Starred by 2 users

Owner:	wanderview@chromium.org **OOO, back Jan 3**
CC:	adetaylor@chromium.org kinuko@chromium.org nhiroki@chromium.org jarrydg@chromium.org staphany@chromium.org jsb...@chromium.org mek@chromium.org pwnall@chromium.org
Status:	Fixed *(Closed)*
Components:	Blink>Storage>CacheStorage Blink>Storage>Quota
Modified:	Apr 14, 2020
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	----
OS:	Linux, Android, Windows, Chrome, Mac
Pri:	2
Type:	Bug-Security

reward-500
Security_Severity-Low
Security_Impact-Stable
allpublic
reward-inprocess
CVE_description-submitted
M-81
Release-0-M81
CVE-2020-6442

Issue 1013906: Security: expose stored (in cache) cross-site response's size

Reported by bar...@gmail.com on Sat, Oct 12, 2019, 6:26 PM EDT

Code

VULNERABILITY DETAILS

I think that there is a way to expose the size of cross-site responses using storing them in a cache.

Intro:
1) XS-Search is an attack that tried to distinguish between responses that had search results and responses that did not. Most of the known XS-Search attacks are time-based. The attack here is not Time based but Length based.

2) The vulnerable code still exists in the last version of chrome that I've checked (73.0.3683.103).

Since version 63, a padding mechanism to cross-site responses that are stored in the cache was developed and integrated. I found a way to bypass the padding mechanism to launch another XS-Search attack, by sending 2 different requests to the same URL.

In this padding mechanism, the response size saved is the sum of the actual response size plus an additional random and unpredictable amount, I refer to as PaddingSize. The padding mechanism prevents Quota Management API from revealing the response's size by adding the PaddingSize to the actual size.

The PaddingSize is calculated as follows:
1. hmac sha256 function is calculated to the request URL using a 128-bit key.
2. The first 8 bytes of the MAC are copied to an uint64 variable.
3. Modulo MAX_PADDING is calculated to meet the constraints of max Padding Size.

The code the calculates the padding size is in cache_storage_cache.cc file on function ComputeResponsePadding(Last version that I've checked is 77.0.3865.112 and the code is here: https://chromium.googlesource.com/chromium/src/+/refs/tags/77.0.3865.112/storage/browser/quota/padding_key.cc)

In light of the above, it can be inferred that responses originating from the same URL will get the same padding size, while responses originating from different URLs will get different padding sizes.

I investigated dozens of search engines on different websites and checked which page was received when we searched a query without an authenticated user (i.e. without sending cookies that identified the user). In 95% of the observed cases, the response was a page that was similar or identical to a page that was received for an authenticated user's search query bearing 0 search results.

In the attached code, I chose to exploit YouTube watching history search interface (https://www.youtube.com/feed/history?query=QUERY) but it can exploit any similar search interface.

Now to the exploit steps:
1) Sample the usage using Quota API.
2) Send a request to the attacked URL without cookies and store the response in the cache. The response is a no-results page, due to the non-authentication.
3) Sample the usage using Quota API again. This discloses the sum of the response size (that bore 0 search results) and the padding size – let it be X.
4) Send a request to the same URL, this time with cookies that identify the victim, and store the response in the cache. This response holds search results from the victim's account, containing private information.
5) Sample the usage with Quota API once more. This discloses the sum of the response size that did bear search results and the padding size – let it be Y.
(note that since the URL did not change, the padding size remained the same).
7) calculate the subtraction of Y and X: Y-X.
8) Y-X is the size of the search records in the response.

Using the size of the search records we can infer how many records returned to the query and by that expose private information in that specific case - watching history.
We can know if the user like a specific band, intersting in a specific subject, and etc.

Note that the exploit works only in versions earlier to version 67. Since version 67, CORB denies suspicious cross-site requests. The attached exploit will work only if a CORB bypass exploit will be found.
Chrome designed with multiple layers of security protections and that why I think you should fix that vulnerability anyway.

Please feel free to contact me if you need any further information.

VERSION
Chrome Version: 65.0.3325.181
Operating System: Win 7

REPRODUCTION CASE
You can use the attached files to expose YouTube watch history of logged in user
The result is how many videos the user watched for the search query.

CREDIT INFORMATION
Reporter credit: B@rMey

index.html
7.8 KB View Download

k2.js
2.7 KB View Download

Comment 1 by ajgo@google.com on Mon, Oct 14, 2019, 1:36 AM EDT

Labels: Security_Severity-Low

Thanks for the report. I will look at reproducing it on Monday. Marking as Low as this must be combined with a CORB bypass to be effective.

Comment 2 by sheriffbot@chromium.org on Mon, Oct 14, 2019, 10:30 AM EDT

Labels: Pri-2

Setting Pri-2 to match security severity Low. 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 3 by ajgo@google.com on Mon, Oct 14, 2019, 5:37 PM EDT

Owner: pwnall@chromium.org
Cc: jarrydg@chromium.org jsb...@chromium.org kinuko@chromium.org nhiroki@chromium.org pwnall@chromium.org staphany@chromium.org
Labels: Security_Impact-Stable
Components: Blink>Storage>Quota Blink>Storage

pwnall@ could you take a look at this report? This is not easily reproducible as CORB blocks the requests.

Comment 4 by sheriffbot@chromium.org on Tue, Oct 15, 2019, 11:18 AM EDT

Status: Assigned (was: Unconfirmed)

Comment 5 by jsb...@chromium.org on Tue, Oct 15, 2019, 6:29 PM EDT

Cc: wanderview@chromium.org
Labels: OS-Android OS-Chrome OS-Linux OS-Mac OS-Windows
Components: -Blink>Storage Blink>Storage>CacheStorage

Comment 6 by jsb...@chromium.org on Tue, Oct 15, 2019, 6:41 PM EDT

Cc: mek@chromium.org

Comment 7 by wanderview@chromium.org on Tue, Oct 15, 2019, 7:33 PM EDT

One possible solution here would be to include the credentials flag in the hash computation. So url+no-credentials would have one stable hash and url+credentials would have a different stable hash.

This wouldn't protect against other potential cases where an attacker can force a resource to change size in a predictable way via other means, though. Maybe that's ok, though, since logged in state is the most important piece of state to protect.

Comment 8 by pwnall@chromium.org on Wed, Oct 16, 2019, 5:17 PM EDT

Owner: wanderview@chromium.org
Cc: -wanderview@chromium.org

Agreed that this is a good step forward in the mitigation.

wanderview@: Can you please implement this fix? Please LMK if you can't fit it on your plate, and I'll figure something out.

Comment 9 by wanderview@chromium.org on Fri, Oct 25, 2019, 9:58 AM EDT

I will try to fit this into my list for M80. If this is not fast enough, we may need to find someone else to tackle it.

Comment 10 by wanderview@chromium.org on Fri, Dec 6, 2019, 4:47 PM EST

CL up for review: https://chromium-review.googlesource.com/c/chromium/src/+/1956044

This CL sets a "credentialed" flag in the fetch_manager based on the request credentials mode and the tainting. For opaque responses we will only get a credentialed state if the mode is 'include'.

The flag is then plumbed through the different fetch serialization types and down to the serialized cache_storage protobuf. We treat older stored responses as simply non-credentialed. I don't think its worth doing a migration of old data since this a defense in depth issue and not directly exploitable.

Finally the flag is used to add additional input to the padding hash calculation.

The CL includes both a unit test and an end-to-end web_test.

Comment 11 by bugdroid on Mon, Dec 9, 2019, 6:21 PM EST

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src.git/+/64d2022359fed5668c631dda83cda2dfd7e956ce

commit 64d2022359fed5668c631dda83cda2dfd7e956ce
Author: Ben Kelly <wanderview@chromium.org>
Date: Mon Dec 09 23:20:56 2019

CacheStorage: Vary opaque padding based on if a response was loaded with credentials.

~~Bug: 1013906~~

Change-Id: Ib65cd440ede20c79affb063ae458dff449dd814d
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/1956044
Commit-Queue: Ben Kelly <wanderview@chromium.org>
Reviewed-by: Victor Costan <pwnall@chromium.org>
Reviewed-by: Kinuko Yasuda <kinuko@chromium.org>
Cr-Commit-Position: refs/heads/master@{#723150}

[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/browser/appcache/appcache_backfillers.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/browser/appcache/appcache_update_job.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/browser/cache_storage/cache_storage.proto
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/browser/cache_storage/cache_storage_cache_unittest.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/browser/cache_storage/cache_storage_manager_unittest.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/browser/cache_storage/legacy/legacy_cache_storage_cache.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/content/common/background_fetch/background_fetch_types.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/storage/browser/quota/padding_key.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/storage/browser/quota/padding_key.h
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/third_party/blink/public/mojom/fetch/fetch_api_response.mojom
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/third_party/blink/renderer/core/fetch/fetch_manager.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/third_party/blink/renderer/core/fetch/fetch_response_data.cc
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/third_party/blink/renderer/core/fetch/fetch_response_data.h
[modify] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/third_party/blink/renderer/core/fetch/response.cc
[add] https://crrev.com/64d2022359fed5668c631dda83cda2dfd7e956ce/third_party/blink/web_tests/http/tests/cachestorage/padding.html

Comment 12 by bar...@gmail.com on Mon, Dec 9, 2019, 7:02 PM EST

Looks great.
Just to make sure we are on the same page, this mitigation is not effective against an attacker that can control the resource size via other means.

By the way, any chance for a bounty?

Comment 13 by pwnall@chromium.org on Mon, Dec 9, 2019, 7:06 PM EST

Cc: adetaylor@chromium.org

+adetaylor@ for the bounty question

Comment 14 by adetaylor@chromium.org on Mon, Dec 9, 2019, 7:08 PM EST

bar320@ It looks like it's going to result in a fix to Chrome which we wouldn't otherwise have made, so it will go to the VRP panel for consideration. It may well take a few weeks for them to get to it. Thanks for the report!

Comment 15 Deleted

Comment 16 by wanderview@chromium.org on Tue, Dec 10, 2019, 5:02 PM EST

Status: Fixed (was: Assigned)

> Just to make sure we are on the same page, this mitigation is not effective against an attacker that can control the resource size via other means.

Correct. This protects mutation based on credentials which tends to map to logged-in state which seems the most important aspect to protect.

Comment 17 by wanderview@chromium.org on Tue, Dec 10, 2019, 5:19 PM EST

If you feel there is another easy avenue of controlling response size, please re-open or file another bug. Thanks.

Comment 18 by sheriffbot@chromium.org on Wed, Dec 11, 2019, 10:51 AM EST

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

Comment 19 by awhalley@chromium.org on Wed, Dec 11, 2019, 5:53 PM EST

Labels: reward-topanel

Comment 20 by natashapabrai@google.com on Thu, Dec 19, 2019, 12:34 PM EST

Labels: -reward-topanel reward-unpaid reward-500

*** 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 21 by natashapabrai@google.com on Thu, Dec 19, 2019, 12:41 PM EST

Congrats! The Panel decided to reward $500 for this report!

Comment 22 by natashapabrai@google.com on Thu, Dec 19, 2019, 12:47 PM EST

Labels: -reward-unpaid reward-inprocess

Comment 23 by bar...@gmail.com on Wed, Feb 5, 2020, 8:56 AM EST

1) When the fix will be released? I'm asking because I would like to publish my paper about cross-site search attacks.
2) Do you assign a CVE for this bug?

Comment 24 by wanderview@chromium.org on Wed, Feb 5, 2020, 10:51 AM EST

Labels: M-81

The fix is in M81 which will not be released to stable until around March 17, 2020. I do not know about the CVE question, but hopefully someone from security will respond.

Comment 25 by adetaylor@chromium.org on Wed, Feb 5, 2020, 4:53 PM EST

bar320@ thanks again for the report! Once this is released in M81, it will be credited in the release notes and we'll allocate a CVE at that time.

Comment 26 by adetaylor@google.com on Fri, Mar 13, 2020, 1:44 PM EDT

Labels: Release-0-M81

Comment 27 by adetaylor@chromium.org on Fri, Mar 13, 2020, 2:31 PM EDT

Labels: CVE-2020-6442 CVE_description-missing

Comment 28 by sheriffbot on Wed, Mar 18, 2020, 1:54 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

Comment 29 by adetaylor@chromium.org on Tue, Apr 14, 2020, 3:14 PM EDT
Labels: -CVE_description-missing CVE_description-submitted