Starred by 5 users

Owner:	dcheng@chromium.org
CC:	falken@chromium.org danakj@chromium.org fergal@google.com adetaylor@chromium.org shimazu@chromium.org rsleevi@chromium.org altimin@chromium.org creis@chromium.org ndevtk@protonmail.com dcheng@chromium.org alex...@chromium.org arthu...@chromium.org amyressler@chromium.org nasko@chromium.org
Status:	Fixed *(Closed)*
Components:	UI>Browser>Navigation Blink>Loader
Modified:	May 22, 2021
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	2021-02-16
OS:	Linux, Windows, Chrome, Mac
Pri:	1
Type:	Bug-Security

Hotlist-Merge-Review
reward-3000
Security_Impact-Stable
Security_Severity-Medium
allpublic
reward-inprocess
CVE_description-submitted
M-89
Target-89
LTS-Security-86
LTS-Security-Failed-86
Release-0-M89
external_security_report
merge-merged-4389

Issue 1111646: Security: Possible to spoof URL after renderer crash

Reported by derce...@gmail.com on Thu, Jul 30, 2020, 11:43 PM EDT

Code

VULNERABILITY DETAILS
When a renderer crashes, if the window associated with it is navigated (by another window) to a URL that's ultimately blocked, an about:blank page will be loaded. That page will be same origin with the window that triggered the navigation, but the URL shown in the omnibox won't have changed.

This means that if one window can can cause another window to crash, it can spoof the URL in that window.

VERSION
Chrome Version: Tested on 84.0.4147.105 (stable) and 86.0.4217.2 (canary)
Operating System: Windows 10, version 1909

REPRODUCTION CASE
1. Download index.html and service_worker.js into a directory, then run the following command:

python3 -m http.server 8080 --bind 127.0.0.1

2. Next, download external_site.html into a directory and run:

python3 -m http.server 8080 --bind 127.0.0.2

3. In the browser, navigate to the following location:

http://127.0.0.1:8080/index.html

4. This page will register a service worker. Wait until the "Service worker registered" message appears in the devtools console.
5. Click the page.
6. Once clicked, the page will open http://127.0.0.2:8080/external_site.html in a new window (the target window).
7. The page will then navigate the target window to cross_origin_filesystem_redirect.html. The service worker will redirect this request to:

filesystem:http://127.0.0.2:8080/temporary/

Attempting to redirect to this filesystem: URL will cause the renderer associated with the target window to crash (due to a failed CHECK).

8. A second later, the page will navigate the target window to blocked_navigation.html, which the service worker will redirect to chrome://settings. This causes the navigation to be cancelled and an about:blank page to be loaded. This page will be same-origin with http://127.0.0.1:8080, but the URL in the omnibox won't have changed - it will still display http://127.0.0.2:8080/external_site.html.

To demonstrate that the original page has access to this window, it will make the following call:

targetWindow.document.write("Content set by " + location.href);

This text should display in the target window.

One other thing to note is that if the target page is a https page, the omnibox should still have the https padlock.

CREDIT INFORMATION
Reporter credit: David Erceg

external_site.html
264 bytes View Download

index.html
2.1 KB View Download

service_worker.js
657 bytes View Download

Comment 1 by derce...@gmail.com on Thu, Jul 30, 2020, 11:46 PM EDT

The reason the filesystem: navigation in step 7 causes the target renderer to crash is because of the presence of this CHECK:

```
CHECK(SecurityOrigin::Create(current_request_url)->CanDisplay(url_) ||
      !params_->web_bundle_claimed_url.IsNull());
```

https://source.chromium.org/chromium/chromium/src/+/master:third_party/blink/renderer/core/loader/document_loader.cc;l=798;drc=1dab3e3d4257cf8f13383ef08f05fa47b7dce698

Being able to crash the target renderer in this way relies on the target page having created a filesystem, as well as the fact that, currently, a service worker can redirect a request to a cross-origin filesystem: URL (see issue 1111213).

However, the URL spoof described above should work regardless of the method used to crash the target renderer. One way to check that is by going through the following steps:

1. Navigate to http://127.0.0.1:8080/index.html and open the devtools.
2. Run the following code:

```
let targetWindow = open("http://127.0.0.2:8080/external_site.html");
```

3. Kill the renderer process associated with the target window (e.g. via Chrome's task manager).
4. In the devtools for the original page, run:

```
targetWindow.location.href = "blocked_navigation.html";
```

5. Verify that you can interact with the target window from the original page and that the URL shown in the omnibox hasn't changed.

Comment 2 by rsleevi@chromium.org on Mon, Aug 3, 2020, 3:02 PM EDT

Status: Untriaged (was: Unconfirmed)
Owner: nasko@chromium.org
Labels: Security_Severity-Medium Security_Impact-Stable OS-Chrome OS-Linux OS-Mac OS-Windows
Components: Blink>Loader UI>Browser>Navigation

Thanks! I have trouble reliably reproducing this in version 84.0.4147.105 (Official Build) (64-bit), but I've confirmed that this sometimes works.

When it fails, I get the expected result, of the console showing:
index.html:41 Uncaught DOMException: Blocked a frame with origin "http://127.0.0.1:8080" from accessing a cross-origin frame.
at http://127.0.0.1:8080/index.html:41:34

That said, as I'm trying in a VM, there could be timing dependencies here.

Nasko: This seems the intersection of your expertise (and potentially intersecting with PlzNavigate?), I'm hoping you might have a pointer for a better owner. Git blame and OWNERS files weren't as much of a help here.

Comment 3 by creis@chromium.org on Mon, Aug 3, 2020, 3:23 PM EDT

Cc: creis@chromium.org falken@chromium.org shimazu@chromium.org
Components: Blink>ServiceWorker

derceg86: Is the service worker part of the repro required for it to work? I tried using a malformed blob URL in step 4 of comment 1 (to trigger the about:blank#blocked URL in the new window), and the URL bar updated. Maybe it's just failing to update in the case of service workers?

I haven't had time to try out the repro and need to run, but that may be worth checking. CC'ing falken@ and shimazu@ just in case.

Comment 4 by sheriffbot on Mon, Aug 3, 2020, 3:48 PM EDT

Status: Assigned (was: Untriaged)

Comment 5 by derce...@gmail.com on Tue, Aug 4, 2020, 12:09 AM EDT

Re #c2:

I see the same sort of flakiness within a VM. The demonstration here is timing dependent, in that the original page has to wait for events that I don't think it can directly detect. That is, it has to wait for the target page to load, then crash, then for the navigation to blocked_navigation.html to complete.

In a VM I have that has poor performance, the demonstration sometimes fails, because the navigations don't complete within the timeouts. However, the timeouts can simply be increased. For example, adjusting each of the three timeouts from 1 second to 10 seconds means that the demonstration works reliably within my VM.

However, 10 seconds is probably more than what would be needed in a realistic scenario, since the VM I'm using is significantly slower than a real-world machine.

Re #c3:

There is one way that I've found to trigger the issue without using a service worker in the final step. If you navigate to a site that takes a significant amount of time to return a response, then stop the load before it completes, the final state will be the same as in the demonstration above: an about:blank page will be loaded that's same-origin with the original page, but the URL in the omnibox won't have updated.

However, that's an imperfect way of doing it, since it seems like the original page can't cancel the load once it's started and instead, the user would have to manually do it.

You can check this by going through steps 1-3 in #c1, then completing the following:

1. Run:

```
{ echo -ne "HTTP/1.0 200 OK\r\nContent-Length: 0\r\n\r\n"; } | nc -l -p 8081 -i 300
```

This will start a netcat server that returns a delayed response.

2. In the devtools for the original page, run:

```
targetWindow.location.href = "http://127.0.0.1:8081/";
```

3. Switch to the target tab and click the stop button in the toolbar or press ESC.

4. Verify that you can interact with the target window from the original page and that the URL shown in the omnibox hasn't changed.

Comment 6 by creis@chromium.org on Tue, Aug 4, 2020, 12:30 PM EDT

Cc: alex...@chromium.org

alexmos@: Maybe this has to do with the optimization to commit the pending RFH early after a sad frame? I'm out for the moment and don't have that bug number handy, but comment 5 aludes to that.

Comment 7 by sheriffbot on Tue, Aug 4, 2020, 2:16 PM EDT

Labels: Target-85 M-85

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

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

Comment 8 by sheriffbot on Tue, Aug 4, 2020, 2:53 PM EDT

Labels: Pri-1

Setting Pri-1 to match security severity Medium. If this is incorrect, please reset the priority. Sheriffbot won't make this change again.

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

Comment 9 by alex...@chromium.org on Mon, Aug 10, 2020, 9:27 PM EDT

Cc: fergal@google.com

I've tried this out and can confirm the URL spoof behavior via the ServiceWorker. I didn't try to use the filesystem CHECK to get a crash in the renderer, as that seems like an unrelated issue (that we should also fix); I simply killed the renderer via task manager. My steps:

1. Go to http://127.0.0.1:8080/index.html
2. From DevTools:
  let targetWindow = open("http://csreis.github.io/");
3. From task manager, kill the csreis.github.io tab.
4. Back in first tab, from DevTools:
  targetWindow.location.href = "blocked_navigation.html";
5. Still in first tab:
  targetWindow.document.write("Content set by " + location.href);
6. Switch to second tab: it has "csreis.github.io" in the omnibox, but the body has content from 127.0.0.1:8080.

I tried a few other methods of getting into this situation and while most didn't work (even when the result ended up on about:blank, it was on an opaque origin that the first tab couldn't script), I did find another one that works without a ServiceWorker, which is via a same-origin 204 response. So, we could replace step 4 with
  targetWindow.location.href = "204.html";
and have the server respond to 204.html with a 204 status code. As an example, I did this by exiting the python web server, running "nc -l 8080", and responding with "HTTP/1.0 204 No Content" when the request comes in at step 4. The resulting page is also at about:blank, with the http://127.0.0.1:8080 origin.

So I do think there might be something here related to how we recover from crashes. I'm not sure whether this is due to immediately committing the speculative RFH vs just failing to update the omnibox if we recreate a renderer process but never commit anything in it, because when we recreate the renderer, we do actually create a document that's on about:blank, and the omnibox should probably reflect that (especially after content is injected into that document).

+fergal@ who's been looking into crashed frame recovery a lot lately in the context of RenderDocument. We should check if https://chromium-review.googlesource.com/c/chromium/src/+/2265893/ (removing the early CommitPending call for crashed frames in issue 1072817) is going to fix this.

Comment 10 by sheriffbot on Fri, Aug 14, 2020, 1:36 PM EDT

nasko: 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 11 by nasko@chromium.org on Fri, Aug 14, 2020, 1:38 PM EDT

Owner: alex...@chromium.org
Cc: -alex...@chromium.org nasko@chromium.org

Given that alexmos@ and fergal@ are more knowledgeable in this area, I'm re-assinging it, since I won't be able to work on this.

Comment 12 by fergal@google.com on Sat, Aug 15, 2020, 2:33 AM EDT

Sorry, I didn't try checking if https://crrev.com/c/2265893 plus its base fix this. I'm going to mostly OOO. Could we check in a failing test for it and I will see if that CL fixe it.

Comment 13 by sheriffbot on Fri, Aug 28, 2020, 1:37 PM EDT

alexmos: Uh oh! This issue still open and hasn't been updated in the last 17 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 14 by falken@chromium.org on Thu, Sep 10, 2020, 10:22 AM EDT

Components: -Blink>ServiceWorker

From what I've read above service worker is only tangentially related to this as the behavior can be triggered without it, so I'll drop the SW component.

Comment 15 by alex...@chromium.org on Thu, Sep 17, 2020, 1:01 PM EDT

Cc: altimin@chromium.org

I've spent some time looking at this issue. The basic setup is that foo.com window.opens bar.com, then bar.com crashes, then foo.com navigates its popup to foo.com/nocontent (or really anything that doesn't commit). The latter navigation creates a blank RenderFrame for the popup in the foo.com process, but doesn't commit anything in it and is canceled at OnResponseStarted time when we realize that there's nothing to commit due to a 204 response. However, the frame is in the foo.com process, with a foo.com origin, so the original foo.com frame can happily script it to inject content. At this point, because of the early CommitPending optimization for crashed frames, we now have a new RFH committed in the popup, but it doesn't have a last committed URL or origin. Further, there's no pending entry after the 204 response is processed, so none of the safe_to_show_pending logic in GetVisibleEntry applies. The popup is still on the NavigationEntry where the URL is "bar.com", and that's why the omnibox is showing "bar.com". Basically, the session history state has diverged from the current RFH and what it can be scripted by.

I've checked quickly what would happen if we skip the early CommitPending() call (i.e., with https://crrev.com/c/2265893, though the exact approach there is still in flux, so I only disabled the early CommitPending() logic without dealing with the rest of the fallout). The /nocontent navigation to foo.com would stay in a speculative RFH, and it would get canceled at OnResponseStarted, which would lead to the corresponding provisional frame being destroyed in blink. The tab stays at the sad tab graphic and at bar.com, and when the opener tries to script it, it gets a cross-origin exception -- presumably, the target frame is still a proxy with the bar.com origin. This seems like exactly the right behavior to me, so I think the work on issue 1072817 is going to fix this.

Now, the question is whether we want a shorter-term fix for this in the meantime, since removing the early CommitPending call is turning out to be really complicated, nor mergeable, and might still be some time away. Some ideas:
1. Commit the pending NavigationEntry when we commit the speculative RFH early (at that point, there presumably should be one, since we're still navigating), so the two can't diverge. Not clear if we then create another entry with replacement for the ongoing navigation?
2. Set the early-committed RFH's last committed origin to unique/opaque, both in browser and renderer processes, so that at least the recreated about:blank document can't be scripted until it really commits a navigation (if ever). Note that we should do this early enough -- doing this when we cancel a navigation on a RFH that committed early isn't sufficient, as the recreated blank frame could already have been scripted (e.g., while the server took a while to respond).
3. Have the omnibox recognize this case. Doesn't seem possible IIUC, as that goes off of the GetVisibleEntry (which is incorrect) rather than last committed URL (which is correctly empty at this point).
4. Something else?

+altimin@ who I think has also been worrying about canceling early-committed navigations in the context of issue 1072817 -- this is a good example of what can go wrong.

Comment 16 by creis@chromium.org on Thu, Sep 17, 2020, 11:11 PM EDT

Thanks for the analysis! I think skipping the early commit in issue 1072817 is a good plan, as long as we don't break things for the first navigation in a tab. Finding a shorter term fix may indeed be desirable, though.

Regarding your options:
(1) I don't think we can commit the pending NavigationEntry for a 204 / download URL. That seems like it would do unexpected things for back/forward/restore, like repeating a download.
(2) I'm intrigued by not allowing the frame to be scriptable, though I agree this is tricky.
(3) It seems like the omnibox code shouldn't have to keep track of this case, ideally, and we wouldn't want all of the other callers of GetVisibleEntry to be wrong.

Conceptually, the popup's history is: foo.com initial empty document -> bar.com (which crashes) -> foo.com/nocontent. If there were no crash, bar.com would navigate to foo.com/nocontent and stay on bar.com. In that sense, it's almost like we should recreate the frame in bar.com's process after the crash and then have foo.com navigate it as a RemoteFrame. If it failed, we would be left on an initial empty document in bar.com's process, without being scriptable by foo.com. (However, do we also need to care about a case where both frames are same-process but cross-origin, and the opener is restored before the popup?)

I like the idea of somehow not putting the tab into the state where it's showing a foo.com origin after the bar.com crash, since that's the closest to the non-crashed case. Just not sure if there's a good way to do that other than skipping the early commit. Maybe we can try to finish that up?

Comment 17 by fergal@google.com on Fri, Sep 18, 2020, 12:51 AM EDT

A new version of the early commit is here https://crrev.com/c/2409886. It depends on a refactor that I would not really like to cherry pick however that refactor is all about clarity and documentation. I think it's possible to land the change without it if that was necessary.

Comment 18 by fergal@chromium.org on Wed, Sep 23, 2020, 10:23 PM EDT

https://crrev.com/c/2426086 is the early commit change ported back to not use the refactor. There are several failing tests now (on both versions) which I will look into today.

Comment 19 by sheriffbot on Wed, Oct 7, 2020, 1:37 PM EDT

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

Comment 20 by alex...@chromium.org on Mon, Oct 26, 2020, 5:12 PM EDT

Blockedon: 1072817

Quick status update: https://chromium-review.googlesource.com/c/chromium/src/+/2426086 landed, which should address this issue by construction. However, it's running into some crashes, and fergal@ is continuing to investigate under issue 1072817. For now, let me mark that issue as a blocker here.

Comment 21 by sheriffbot on Fri, Oct 30, 2020, 6:46 PM EDT

Labels: reward-potential

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

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

Comment 23 by sheriffbot on Wed, Jan 20, 2021, 12:22 PM EST

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

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

Labels: -reward-potential external_security_report

Comment 25 by creis@chromium.org on Wed, Feb 3, 2021, 12:14 AM EST

Cc: danakj@chromium.org alex...@chromium.org rsleevi@chromium.org arthu...@chromium.org

~~issue 1172699~~ has been merged into this issue.

Comment 26 by creis@chromium.org on Wed, Feb 3, 2021, 12:18 AM EST

Cc: ndevtk@protonmail.com

CC'ing the reporter of ~~issue 1172699~~, who found that the same spoof is possible if the opener window navigates the crashed tab to a data: URL. This gets blocked (and thus doesn't commit), resulting in the same scriptable about:blank window with no corresponding NavigationEntry.

Alex/Fergal: It appears the RenderDocumentSkipEarlyCommit field trial for issue 1072817 does resolve this security bug, but we haven't enabled it by default. Can you give an update on the status of that? Do we need to look into an alternative fix in the meantime?

(I was hoping we could do something simple like update GetVisibleEntry() to notice when GetLastCommittedEntry() returns a NavigationEntry whose SiteInstance differs from the current RFH. That should detect this situation, but it doesn't give us great options about what to return-- I don't know what would happen if we fabricated a new about:blank NavigationEntry there for example. I'm also still nervous about actually committing something in NavigationController, which could affect the user's session history in weird ways.)

Comment 27 by creis@chromium.org on Wed, Feb 3, 2021, 12:24 AM EST

For reference, here's the simple repro steps I was using to simulate the renderer crash and spoof with a data: URL:
1) Visit http://csreis.github.io/
2) In DevTools: w = window.open("https://chromium.org")
3) Kill the chromium.org process in Chrome's task manager.
4) In the original tab: w.location = "data:text/html,foo"
5) w.document.write("foo")

Note that step 5 has started crashing in blink::BindingSecurity::FailedAccessCheckFor, at least if you do it in DevTools (crash/8190e4d08355c4fa). Not sure if that needs a bug of its own or if we expect that invariant won't fail after the early commit fix is enabled.

Comment 28 by creis@chromium.org on Wed, Feb 3, 2021, 1:36 AM EST

As one potential but ugly short-term fix, I thought we might return null from GetLastCommittedEntry() if its SiteInstance differed from the last committed RFH's SiteInstance, hoping that such a case only occurs during a weird broken invariant situation like this bug. We already have to handle a null GetLastCommittedEntry() in other cases and we know how to display about:blank for it, so it might work. Unfortunately, there's a bunch of calls to GetLastCommittedEntry() during cross-process navigations (after the RFH swap but before the last committed entry has been updated), and that code would break if we returned null rather than the previous entry.

Maybe it's marginally less of a problem to put the hack into GetVisibleEntry instead of GetLastCommittedEntry, but it still feels quite risky. Happy to discuss other ideas if we can't get the early commit fix enabled in the short term.

Comment 29 by fergal@google.com on Wed, Feb 3, 2021, 2:26 AM EST

Status of skip-early-commit is that it causes render crashes

https://crbug.com/1171246 (even after this is fixed, there is still a state machine violation, we will track that in a different bug).
https://crbug.com/1139104

I was stuck on progress on those and have not come back to them. It's possible that after danakj's fix, understanding the state machine problem might allow me to roll it out further.

Comment 30 by creis@chromium.org on Wed, Feb 3, 2021, 4:27 PM EST

One idea to consider: we might be able to use a post-commit NavigationEntry (like we use for subresource interstitials) to temporarily have something in place after the early commit optimization. It has the advantage of disappearing after you navigate to anything else. Haven't tried it yet (or considered whether this post-crash case could collide with an interstitial case), but thought I'd list it as an idea.

Comment 31 by creis@chromium.org on Thu, Feb 4, 2021, 2:54 AM EST

I'm trying out the idea from comment 30 in https://chromium-review.googlesource.com/c/chromium/src/+/2674708 as a short-term fix. There's still some edge cases that could be a problem, but we might be able to make it work until the early commit optimization is removed in issue 1072817.

Comment 32 by ndevtk@protonmail.com on Thu, Feb 4, 2021, 3:38 PM EST

I tried using the SkipEarlyCommitPendingForCrashedFrame feature from issue 1072817
It seems to fix the issue however its still possible to get a blank x tab using:
x.location = "https://www.google.com";
x.location = "data:text/html,<html>";
But I dont know if this is exploitable.

Comment 33 by creis@chromium.org on Tue, Feb 9, 2021, 2:22 AM EST

Status: Started (was: Assigned)
Owner: creis@chromium.org
Cc: dcheng@chromium.org

Comment 32: Thanks for noticing that! I don't think there's a security issue, but I've posted it to issue 1072817 so that it gets fixed as well before that mode is enabled.

The short term fix from comment 31 is looking promising, so I'll try to wrap that up and land it.

There's one interesting thing I noticed along the way-- we seemed to have started crashing in the renderer recently during the repro steps (e.g., step 5 in comment 27). I bisected it to r842470 (89.0.4387.0) for issue 1063150, where dcheng@ was cleaning up some frame swap logic. The bindings now crash because the target frame isn't found. I don't feel confident in it as a defense (since I'm able to still using document.write in a javascript: URL), so landing this fix still seems worthwhile. I do wonder whether that crash should be fixed as a separate bug, though-- maybe dcheng@ has thoughts on it, based on crash/8190e4d08355c4fa that I encountered? (Hopefully this weird renderer state won't stay possible too long, but it's possible for now and probably shouldn't crash.)

Comment 34 by dcheng@chromium.org on Wed, Feb 10, 2021, 1:24 AM EST

I've filed ~~issue 1176555~~ to track the crash.

Comment 35 by dcheng@chromium.org on Wed, Feb 10, 2021, 1:54 AM EST

OK, I wrote a fix for the crash. It has the side effect of making the initial empty document in the provisional frame have a unique opaque origin. Does that fix this bug by blocking the document.write?

Comment 36 by creis@chromium.org on Wed, Feb 10, 2021, 2:30 AM EST

Thanks! If that CL works out, it does sound promising for disrupting the crash from a non-exploited renderer. I think my https://chromium-review.googlesource.com/c/chromium/src/+/2674708 might still be useful against a compromised renderer (since we aim to be robust against URL spoofs from compromised renderers), but I would love to avoid inheriting the origin here for the common case.

Comment 37 by bugdroid on Wed, Feb 10, 2021, 3:11 AM EST

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/eaa61ce1dd876809d28ec3d44993c62f22a143b4

commit eaa61ce1dd876809d28ec3d44993c62f22a143b4
Author: Charlie Reis <creis@chromium.org>
Date: Wed Feb 10 08:10:58 2021

Hide previous NavigationEntry when doing an early RFH swap.

When the previous RFH has crashed and we eagerly swap in the
speculative RFH for a new navigation, ensure we do not show
the old URL above a potentially scriptable initial empty
document of the new RFH.

This approach can be removed after issue 1072817 is fixed.

~~Bug: 1111646~~, 1072817
Change-Id: I8ded08a40b6f3df17b072d66da31f728671e599f
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2674708
Commit-Queue: Charlie Reis <creis@chromium.org>
Reviewed-by: Alex Moshchuk <alexmos@chromium.org>
Cr-Commit-Position: refs/heads/master@{#852534}

[modify] https://crrev.com/eaa61ce1dd876809d28ec3d44993c62f22a143b4/content/browser/renderer_host/navigation_controller_impl_browsertest.cc
[modify] https://crrev.com/eaa61ce1dd876809d28ec3d44993c62f22a143b4/content/browser/renderer_host/navigation_controller_impl.cc
[modify] https://crrev.com/eaa61ce1dd876809d28ec3d44993c62f22a143b4/content/browser/renderer_host/navigation_controller_impl.h
[modify] https://crrev.com/eaa61ce1dd876809d28ec3d44993c62f22a143b4/content/browser/renderer_host/render_frame_host_manager.cc

Comment 38 by bugdroid on Fri, Feb 12, 2021, 12:21 AM EST

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/12b3d6403a58360e5d398072ba96e24ee67d6511

commit 12b3d6403a58360e5d398072ba96e24ee67d6511
Author: Daniel Cheng <dcheng@chromium.org>
Date: Fri Feb 12 05:17:57 2021

Ensure WindowProxy is initialized when early swapping RFH for navigation

When starting a navigation in a crashed frame, the navigation logic
immediately swaps in the RFH to avoid displaying a crashed frame while
the navigation is ongoing. Before r842470, swapping frames had the
additional side effect of initializing the WindowProxy. After r842470,
WindowProxy initialization when swapping in a LocalFrame is delayed
until a navigation committed, in order to avoid double-initializing the
WindowProxy. The fix is to simply also initialize the window proxy when
handling an early swap.

However, fixing this exposes the initial empty Document in a provisional

frame to the web. As this Document inherits the origin of the parent or
opener frame (when either parent or opener are present), what should be
a crashed frame suddenly becomes scriptable. To avoid exposing this
implementation detail to the web, disallow provisional frames from ever
inheriting an origin.

~~Bug: 1111646~~, ~~1176555~~
Change-Id: Ibb9145152 65cec45b6a9847f4c10d2afd58a009f
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2686905
Reviewed-by: Charlie Reis <creis@chromium.org>
Reviewed-by: Lucas Gadani <lfg@chromium.org>
Reviewed-by: Fergal Daly <fergal@chromium.org>
Commit-Queue: Daniel Cheng <dcheng@chromium.org>
Cr-Commit-Position: refs/heads/master@{#853403}

[modify] https://crrev.com/12b3d6403a58360e5d398072ba96e24ee67d6511/third_party/blink/renderer/core/loader/document_loader.cc
[modify] https://crrev.com/12b3d6403a58360e5d398072ba96e24ee67d6511/content/browser/site_per_process_browsertest.cc
[modify] https://crrev.com/12b3d6403a58360e5d398072ba96e24ee67d6511/third_party/blink/renderer/core/frame/web_frame_test.cc
[modify] https://crrev.com/12b3d6403a58360e5d398072ba96e24ee67d6511/third_party/blink/renderer/core/frame/local_frame.cc

Comment 39 by creis@chromium.org on Fri, Feb 12, 2021, 3:40 PM EST

dcheng@: Thanks for landing r853403 as well! That should also be a sufficient fix for cases without a compromised renderer, since the provisional frame will not be scriptable anymore.

That's a good thing, because I may need to do another round on my fix (r852534), after it caused some extension-related crashes in issue 1177398. I'll plan to revert and investigate it, hopefully re-landing it so that we can address this URL spoof for the compromised renderer case as well.

Comment 40 by bugdroid on Fri, Feb 12, 2021, 5:28 PM EST

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/d07032be266995a39f6f28e0a1d26eebe3762b01

commit d07032be266995a39f6f28e0a1d26eebe3762b01
Author: Charlie Reis <creis@chromium.org>
Date: Fri Feb 12 22:27:50 2021

Revert "Hide previous NavigationEntry when doing an early RFH swap."

This reverts commit eaa61ce1dd876809d28ec3d44993c62f22a143b4.

Reason for revert: Caused crashes for extension navigations in
https://crbug.com/1177398.

Original change's description:
> Hide previous NavigationEntry when doing an early RFH swap.
>
> When the previous RFH has crashed and we eagerly swap in the
> speculative RFH for a new navigation, ensure we do not show
> the old URL above a potentially scriptable initial empty
> document of the new RFH.
>
> This approach can be removed after issue 1072817 is fixed.
>
> ~~Bug: 1111646~~, 1072817
> Change-Id: I8ded08a40b6f3df17b072d66da31f728671e599f
> Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2674708
> Commit-Queue: Charlie Reis <creis@chromium.org>
> Reviewed-by: Alex Moshchuk <alexmos@chromium.org>
> Cr-Commit-Position: refs/heads/master@{#852534}

TBR=creis@chromium.org,alexmos@chromium.org,chromium-scoped@luci-project-accounts.iam.gserviceaccount.com

Not skipping CQ checks because original CL landed > 1 day ago.

~~Bug: 1111646~~
Bug: 1072817
Change-Id: Ic9cebd5c4de4e4fcfcae54a81112dd70b60cdd6a
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2693218
Reviewed-by: Charlie Reis <creis@chromium.org>
Commit-Queue: Charlie Reis <creis@chromium.org>
Cr-Commit-Position: refs/heads/master@{#853692}

[modify] https://crrev.com/d07032be266995a39f6f28e0a1d26eebe3762b01/content/browser/renderer_host/navigation_controller_impl_browsertest.cc
[modify] https://crrev.com/d07032be266995a39f6f28e0a1d26eebe3762b01/content/browser/renderer_host/navigation_controller_impl.cc
[modify] https://crrev.com/d07032be266995a39f6f28e0a1d26eebe3762b01/content/browser/renderer_host/navigation_controller_impl.h
[modify] https://crrev.com/d07032be266995a39f6f28e0a1d26eebe3762b01/content/browser/renderer_host/render_frame_host_manager.cc

Comment 41 by dcheng@chromium.org on Fri, Feb 12, 2021, 7:46 PM EST

I've done some local testing, and I can confirm that r853403 by itself addresses the URL spoof as presented.

While the omnibox does *not* display the correct URL, the document associated with the incorrect URL has a unique opaque origin now, and can no longer be manipulated in the manner described in this bug.

Note: the full fix for r853403 cannot be merged back to before M89, since it has a dependency on some changes that first landed in M89.

Comment 42 by creis@chromium.org on Fri, Feb 12, 2021, 7:57 PM EST

Status: Fixed (was: Started)
Owner: dcheng@chromium.org
Labels: -Target-88 -Target-85 -Target-86 -Target-87 Target-89
NextAction: 2021-02-16

Thanks! Let's consider that the fix for this, and I can follow up separately with my CL to prevent the attack from a compromised renderer as well.

I'll mark it fixed, and we can let it bake over the weekend and request merge to M89 if it looks good next week? I'll be OOO and will rely on you for the merge, but I can discuss any followup issues when I return.

Comment 43 by creis@chromium.org on Fri, Feb 12, 2021, 8:03 PM EST

I can also confirm on tip-of-tree that the repro steps in comment 27 are disrupted with just r853403 in place:
w.document.write("foo")
VM20:1 Uncaught DOMException: Blocked a frame with origin "http://csreis.github.io" from accessing a cross-origin frame.
at <anonymous>:1:3

Thanks!

Comment 44 by sheriffbot on Sun, Feb 14, 2021, 12:42 PM EST

Labels: reward-topanel

Comment 45 by sheriffbot on Sun, Feb 14, 2021, 1:56 PM EST

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

Comment 46 by sheriffbot on Sun, Feb 14, 2021, 2:21 PM EST

Labels: Merge-Request-89

Requesting merge to beta M89 because latest trunk commit (852534) appears to be after beta branch point (843830).

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

Comment 47 by sheriffbot on Sun, Feb 14, 2021, 2:22 PM EST

Labels: -Merge-Request-89 Merge-Review-89 Hotlist-Merge-Review

This bug requires manual review: Reverts referenced in bugdroid comments after merge request.
Before a merge request will be considered, the following information is required to be added to this bug:

1. Does your merge fit within the Merge Decision Guidelines?
- Chrome: https://chromium.googlesource.com/chromium/src.git/+/master/docs/process/merge_request.md#when-to-request-a-merge
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines
2. Links to the CLs you are requesting to merge.
3. Has the change landed and been verified on ToT?
4. Does this change need to be merged into other active release branches (M-1, M+1)?
5. Why are these changes required in this milestone after branch?
6. Is this a new feature?
7. If it is a new feature, is it behind a flag using finch?

Chrome OS Only:
8. Was the change reviewed and approved by the Eng Prod Representative? See Eng Prod ownership by component: http://go/cros-engprodcomponents

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

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

Comment 48 by dcheng@chromium.org on Tue, Feb 16, 2021, 3:37 PM EST

1. Does your merge fit within the Merge Decision Guidelines?
- Chrome: https://chromium.googlesource.com/chromium/src.git/+/master/docs/process/merge_request.md#when-to-request-a-merge
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines

Sorry, I never really know how to answer this question :)
This is an externally reported security bug with URL spoof potential. I think we'd like to at least have it fixed in M89 which is rolling to stable soon.

2. Links to the CLs you are requesting to merge.

https://crrev.com/12b3d6403a58360e5d398072ba96e24ee67d6511

3. Has the change landed and been verified on ToT?

Yes

4. Does this change need to be merged into other active release branches (M-1, M+1)?

In theory, M88, but it's kind of late.

5. Why are these changes required in this milestone after branch?

Because the fix landed after the branch.

6. Is this a new feature?

No.

7. If it is a new feature, is it behind a flag using finch?

n/a

Comment 49 by pbommana@google.com on Tue, Feb 16, 2021, 4:43 PM EST

Cc: adetaylor@chromium.org amyressler@chromium.org

+Security TPMs for M89 merge review.

Comment 50 Deleted

Comment 51 by adetaylor@google.com on Wed, Feb 17, 2021, 6:34 PM EST

Approving merge to M89, branch 4389 (It's unlikely that there will be another M88 release.)

Comment 52 by amyressler@google.com on Wed, Feb 17, 2021, 7:12 PM EST

Labels: -reward-topanel reward-unpaid reward-3000

*** 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 53 by amyressler@google.com on Wed, Feb 17, 2021, 7:40 PM EST

Congratulations, David! The VRP Panel has decided to award you $3,000 for this report. Thank you for this submission!

Comment 54 by ndevtk@protonmail.com on Wed, Feb 17, 2021, 10:39 PM EST

Congratulations, David

Comment 55 by bugdroid on Thu, Feb 18, 2021, 8:48 PM EST

Labels: -merge-approved-89 merge-merged-89 merge-merged-4389

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/e46c7b908d96348e74b4657ecb3fc2570e9b9b93

commit e46c7b908d96348e74b4657ecb3fc2570e9b9b93
Author: Daniel Cheng <dcheng@chromium.org>
Date: Fri Feb 19 01:47:46 2021

Ensure WindowProxy is initialized when early swapping RFH for navigation

When starting a navigation in a crashed frame, the navigation logic immediately swaps in the RFH to avoid displaying a crashed frame while the navigation is ongoing. Before r842470, swapping frames had the additional side effect of initializing the WindowProxy. After r842470, WindowProxy initialization when swapping in a LocalFrame is delayed until a navigation committed, in order to avoid double-initializing the WindowProxy. The fix is to simply also initialize the window proxy when handling an early swap.

However, fixing this exposes the initial empty Document in a provisional frame to the web. As this Document inherits the origin of the parent or opener frame (when either parent or opener are present), what should be a crashed frame suddenly becomes scriptable. To avoid exposing this implementation detail to the web, disallow provisional frames from ever inheriting an origin.

(cherry picked from commit 12b3d6403a58360e5d398072ba96e24ee67d6511)

~~Bug: 1111646~~, ~~1176555~~
Change-Id: Ibb914515265cec45b6a9847f4c10d2afd58a009f
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2686905
Reviewed-by: Charlie Reis <creis@chromium.org>
Reviewed-by: Lucas Gadani <lfg@chromium.org>
Reviewed-by: Fergal Daly <fergal@chromium.org>
Commit-Queue: Daniel Cheng <dcheng@chromium.org>
Cr-Original-Commit-Position: refs/heads/master@{#853403}
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/2704529
Auto-Submit: Daniel Cheng <dcheng@chromium.org>
Bot-Commit: Rubber Stamper <rubber-stamper@appspot.gserviceaccount.com>
Cr-Commit-Position: refs/branch-heads/4389@{#1195}
Cr-Branched-From: 9251c5db2b6d5a59fe4eac7aafa5fed37c139bb7-refs/heads/master@{#843830}

[modify] https://crrev.com/e46c7b908d96348e74b4657ecb3fc2570e9b9b93/third_party/blink/renderer/core/loader/document_loader.cc
[modify] https://crrev.com/e46c7b908d96348e74b4657ecb3fc2570e9b9b93/content/browser/site_per_process_browsertest.cc
[modify] https://crrev.com/e46c7b908d96348e74b4657ecb3fc2570e9b9b93/third_party/blink/renderer/core/frame/web_frame_test.cc
[modify] https://crrev.com/e46c7b908d96348e74b4657ecb3fc2570e9b9b93/third_party/blink/renderer/core/frame/local_frame.cc

Comment 56 by awhalley@google.com on Fri, Feb 19, 2021, 5:37 PM EST
Labels: -reward-unpaid reward-inprocess

Comment 57 by adetaylor@google.com on Fri, Feb 26, 2021, 1:08 PM EST
Labels: Release-0-M89

Comment 58 by adetaylor@google.com on Mon, Mar 1, 2021, 7:27 PM EST
Labels: CVE-2021-21170 CVE_description-missing

Comment 59 by vsavu@google.com on Wed, Mar 3, 2021, 5:58 AM EST
Labels: LTS-Merge-Request-86

Comment 60 by vsavu@google.com on Wed, Mar 3, 2021, 6:02 AM EST
Labels: LTS-Security-86

Comment 61 by gianluca@google.com on Wed, Mar 3, 2021, 10:38 AM EST
Labels: LTS-Merge-Approved-86

Comment 62 by sheriffbot on Wed, Mar 3, 2021, 12:22 PM EST
Labels: -M-88 M-89

Comment 63 by vsavu@google.com on Thu, Mar 4, 2021, 10:39 AM EST
Labels: -LTS-Merge-Approved-86 -LTS-Merge-Request-86 LTS-Security-Failed-86

Comment 64 by amyressler@google.com on Tue, Mar 9, 2021, 12:58 PM EST
Labels: -CVE_description-missing CVE_description-submitted

Comment 65 by sheriffbot on Sat, May 22, 2021, 1:49 PM EDT
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