nov 18, 2020, 12:22 PM EST Project Member

Labels: -M-86 M-87 Target-87

Comment 22 by tsepez@chromium.org on Mon, Nov 30, 2020, 1:32 PM EST Project Member

Cc: mkwst@chromium.org

+mkwst, anyone in OWP have an interest in this? Only tangentially related, but when it comes to taking away capabilities within a renderer, ahem ...

Comment 23 by sheriffbot on Wed, Jan 20, 2021, 12:23 PM EST Project Member

Labels: -M-87 Target-88 M-88

Comment 24 by adetaylor@google.com on Wed, Jan 20, 2021, 6:56 PM EST Project Member

Labels: -reward-potential external_security_report

Comment 25 by sheriffbot on Wed, Mar 3, 2021, 12:22 PM EST Project Member

Labels: -M-88 Target-89 M-89

Comment 26 by sheriffbot on Wed, Mar 10, 2021, 8:05 PM EST Project Member

Labels: reward-potential

Comment 27 by zhangtiff@google.com on Wed, Mar 17, 2021, 7:12 PM EDT Project Member

Labels: -reward-potential external_security_bug

Comment 28 by sheriffbot on Thu, Apr 15, 2021, 12:23 PM EDT Project Member

Labels: -M-89 M-90 Target-90

Comment 29 by sheriffbot on Wed, May 26, 2021, 12:24 PM EDT Project Member

Labels: -M-90 M-91 Target-91

Comment 30 by adetaylor@google.com on Thu, Jul 8, 2021, 4:26 PM EDT Project Member

Labels: FoundIn-84

Comment 31 by sheriffbot on Thu, Aug 5, 2021, 1:43 PM EDT Project Member

Labels: -Security_Impact-Stable Security_Impact-Extended

Comment 32 by sheriffbot on Fri, Aug 6, 2021, 12:23 PM EDT Project Member

Labels: -Security_Impact-Extended

Comment 33 by sheriffbot on Fri, Aug 6, 2021, 12:28 PM EDT Project Member

Labels: Security_Impact-Extended

Comment 34 by sheriffbot on Fri, Aug 6, 2021, 1:30 PM EDT Project Member

Labels: -Security_Impact-Extended Security_Impact-Stable

Comment 35 by sheriffbot on Sat, Aug 7, 2021, 12:23 PM EDT Project Member

Labels: -M-91 Target-92 M-92

Comment 36 by sheriffbot on Mon, Aug 16, 2021, 1:14 PM EDT Project Member

Labels: -Security_Impact-Stable Security_Impact-Extended

Comment 37 by sheriffbot on Sat, Sep 11, 2021, 12:23 PM EDT Project Member

Labels: -M-92 M-93 Target-93

Comment 38 by sheriffbot on Wed, Sep 22, 2021, 12:24 PM EDT Project Member

Labels: -M-93 Target-94 M-94

Comment 39 by sheriffbot on Mon, Nov 15, 2021, 12:24 PM EST Project Member

Labels: -M-94 Target-96 M-96

Comment 40 by creis@chromium.org on Thu, Dec 2, 2021, 12:28 AM EST Project Member

Cc: adetaylor@chromium.org

Thanks to adetaylor@ for a ping on this. Devlin and I both looked a little more closely today. I really appreciate the detailed report and followup comments, and it's unfortunate this fell between the cracks.

First, I think the view-source privileges part is probably a red herring. I think view-source pages are put into the same SiteInstance / principal as a normal URL from that origin, and loaded in a special RenderFrame mode. Even if we went to the effort of putting view-source: pages into separate unprivileged processes (still separate from other sites), they would still be able to navigate to their own non-view-source URL and then end up in a privileged process (e.g., using injected scripts as done here). So I'm guessing the attack would still work, with an extra "location.href = location.href" step to exit view-source mode. We can have a separate discussion of whether it's worth the process model complexity to reduce view-source privileges anyway, but I think there are more important bugs to solve here.

Second, I'm wondering if the severity is set properly here. The fact that the browser cannot be the system default browser is one mitigation, though perhaps not enough to reduce to Medium severity on its own. Another mitigation is having to install a malicious extension with a "Manage your downloads" message shown during installation due to the downloads permission. That already seems close to something that could do a sandbox escape for free-- I think the only distinction is that a second downloads.open permission is technically required to open a downloaded file (per https://developer.chrome.com/docs/extensions/reference/downloads/#method-open), though that isn't documented at https://developer.chrome.com/docs/extensions/mv3/declare_permissions/? Combined, I wonder if this is Medium rather than high, but I'll defer to security sheriffs on that.

Still, it seems like we have a lot of partial security checks here that are being evaded, and we should try to fix them. Here's a list I've identified:

1) If chrome.tabs.update doesn't allow navigating to the view-source:chrome-extension://gfdkimpbcpahaombhbimeihdjnejgicl/js/feedback.js URL, can we prevent it via Page.navigate as well?
At first glance, it seems wrong that the restrictions for these differ.

2) Can we block view-source platform app URLs from loading in tabs, if their equivalent platform app URLs can't load in tabs?
Devlin suggested this over chat, and it makes sense to me. Not sure where the platform app URL restriction lives.

3) Can we prevent DevTools extensions from running code in extensions (similar to chrome:// and devtools:// pages)?
This is suggestion 1 in comment 9, confirmed by Devlin in comment 13. Maybe this involves excluding chrome-extension:// from the _canInspectURL logic here?
https://source.chromium.org/chromium/chromium/src/+/master:third_party/devtools-frontend/src/front_end/extensions/ExtensionServer.js;l=1030;drc=4576282f49d8129b6374dd0fd389271c0a291b6a
I think this would also fix the Chrome Web Store attack vector described in comment 2.

4) Can we remove the component app exception that allows creating a window to a file:// URL?
This is related to suggestion 2 in comment 9, and Devlin turned up the comment below which suggests that we might be able to remove this ability for component apps:
https://source.chromium.org/chromium/chromium/src/+/master:extensions/browser/api/app_window/app_window_api.cc;l=150-152;drc=12b407642b48f5f29782f09a703021be62d0ccfb
In particular, it looks like that ability was added in r161828 for issue 130210, and I wonder if allowing chrome:// URLs is sufficient and safe (rather than arbitrary URLs like file://).

5) Can we block platform apps from sending executable/dangerous URLs to the OS (when Chrome isn't the default browser)?
See
https://source.chromium.org/chromium/chromium/src/+/master:chrome/browser/ui/apps/chrome_app_delegate.cc;l=109;drc=3abb32da2944ffe178dd66f404e7e1bb88a58ed0 from comment 1.

These seem split across DevTools and extension/app code. Maybe Devlin can work with DevTools folks to split them up, if these changes make sense?

Comment 41 by rdevl...@chromium.org on Thu, Dec 2, 2021, 1:07 PM EST   Project Member

Cc: dominickn@chromium.org
Components: Platform>Apps

Thanks, Charlie!

> So I'm guessing the attack would still work, with an extra "location.href = location.href" step to exit view-source mode.

Except that then it's trying to open a platform app resource in a chrome tab, which wouldn't work :) So I think this *would* stop the attack vector in this case. That said, I agree there's complexity here, and I don't think we should remove privileges for view-source just for this issue. It might still be worth discussing, though.

Thanks for concretizing the tasks here! 2) and 4) seem very much in the extensions area; I'll file separate bugs for those and see if we can get some attention on them. 1) is in CDP, so I'll leave that to dsv@. dsv@ also asked about 3) recently in another thread; dsv@, are you planning on looking into that? 5) is apps-related - dominickn@, can you help find someone to tackle that?

Comment 42 by rdevl...@chromium.org on Thu, Dec 2, 2021, 1:28 PM EST Project Member

Filed issue 1276041 for 2) and issue 1276046 for 4) from c#40.

Comment 43 by rdevl...@chromium.org on Thu, Dec 2, 2021, 1:29 PM EST Project Member

Blockedon: 1276041

Comment 44 by rdevl...@chromium.org on Thu, Dec 2, 2021, 1:29 PM EST Project Member

Blockedon: 1276046

Comment 45 by dominickn@chromium.org on Thu, Dec 2, 2021, 5:25 PM EST Project Member

> 5) Can we block platform apps from sending executable/dangerous URLs to the OS (when Chrome isn't the default browser)?

What is an executable/dangerous URL in this context? And is it expected behaviour for platform apps to be able to execute something like window.open("file://path/to/local") (given that platform apps are technically "local" files)?

Comment 46 by dsv@chromium.org on Fri, Dec 3, 2021, 3:27 AM EST Project Member

1) We could add check to Page.navigate, but it would still be possible to do achieve the same result using Runtime.evaluate("location.href=...") or similar with
DOM.SetOuterHTML("<iframe src=...>") so I'm not sure we could really address this.

3) Yes, I am.

Comment 47 by rdevl...@chromium.org on Tue, Dec 14, 2021, 8:19 PM EST Project Member

@ #c45

> What is an executable/dangerous URL in this context? And is it expected behaviour for platform apps to be able to execute something like window.open("file://path/to/local") (given that platform apps are technically "local" files)?

Hmm... I'd lean towards saying platform apps shouldn't be able to open arbitrary files, no matter what. This capability today is only possible for component apps, so it seems like there's little risk of breakage there, and Chrome already blocks renderer-initiated navigation to file: URLs. "Executable/dangerous" here could be something like an exe - but I'd argue we should just disallow this in general, and probably shouldn't be punting to the system for opening any resources. Of course, I might be missing a use case that relies on this, but I can't think of any off the top of my head. I'd be interested in the code history there to see why we added it.

Comment 48 by sheriffbot on Wed, Feb 2, 2022, 12:24 PM EST Project Member

Labels: -M-96 M-98 Target-98

Comment 49 by adetaylor@google.com on Tue, Mar 22, 2022, 10:22 PM EDT Project Member

dsv@ are you a better owner here these days?

And, could you confirm what needs to be done under this particular crbug (as opposed to the bugs which Devlin raised separately?)

Comment 50 by dsv@chromium.org on Wed, Mar 23, 2022, 4:12 AM EDT Project Member

In crbug/1115460 I already forbid devtools extension access to component extension. Do I understand correctly that forbidding "view-source:" prefix to the list of forbidden origin would close this vulnerability?

Comment 51 by adetaylor@chromium.org on Thu, Mar 24, 2022, 10:36 AM EDT Project Member

(to state the obvious - I don't understand this bug well enough to answer #c50 definitively - so I hope that question was addressed at Devlin or Charlie!)

Comment 52 by rdevl...@chromium.org on Mon, Mar 28, 2022, 4:37 PM EDT Project Member

@50: I think that would be sufficient. I'd prefer the farther-reaching change of changing view-source pages to be permission-less text-only contexts (which would then solve this issue and any related issues), but that's a much farther reaching change.

While issue 1276041 is nice-to-have, I don't think it needs to block resolution of this bug.

Comment 53 by sheriffbot on Wed, Mar 30, 2022, 12:24 PM EDT Project Member

Labels: -M-98 M-100 Target-100

Comment 54 by Git Watcher on Mon, Apr 4, 2022, 4:34 AM EDT Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/359da7a1736043e30ee0a9f1e5ac9597422bf877

commit 359da7a1736043e30ee0a9f1e5ac9597422bf877
Author: Danil Somsikov <dsv@chromium.org>
Date: Mon Apr 04 08:32:54 2022

Add an extra test to check that devtools extension can't use
view-source: scheme to circumvent access checks.

~~Bug: 1106456~~
Change-Id: Icedc2c65f94de97e18ecb66e622ac1fa2b0ef044
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/3563061
Auto-Submit: Danil Somsikov <dsv@chromium.org>
Reviewed-by: Yang Guo <yangguo@chromium.org>
Commit-Queue: Yang Guo <yangguo@chromium.org>
Cr-Commit-Position: refs/heads/main@{#988409}

[modify]
https://crrev.com/359da7a1736043e30ee0a9f1e5ac9597422bf877/chrome/browser/devtools/devtools_browsertest.cc
[modify]
https://crrev.com/359da7a1736043e30ee0a9f1e5ac9597422bf877/chrome/test/data/devtools/extensions/can_inspect_url/d

evtools.js
[modify]
https://crrev.com/359da7a1736043e30ee0a9f1e5ac9597422bf877/chrome/test/data/devtools/extensions/can_inspect_url/manifest.json

Comment 55 by dsv@chromium.org on Mon, Apr 4, 2022, 5:26 AM EDT Project Member

Status: Fixed (was: Assigned)
Owner: dsv@chromium.org

I can't reproduce the issue and I think my previous CLs that forbid connecting to the component extensions are enough as view-source: URLs are handled specially [1] and DevTools gets to see a URL without a view-source: prefix. I've added a test to confirm that and I think the issue could be closed.

[1]:
https://source.chromium.org/chromium/chromium/src/+/main:content/browser/renderer_host/navigation_controller_impl.cc;l=428;drc=7fb345a0da63049b102e1c0bcdc8d7831110e324;bpv=0;bpt=1

Comment 56 by sheriffbot on Mon, Apr 4, 2022, 12:42 PM EDT Project Member

Labels: reward-topanel

Comment 57 by sheriffbot on Mon, Apr 4, 2022, 1:40 PM EDT Project Member

Labels: -Restrict-View-SecurityTeam Restrict-View-SecurityNotify

Comment 58 by sheriffbot on Mon, Apr 4, 2022, 2:00 PM EDT Project Member

Labels: Merge-Request-101 Merge-Request-100

Requesting merge to stable M100 because latest trunk commit (988409) appears to be after stable branch point (972766).

Requesting merge to beta M101 because latest trunk commit (988409) appears to be after beta branch point (982481).

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 59 by sheriffbot on Tue, Apr 5, 2022, 4:37 AM EDT Project Member

Labels: -Merge-Request-101 Merge-Review-101 Hotlist-Merge-Review

Merge review required: M101 is already shipping to beta.

Please answer the following questions so that we can safely process your merge request:
1. Why does your merge fit within the merge criteria for these milestones?
- Chrome Browser: https://chromiumdash.appspot.com/branches
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines
2. What changes specifically would you like to merge? Please link to Gerrit.
3. Have the changes been released and tested on canary?
4. Is this a new feature? If yes, is it behind a Finch flag and are experiments active in any release channels?
5. [Chrome OS only]: Was the change reviewed and approved by the Eng Prod Representative?
https://goto.google.com/cros-engprodcomponents
6. If this merge addresses a major issue in the stable channel, does it require manual verification by the test team? If so, please describe required testing.

Please contact the milestone owner if you have questions.
Owners: benmason (Android), harrysouders (iOS), matthewjoseph (ChromeOS), pbommana (Desktop)

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 60 by sheriffbot on Tue, Apr 5, 2022, 4:37 AM EDT Project Member

Labels: -Merge-Request-100 Merge-Review-100

Merge review required: M100 is already shipping to stable.

Please answer the following questions so that we can safely process your merge request:
1. Why does your merge fit within the merge criteria for these milestones?
- Chrome Browser: https://chromiumdash.appspot.com/branches
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines
2. What changes specifically would you like to merge? Please link to Gerrit.
3. Have the changes been released and tested on canary?
4. Is this a new feature? If yes, is it behind a Finch flag and are experiments active in any release channels?
5. [Chrome OS only]: Was the change reviewed and approved by the Eng Prod Representative?
https://goto.google.com/cros-engprodcomponents
6. If this merge addresses a major issue in the stable channel, does it require manual verification by the test team? If so, please describe required testing.

Please contact the milestone owner if you have questions.
Owners: govind (Android), harrysouders (iOS), dgagnon (ChromeOS), srinivassista (Desktop)

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 61 by dsv@chromium.org on Tue, Apr 5, 2022, 4:46 AM EDT Project Member

Labels: -Merge-Review-100 -Merge-Review-101

No need to merge, the fix is https://chromiumdash.appspot.com/commit/09927ca7399717d3bb45c2948ffbb7682526b579 and is already in M100M.

Comment 62 by adetaylor@google.com on Fri, Apr 8, 2022, 12:41 PM EDT Project Member

Labels: Release-2-M100

(labelling as Release-2-M100 so this gets picked up in the next release notes, even though we fixed this a while back)

Comment 63 by adetaylor@google.com on Mon, Apr 11, 2022, 1:29 PM EDT Project Member

Labels: CVE-2022-1309 CVE_description-missing

Comment 64 by amyressler@google.com on Wed, Apr 13, 2022, 7:42 PM EDT Project Member

Labels: -reward-topanel reward-unpaid reward-15000

*** Boilerplate reminders! ***
Please do NOT publicly disclose details until a fix has been released to all our users. Early public disclosure may cancel the provisional reward. Also, please be considerate about disclosure when the bug affects a core library that may be used by other products. Please do NOT share this information with third parties who are not directly involved in fixing the bug. Doing so may cancel the provisional reward. Please be honest if you have already disclosed anything publicly or to third parties. Lastly, we understand that some of you are not interested in money. We offer the option to donate your reward to an eligible charity. If you prefer this option, let us know and we will also match your donation - subject to our discretion. Any rewards that are unclaimed after 12 months will be donated to a charity of our choosing.

Please contact security-vrp@chromium.org with any questions.
****

Comment 65 by amyressler@chromium.org on Wed, Apr 13, 2022, 7:46 PM EDT Project Member

Congratulations, David! The VRP Panel has decided to award you $15,000 for this report. Thanks for all your efforts and great work!

Comment 66 by amyressler@google.com on Fri, Apr 15, 2022, 10:03 PM EDT Project Member

Labels: -reward-unpaid reward-inprocess

Comment 67 by sheriffbot on Tue, Jul 12, 2022, 1:32 PM EDT Project Member

Labels: -Restrict-View-SecurityNotify allpublic

This bug has been closed for more than 14 weeks. Removing security view restrictions.

For more details visit https://www.chromium.org/issue-tracking/autotriage - Your friendly Sheriffbot

Comment 68 by amyressler@google.com on Tue, Jul 26, 2022, 4:57 PM EDT Project Member

Labels: CVE_description-submitted -CVE_description-missing

Comment 69 by amyressler@chromium.org on Fri, Jul 29, 2022, 5:27 PM EDT Project Member

Labels: -CVE_description-missing --CVE_description-missing