Starred by 4 users

Owner:	marja@chromium.org
CC:	adetaylor@chromium.org syg@chromium.org srinivassista@chromium.org mslekova@chromium.org ishell@chromium.org amyressler@chromium.org vahl@chromium.org leszeks@chromium.org verwa...@chromium.org ecmziegler@google.com
Status:	Verified *(Closed)*
Components:	Blink>JavaScript>Runtime
Modified:	Jul 29, 2022
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	----
OS:	Linux, Android, Windows, Chrome, Mac, Fuchsia, Lacros
Pri:	1
Type:	Bug-Security

M-100
Merge-Merged
Security_Severity-High
allpublic
ClusterFuzz-Verified
CVE_description-submitted
Target-100
FoundIn-98
FoundIn-101
Security_Impact-Extended
merge-merged-9.6
V8-postmortem
LTS-Merge-Merged-96
merge-merged-10.0
merge-merged-10.1
merge-merged-4962

Release-0-M100
CVE-2022-1134

Issue 1308360: Type confusion when using simple api call accessors with SuperIC

Reported by m...@semmle.com on Mon, Mar 21, 2022, 7:55 AM EDT

Code

Vulnerability Details

When accessing properties using accessors, the holder of the accessors may be in the prototype of the object, but the accessors will be applied to the receiver itself. Many blink objects have accessors that are simple api calls and these accessors have specific signatures meaning that they can only operate on v8 object of the expected type. When a simple api call accessor is encountered during the creation of an IC handler, the expected type of the accessor will be checked against the `lookup_start_object_map` [1]:

```
      CallOptimization call_optimization(isolate(), getter);
      if (call_optimization.is_simple_api_call()) {
        CallOptimization::HolderLookup holder_lookup;
        Handle<JSObject> api_holder =
            call_optimization.LookupHolderOfExpectedType(isolate(), map,
                                                         &holder_lookup);    //<---------- Checks that map is compatible with the expected type of the simple_api_call
```

The map used here is the `lookup_start_object_map` [2]. On the other hand, when using the IC handler, the accessor is used with the `receiver`, instead of the `lookup_start_object`[3]:

```
void AccessorAssembler::HandleLoadAccessor(
    const LazyLoadICParameters* p, TNode<CallHandlerInfo> call_handler_info,
    TNode<WordT> handler_word, TNode<DataHandler> handler,
    TNode<IntPtrT> handler_kind, ExitPoint* exit_point) {
  Comment("api_getter");
  ...
  BIND(&load);
  TNode<IntPtrT> argc = IntPtrConstant(0);
  exit_point->Return(CallApiCallback(context, callback, argc, data,
                                     api_holder.value(), p->receiver()));  //<------- accessor used with receiver
```

This is normally correct. However, in the case of a super property access, the `lookup_start_object` is not the same as the `receiver`, so by creating an IC handler for the `lookup_start_object` that passes the signature test, and then uses this IC handler in a super ic call with an incompatible `receiver`, a type confusion occurs and a simple api accessor will be called with an incompatible receiver.

Reproduction Case

Host the attached files `superic.html` and open it in Chrome

./out/Release/chrome --user-data-dir=/tmp

I've tested this on release build of commit 76ae53f on the master branch and on 98.0.4758.102. If successful, this would crash the renderer tab

crash the renderer tab.

Thank you very much for your help and please let me know if there is anything that I can help. Thanks.

VERSION
Chrome version: master branch build 76ae53f and 98.0.4758.102
Operating System: Ubuntu 20.04

1. https://source.chromium.org/chromium/chromium/src/+/dd1159653baab787bc341ddbf42af5aeab3c1634:v8/src/ic/ic.cc;l=1055
2. https://source.chromium.org/chromium/chromium/src/+/dd1159653baab787bc341ddbf42af5aeab3c1634:v8/src/ic/ic.cc;l=963
3. https://source.chromium.org/chromium/chromium/src/+/af93b3d584c22547ae5d6c49c56df07f2f7a2ca5:v8/src/ic/accessor-assembler.cc;l=285

CREDIT INFORMATION
Reporter credit: Man Yue Mo of GitHub Security Lab

superic.html
470 bytes View Download

Comment 1 by ClusterFuzz on Mon, Mar 21, 2022, 4:02 PM EDT

ClusterFuzz is analyzing your testcase. Developers can follow the progress at https://clusterfuzz.com/testcase?key=5712077410598912.

Comment 2 by ClusterFuzz on Mon, Mar 21, 2022, 11:55 PM EDT

Labels: OS-Windows

Comment 3 by ClusterFuzz on Tue, Mar 22, 2022, 1:42 AM EDT

Labels: OS-Linux

Comment 4 by ClusterFuzz on Tue, Mar 22, 2022, 1:26 PM EDT

Labels: FoundIn-101 Security_Impact-Head Security_Severity-Medium

Detailed Report: https://clusterfuzz.com/testcase?key=5712077410598912

Fuzzer: None
Job Type: linux_asan_chrome_mp
Platform Id: linux

Crash Type: UNKNOWN READ
Crash Address: 0x00017fff802e
Crash State:
  blink::v8_offline_audio_context::LengthAttributeGetCallback
  v8::internal::Invoke
  v8::internal::Execution::CallScript

Sanitizer: address (ASAN)

Recommended Security Severity: Medium

Regressed: https://clusterfuzz.com/revisions?job=linux_asan_chrome_mp&range=866941:866942

Reproducer Testcase: https://clusterfuzz.com/download?testcase_id=5712077410598912

To reproduce this, please build the target in this report and run it against the reproducer testcase. Please use the GN arguments provided at bottom of this report when building the binary.

If you have trouble reproducing, please also export the environment variables listed under "[Environment]" in the crash stacktrace.

If you have any feedback on reproducing test cases, let us know at https://forms.gle/Yh3qCYFveHj6E5jz5 so we can improve.

A recommended severity was added to this bug. Please change the severity if it is inaccurate.

Comment 5 by adetaylor@google.com on Tue, Mar 22, 2022, 5:57 PM EDT

Status: Assigned (was: Unconfirmed)
Owner: verwa...@chromium.org
Cc: ishell@chromium.org
Labels: FoundIn-98 OS-Android OS-Chrome OS-Fuchsia OS-Mac OS-Lacros Pri-1
Components: Blink>JavaScript>Runtime

Thanks for the report mmo@!

It's not clear why CF reckons that the regression range in #c4 is 101, so correcting the FoundIn label to say this affects anything from extended stable onwards.

I _think_ (from some of vahl's slides) that V8 runtime issues should go to verwaest@ - Toon, please reassign if appropriate!

Comment 6 by sheriffbot on Tue, Mar 22, 2022, 6:00 PM EDT

Labels: -Security_Impact-Head Security_Impact-Extended

Comment 7 by ishell@chromium.org on Wed, Mar 23, 2022, 5:11 AM EDT

Owner: marja@chromium.org

Comment 8 by marja@chromium.org on Wed, Mar 23, 2022, 5:24 AM EDT

Status: Started (was: Assigned)

Yep, this is mine. Looking...

Comment 9 by marja@chromium.org on Wed, Mar 23, 2022, 7:04 AM EDT

Thanks for the report & analysis. The bug is exactly where the OP said it is. I quickly checked and there doesn't seem to be related bugs (like, receiver vs lookup start object vs holder confusion on the handler creation side), so we should be good to go with the simple fix.

https://chromium-review.googlesource.com/c/v8/v8/+/3545172

Comment 10 by Git Watcher on Wed, Mar 23, 2022, 8:27 AM EDT

The following revision refers to this bug:

https://chromium.googlesource.com/v8/v8/+/9c3d4b3556b2797fa9d9f4bee915e8502608312f

commit 9c3d4b3556b2797fa9d9f4bee915e8502608312f
Author: Marja Hölttä <marja@chromium.org>
Date: Wed Mar 23 11:01:47 2022

[super IC] Fix receiver vs lookup start object confusion

~~Bug: v8:9237~~,~~chromium:1308360~~
Change-Id: I11e3c14a6cecb9d88a834711fb6252191494d5f7
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3545172
Reviewed-by: Igor Sheludko <ishell@chromium.org>
Commit-Queue: Marja Hölttä <marja@chromium.org>
Cr-Commit-Position: refs/heads/main@{#79571}

[modify] https://crrev.com/9c3d4b3556b2797fa9d9f4bee915e8502608312f/src/ic/accessor-assembler.cc

Comment 11 by m...@semmle.com on Wed, Mar 23, 2022, 9:58 AM EDT

Thanks for looking into this. I have a few comments about the issue and the patch.
1. I have tested this issue on both 98.0.4758.102 and 99.0.4844.74, so I believe the impact should be Stable instead of Head.
2. I believe the severity should be High instead of Medium. The issue is a type confusion that allows a simple api call to be called on a very much arbitrary object. (In fact, the fields x2 and x3 together specify the address of which the api call will be used as a receiver) The test case causes a dereference of an invalid receiver address, which manifests as an unknown read in cluster fuzz, causing it to label it as "Medium", but I believe it should be rated "High" according to the guidelines. (Wide range of function call on a more or less arbitrary object)
3. I've tested the patch and it does prevent the bug here, however, there is also a JIT version of the bug which can bypass the patch. I've opened another ticket for tracking, (1309467) in case it is someone else's responsibility for the JIT code, but both should probably be fixed before release.

Comment 12 by marja@chromium.org on Wed, Mar 23, 2022, 10:09 AM EDT

Yep, this is on the Stable channel and we'd need to apply the patch to everything possible.

Can someone cc me in the JIT bug (https://bugs.chromium.org/p/chromium/issues/detail?id=1309467) , I can't access it.

Comment 13 by m...@semmle.com on Wed, Mar 23, 2022, 12:35 PM EDT

Sorry about that, let me copy the details of 1309467 here for you:

In `ReduceNamedAccess`, `map` is passed to `GetPropertyAccessInfo` to create `PropertyAccessInfo` [1]:

```
    PropertyAccessInfo access_info = broker()->GetPropertyAccessInfo(
        map, feedback.name(), access_mode, dependencies());
    access_infos_for_feedback.push_back(access_info);
  }
```

In the case of super property access, this map comes from the feedback collected for the `lookup_start_object` [2]:

```
if (!InferMaps(lookup_start_object, effect, &inferred_maps)) {
```

```
  if (!InferMaps(lookup_start_object, effect, &inferred_maps)) {
    for (const MapRef& map : feedback.maps()) {
      inferred_maps.push_back(map);
    }
  }
```

In the case of an accessor that is a simple api call, this map is then used in `AccessorAccessInfoHelper` to check for compatibility between the api call and the map [3]:

```
PropertyAccessInfo AccessorAccessInfoHelper(...) {
  ...

    CallOptimization::HolderLookup lookup;
    Handle<JSObject> holder_handle = broker->CanonicalPersistentHandle(
        optimization.LookupHolderOfExpectedType(
            broker->local_isolate_or_isolate(), receiver_map.object(),   //<------- receiver_map here is the map for `lookup_start_object` from `ReduceNamedAccess`
            &lookup));
```

`ReducedNameAccess` will then carry on to try inlining the getter call [4]:

```
base::Optional<JSNativeContextSpecialization::ValueEffectControl>
JSNativeContextSpecialization::BuildPropertyLoad(
    Node* lookup_start_object, Node* receiver, Node* context, Node* frame_state,
    Node* effect, Node* control, NameRef const& name,
    ZoneVector<Node*>* if_exceptions, PropertyAccessInfo const& access_info) {
  ...
  if (access_info.IsNotFound()) {
    ...
  } else if (access_info.IsFastAccessorConstant() ||
             access_info.IsDictionaryProtoAccessorConstant()) {
    ConvertReceiverMode receiver_mode =
        receiver == lookup_start_object
            ? ConvertReceiverMode::kNotNullOrUndefined
            : ConvertReceiverMode::kAny;
    value =
        InlinePropertyGetterCall(receiver, receiver_mode, context, frame_state,
                                 &effect, &control, if_exceptions, access_info);  //<------- receiver here is used for the inlined call
```

This will inline the simple api call with `receiver`, instead of `lookup_start_object`. By providing a `lookup_start_object` that passes the `LookupHolderOfExpectedType` test above and a `receiver` of a different type, a type confusion occurs.

The test case is `superic_jit.html`. If you test it after applying the patch, you'll see that it still crashes. Thanks.

superic_jit.html
378 bytes View Download

Comment 14 by sheriffbot on Wed, Mar 23, 2022, 12:52 PM EDT

Labels: M-100 Target-100

Setting milestone and target because of medium severity.

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

Comment 15 by marja@chromium.org on Thu, Mar 24, 2022, 4:19 AM EDT

I am having second thoughts about this fix. The API Callback surely needs the holder (the object where the property lives), but why does it need the lookup_start_object in addition to that? It shouldn't. It might theoretically do something with the receiver, although practically, in most cases it wouldn't.

Maybe the bug is on the Blink side, where it's using the passed receiver instead of the holder, when reading the property.

Comment 16 by marja@chromium.org on Thu, Mar 24, 2022, 4:28 AM EDT

This doesn't look right:

```
void LengthAttributeGetCallback(
    const v8::FunctionCallbackInfo<v8::Value>& info) {
  RUNTIME_CALL_TIMER_SCOPE_DISABLED_BY_DEFAULT(
      info.GetIsolate(), "Blink_OfflineAudioContext_length_Getter");
  BLINK_BINDINGS_TRACE_EVENT("OfflineAudioContext.length.get");

  v8::Local<v8::Object> v8_receiver = info.This(); <<< Noooooooo
  OfflineAudioContext* blink_receiver =
      V8OfflineAudioContext::ToWrappableUnsafe(v8_receiver); <<< Noooooooo!!!
  auto&& return_value = blink_receiver->length();
  bindings::V8SetReturnValue(
      info, return_value, bindings::V8ReturnValue::PrimitiveType<uint32_t>());
}
```

Comment 17 by marja@chromium.org on Thu, Mar 24, 2022, 4:32 AM EDT

And to make it more interesting:

```
/**
 * The argument information given to function call callbacks.  This
 * class provides access to information about the context of the call,
 * including the receiver, the number and values of arguments, and
 * the holder of the function.
 */
template <typename T>
class FunctionCallbackInfo {
 public:
  ...
  /** Returns the receiver. This corresponds to the "this" value. */
  V8_INLINE Local<Object> This() const;
  /**
   * If the callback was created without a Signature, this is the same
   * value as This(). If there is a signature, and the signature didn't match
   * This() but one of its hidden prototypes, this will be the respective
   * hidden prototype.
   *

   * Note that this is not the prototype of This() on which the accessor
   * referencing this callback was found (which in V8 internally is often
   * referred to as holder [sic]).
```

```
   " referred to as holder [sic]).
   */
  V8_INLINE Local<Object> Holder() const;
  ...
};
```

-> So FunctionCallbackInfo::Holder is *not* the same as the V8 holder, which we would need here.

Comment 18 by m...@semmle.com on Thu, Mar 24, 2022, 6:13 AM EDT

I believe the various objects that are used in the context of a super property accessor have the following semantics:

When accessing a super property, the the search starts from the prototype of the current object. The object where the search starts is the `lookup_start_object` and the current object is the `receiver`. So for example, the following will print out `undefined` because `x` is not a property found in the prototype chain of `lookup_start_object` (`{}`), even though it is found in the `receiver` (`b`):

```
class B {
  constructor() {
    this.x = 1;
  }
  m() {
    return super.x;
  }

}

B.prototype.__proto__ = {};

b = new B();
console.log(b.m());
```

The search for the property will be done in the prototype chain of the `lookup_start_object`. If the property does not exist in the `lookup_start_object`, but in a prototype of it, say `X`, then `X` will be the `holder` of the property. So it can happen that `holder` is neither the `lookup_start_object` nor the `receiver`.

In the case of an accessor. The `holder` is the object that defines the `getter` and `setter` pairs. However, when returning the property, these `getter` and `setter` are called with a receiver. This receiver would be the actual object that is making the `super` property call, instead of the `lookup_start_object` or the `holder`. This is the behaviour of the runtime code. (See also, https://tc39.es/ecma262/multipage/ecmascript-language-expressions.html#sec-makesuperpropertyreference in which `actualThis` is passed)

For example, the following will output `B` instead of `A`:

```
class B {
  constructor() {
    this.x = 'B';
  }
  m() {
    return super.prop;
  }
```

```
}

var a = {x: 'A', get prop() {return this.x;}};


B.prototype.__proto__ = a;

b = new B();
console.log(b.m());
```

As the patch replaces `receiver` by `lookup_start_object` when calling the accessor, it actually causes inconsistency between the runtime code behaviour and the IC behaviour. For example, when running the attached `superic2.html` on a patched version of the code, the console output will be 5000 and 38400 respectively. The former, `5000` is the runtime behaviour, which fetches `length` from the `receiver` whereas the IC fetches `length` from the `lookup_start_object`, which is 38400, which is not consistent.

The root cause of this, it seems to me, is that the map of `lookup_start_object` is used in the type check in `LookupHolderOfExpectedType` when the IC handler is created, when in fact the `receiver_map` should be used because it is the object where the accessor will actually be used as an argument. So I'd suggest checking the `receiver_map` in `LookupHolderOfExpectedType` instead of `lookup_start_object` to fix this.

superic2.html
418 bytes View Download

Comment 19 by marja@chromium.org on Thu, Mar 24, 2022, 8:29 AM EDT

I was confused by this:

```
class B {
  m() {
    return super.length;
  }
}
B.prototype.__proto__ = new OfflineAudioContext(1, 38400, 38400);
let b = new B();
console.log(b.m()); // TypeError: Illegal invocation
```

---> The expected (though surprising) behavior is that the API getter is *not* called.

It doesn't match the "pure JS" behavior; if it was a JS class with a getter, the getter would be called:

```
class B {
  m() {
    return super.length;
  }
}
class PureJSClass {
  get length() {
      console.log(this.x); // 'this property marks the receiver'
      return 38;

  }
}
B.prototype.__proto__ = new PureJSClass();
```

```
B.prototype.__proto__ = new PureJSClass();
let b = new B();
b.x = 'this property marks the receiver';
console.log(b.m()); // 38

class B {
  m() {
    return super.length;
  }
}
class PureJSClass {
  get length() { // receiver is 'b' when this is called
      return 38;
  }
}
B.prototype.__proto__ = new PureJSClass();
let b = new B();
console.log(b.m()); // 38
```

--> So, here we need to replicate the "Illegal invocation" behavior. Comment 18 seems legit. I'll need to dig up the exact spec though to make sure we get this right... it's not in the ecma262 spec since it's about API getters.

Comment 20 by m...@semmle.com on Thu, Mar 24, 2022, 9:06 AM EDT

The api getter do get called in the first case, I think what you're seeing is this check here that failed when the api call is invoked:

https://source.chromium.org/chromium/chromium/src/+/334c8a70297520a868e1a4aa4f48bb157e1ce6d3:v8/src/builtins/builtins-api.cc;l=98

I'm not totally sure what should the correct behaviour be for pure js. In the end of the day, you'll ended up calling the getter on a JS object, but unless the accessor, which is a JS function, is expecting the receiver to be a certain type, which shouldn't be the case for JS function (pure JS functions should be able to handle objects of different types without causing security issue, and if it is expected a certain type for correctness, then it should check it in the function) Things may be different if the getter then gets optimized and is assuming a receiver to be of certain type, but I'd consider that a problem in the optimized code and should be taken care of by proper map checks etc. in the optimized code.

I can see that there is some inconsistencies here as in pure js doesn't throw but blink api throws, although I'd consider it as a case of blink objects putting extra checks in the accessors, as if it is defined in JS as follows:

```
class OfflineAudioContext {

  get length() {
    if (!(this instanceof OfflineAudioContext)) {
      throw new TypeError("Illegal Invocation\n");
    }
  }
  ...
}
```

So I don't see this as something that contradicts the behaviour of pure js, but rather some implementation details of WebAPI.

Comment 21 by marja@chromium.org on Thu, Mar 24, 2022, 9:11 AM EDT

Re: "I'm not totally sure what should the correct behaviour be for pure js."

What's the unclear thing there? That's all specced in https://tc39.es/ecma262 , so easy to figure out ("easy" in comparison to the web stuff which is spread across multiple specs).

Clarification, with "--> So, here we need to replicate the "Illegal invocation" behavior." I meant, in this bug we need to make sure to follow the behavior that produces that TypeError. Pure JS semantics are working fine, afaict, so anything related to super property access in pure JS should not change.

I found https://webidl.spec.whatwg.org/#dfn-attribute-getter which might be an entry point to the behavior we want for the API object case.

Comment 22 by Git Watcher on Thu, Mar 24, 2022, 9:24 AM EDT

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/96c5daaea685c72abbc20b8083f6e40c87cabccd

commit 96c5daaea685c72abbc20b8083f6e40c87cabccd
Author: Marja Hölttä <marja@chromium.org>
Date: Thu Mar 24 12:31:33 2022

Revert "[super IC] Fix receiver vs lookup start object confusion"

This reverts commit 9c3d4b3556b2797fa9d9f4bee915e8502608312f.

Reason for revert: This is not the right fix (see bug).

Original change's description:
> [super IC] Fix receiver vs lookup start object confusion
>
> ~~Bug: v8:9237~~,~~chromium:1308360~~
> Change-Id: I11e3c14a6cecb9d88a834711fb6252191494d5f7
> Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3545172
> Reviewed-by: Igor Sheludko <ishell@chromium.org>
> Commit-Queue: Marja Hölttä <marja@chromium.org>
> Cr-Commit-Position: refs/heads/main@{#79571}

~~Bug: v8:9237~~,~~chromium:1308360~~
Change-Id: I0efa6ab561482ffc323b63500acfeb80598f3e7c
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3548896
Auto-Submit: Marja Hölttä <marja@chromium.org>
Bot-Commit: Rubber Stamper <rubber-stamper@appspot.gserviceaccount.com>
Commit-Queue: Marja Hölttä <marja@chromium.org>
Reviewed-by: Igor Sheludko <ishell@chromium.org>
Cr-Commit-Position: refs/heads/main@{#79604}

[modify] https://crrev.com/96c5daaea685c72abbc20b8083f6e40c87cabccd/src/ic/accessor-assembler.cc

Comment 23 by m...@semmle.com on Thu, Mar 24, 2022, 9:34 AM EDT

Thanks for clarifying, I wasn't sure if you meant pure js should also throw when the type is incorrect, but it is now clear to me that the pure js behaviour is correct. Thanks!

Comment 24 by marja@chromium.org on Thu, Mar 24, 2022, 12:00 PM EDT

Ok so, my latest theory:

When deciding whether an API getter can be called, we need to do checks on the receiver. However, the super IC system is not prepared for that - it assumes it can reuse the IC system with a different lookup-start-object.

When deciding when a handler from the megamorphic stub cache applies, it's not enough to just check stuff on the lookup start object. Thus, we end up using the wrong handler.

Comment 25 by marja@chromium.org on Thu, Mar 24, 2022, 1:02 PM EDT

Also, mmo@, thanks for reporting this bug, this is an awesome bug report! :)

For now, I'm landing a big-hammer solution of turning off super IC altogether (should appear here soon).

I'm OOO on Friday. If folks need to merge the kill switch CL, please do so. I'll work on a proper analysis & fix next week.

Comment 26 by Git Watcher on Thu, Mar 24, 2022, 2:00 PM EDT

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/c6b68cbfbd49a24bd9d343d718132125370da729

commit c6b68cbfbd49a24bd9d343d718132125370da729
Author: Marja Hölttä <marja@chromium.org>
Date: Thu Mar 24 16:30:16 2022

[super IC] Turn off super ICs

They make assumptions which don't hold for API handlers.

~~Bug: v8:9237~~,~~chromium:1308360~~
Change-Id: I9f122c4e75a24d83ef3653cbf7a223ed522e4d13
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3548899
Reviewed-by: Igor Sheludko <ishell@chromium.org>
Commit-Queue: Marja Hölttä <marja@chromium.org>
Cr-Commit-Position: refs/heads/main@{#79614}

[modify]
https://crrev.com/c6b68cbfbd49a24bd9d343d718132125370da729/test/cctest/interpreter/bytecode_expectations/ClassAndSuperClass.golden
[modify] https://crrev.com/c6b68cbfbd49a24bd9d343d718132125370da729/src/flags/flag-definitions.h

Comment 27 by adetaylor@google.com on Thu, Mar 24, 2022, 2:30 PM EDT

Cc: amyressler@google.com
Labels: Merge-Request-100

amyressler@ I was discussing another fix with ishell@ and they suggest we might want #c26 in the initial M100 release, even though we're kind of out of time. Adding a merge request for your consideration.

Comment 28 by gov...@chromium.org on Thu, Mar 24, 2022, 5:43 PM EDT

Labels: Merge-Request-101

This also need a merge to M101

Comment 29 by adetaylor@google.com on Thu, Mar 24, 2022, 6:00 PM EDT

Labels: -Security_Severity-Medium Security_Severity-High

Re-assessing as High based on #c11.

Comment 30 by adetaylor@google.com on Thu, Mar 24, 2022, 6:07 PM EDT

marja@ please would you create cherry-pick CLs to M100 and M101, and get them +1'd and, if applicable, CQ dry-runned. We're likely to want to merge to M100 first thing tomorrow (Friday) US time, assuming no problems show up with the cherry-pick into canary (https://chromium-review.googlesource.com/c/v8/v8/+/3550658)

Comment 31 by ClusterFuzz on Thu, Mar 24, 2022, 6:10 PM EDT

Labels: -Security_Impact-Extended Security_Impact-Head

Detailed Report: https://clusterfuzz.com/testcase?key=5712077410598912

Fuzzer: None
Job Type: linux_asan_chrome_mp
Platform Id: linux

Crash Type: UNKNOWN READ
Crash Address: 0x00017fff802e
Crash State:
  blink::v8_offline_audio_context::LengthAttributeGetCallback
  v8::internal::Invoke
  v8::internal::Execution::CallScript

Sanitizer: address (ASAN)

Recommended Security Severity: Medium

Regressed: https://clusterfuzz.com/revisions?job=linux_asan_chrome_mp&range=866941:866942

Reproducer Testcase: https://clusterfuzz.com/download?testcase_id=5712077410598912

To reproduce this, please build the target in this report and run it against the reproducer testcase. Please use the GN arguments provided at bottom of this report when building the binary.

If you have trouble reproducing, please also export the environment variables listed under "[Environment]" in the crash stacktrace.

If you have any feedback on reproducing test cases, let us know at https://forms.gle/Yh3qCYFveHj6E5jz5 so we can improve.

The recommended severity (Security_Severity-Medium) is different from what was assigned to the bug. Please double check the accuracy of the assigned severity.

Comment 32 by sheriffbot on Thu, Mar 24, 2022, 6:10 PM EDT

Labels: -Security_Impact-Head Security_Impact-Extended

Comment 33 by Git Watcher on Thu, Mar 24, 2022, 6:11 PM EDT

Labels: merge-merged-4962

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/464987b1183c54d8c7039ffe8a8c464c69e43912

commit 464987b1183c54d8c7039ffe8a8c464c69e43912
Author: Marja Hölttä <marja@chromium.org>
Date: Thu Mar 24 16:30:16 2022

[super IC] Turn off super ICs

They make assumptions which don't hold for API handlers.

~~Bug: v8:9237~~,~~chromium:1308360~~
Change-Id: I9f122c4e75a24d83ef3653cbf7a223ed522e4d13
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3548899
Reviewed-by: Igor Sheludko <ishell@chromium.org>
Commit-Queue: Marja Hölttä <marja@chromium.org>
Cr-Commit-Position: refs/heads/main@{#79614}
(cherry picked from commit c6b68cbfbd49a24bd9d343d7181321253 70da729)
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3550658
Reviewed-by: Shu-yu Guo <syg@chromium.org>
Commit-Queue: Shu-yu Guo <syg@chromium.org>

[modify]
https://crrev.com/464987b1183c54d8c7039ffe8a8c464c69e43912/test/cctest/interpreter/bytecode_expectations/ClassAndSuperClass.golden
[modify] https://crrev.com/464987b1183c54d8c7039ffe8a8c464c69e43912/src/flags/flag-definitions.h

Comment 34 Deleted

Comment 35 by gov...@chromium.org on Thu, Mar 24, 2022, 6:17 PM EDT

Android and Desktop Canary #102.0.4962.3 (currently building) include this fix. Requesting to verify the change and check stability.

Comment 36 by ClusterFuzz on Thu, Mar 24, 2022, 6:28 PM EDT

Status: Verified (was: Started)
Labels: ClusterFuzz-Verified

ClusterFuzz testcase 5712077410598912 is verified as fixed in https://clusterfuzz.com/revisions?job=linux_asan_chrome_mp&range=984656:984660

If this is incorrect, please add the ClusterFuzz-Wrong label and re-open the issue.

Comment 37 by sheriffbot on Thu, Mar 24, 2022, 6:30 PM EDT

Labels: LTS-Merge-Candidate

LTS Milestone M96

This issue has been flagged as a merge candidate for Chrome OS' LTS channel. If selected, our merge team will handle any additional merges. To help us determine if this issue requires a merge to LTS, please answer this short questionnaire:
1. Was this issue a regression for the milestone it was found in?

2. Is this issue related to a change or feature merged after the latest LTS Milestone?

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

Comment 38 by sheriffbot on Fri, Mar 25, 2022, 4:35 AM EDT

Labels: V8-postmortem

This high+ V8 security issue with stable impact requires a lightweight post mortem. Please take some time to answer questions asked in this form [1] to help us improve V8 security. [1] https://docs.google.com/forms/d/e/1FAIpQLSdSMCiEpIFLLFkMbgtulK1sf1B-idQmkFaA4XP2Rz5mN1cqWg/viewform?usp=pp_url&entry.307501673=1308360&entry.364066060=External&entry.958145677=Android&entry.958145677=Chrome&entry.958145677=Fuchsia&entry.958145677=Linux&entry.958145677=Mac&entry.958145677=Windows&entry.958145677=Lacros&entry.763880440=Extended&entry.1678852700=High&entry.763402679=Blink>JavaScript>Runtime&entry.975983575=marja@chromium.org Please ensure to copy the full link, as otherwise some issue meta data might not be populated automatically.

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

Comment 39 by amyressler@chromium.org on Fri, Mar 25, 2022, 10:56 AM EDT

Cc: -amyressler@google.com amyressler@chromium.org
Labels: -Merge-Request-100 -Merge-Request-101 Merge-Approved-100 Merge-Approved-101

While it's limited data given the timing, I'm not seeing any issues on 102.0.4962.3 so far, so approving for merge M100 (4896) and M101 (4951)

Comment 40 by rzanoni@google.com on Fri, Mar 25, 2022, 11:15 AM EDT

Labels: LTS-Merge-Request-96

Comment 41 by sheriffbot on Fri, Mar 25, 2022, 11:21 AM EDT

Labels: -LTS-Merge-Request-96 LTS-Merge-Review-96

This issue requires additional review before it can be merged to the LTS channel. Please answer the following questions to help us evaluate this merge:

1. Number of CLs needed for this fix and links to them.
2. Level of complexity (High, Medium, Low - Explain)
3. Has this been merged to a stable release? beta release?
4. Overall Recommendation (Yes, No)

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

Comment 42 by rzanoni@google.com on Fri, Mar 25, 2022, 11:26 AM EDT

1. Just https://crrev.com/c/3553109
2. Low, conflict on test expectations
3. Merge approved for 100 and 101
4. Yes

Comment 43 by Git Watcher on Fri, Mar 25, 2022, 12:14 PM EDT

Labels: merge-merged-10.0

The following revision refers to this bug:

https://chromium.googlesource.com/v8/v8/+/32fe54204859f8d04e55c62718ed776a35dc14aa

commit 32fe54204859f8d04e55c62718ed776a35dc14aa
Author: Igor Sheludko <ishell@chromium.org>
Date: Fri Mar 25 15:53:55 2022

Merged: [super IC] Turn off super ICs

They make assumptions which don't hold for API handlers.

Merge issues:
Conflicts in the bytecode test expectations

(cherry picked from commit c6b68cbfbd49a24bd9d343d718132125370da729)

~~Bug: v8:9237~~,~~chromium:1308360~~
Change-Id: Iffe4adddb6d0ab469b6bc78af1d40d73f29ffa38
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3550584
Reviewed-by: Toon Verwaest <verwaest@chromium.org>
Commit-Queue: Igor Sheludko <ishell@chromium.org>
Cr-Commit-Position: refs/branch-heads/10.0@{#14}
Cr-Branched-From: 6ea73a738c467dc26abbbe84e27a36aac1c6e119-refs/heads/10.0.139@{#1}
Cr-Branched-From: ccc689011280419901e6ee42cae39980c0e96030-refs/heads/main@{#79131}

[modify]
https://crrev.com/32fe54204859f8d04e55c62718ed776a35dc14aa/test/cctest/interpreter/bytecode_expectations/ClassAndSuperClass.golden
[modify] https://crrev.com/32fe54204859f8d04e55c62718ed776a35dc14aa/src/flags/flag-definitions.h

Comment 44 by Git Watcher on Fri, Mar 25, 2022, 12:16 PM EDT

Labels: merge-merged-10.1

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/6bdb9281a8d3cbe5771cdb0f42825050edd73384

commit 6bdb9281a8d3cbe5771cdb0f42825050edd73384
Author: Marja Hölttä <marja@chromium.org>
Date: Thu Mar 24 16:30:16 2022

Merged: [super IC] Turn off super ICs

They make assumptions which don't hold for API handlers.

~~Bug: v8:9237~~,~~chromium:1308360~~
(cherry picked from commit c6b68cbfbd49a24bd9d343d718132125370da729)

Change-Id: Ibcf956907d437b8d12b025596e5c7717af54824f
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3553110
Reviewed-by: Toon Verwaest <verwaest@chromium.org>
Commit-Queue: Igor Sheludko <ishell@chromium.org>
Cr-Commit-Position: refs/branch-heads/10.1@{#8}
Cr-Branched-From: b003970395b7efcc309eb30b4ca06dd8385acd55-refs/heads/10.1.124@{#1}
Cr-Branched-From: e62f556862624103ea1da5b9dcef9b216832033b-refs/heads/main@{#79503}

[modify] https://crrev.com/6bdb9281a8d3cbe5771cdb0f42825050edd73384/test/cctest/interpreter/bytecode_expectations/ClassAndSuperClass.golden
[modify] https://crrev.com/6bdb9281a8d3cbe5771cdb0f42825050edd73384/src/flags/flag-definitions.h

Comment 45 by srinivassista@google.com on Fri, Mar 25, 2022, 12:51 PM EDT

Labels: -Merge-Approved-100 -Merge-Approved-101 Merge-Merged

Dropping merge approved labels as the merges to M100 and M101 are completed in comments 43/44

Comment 46 by amyressler@chromium.org on Fri, Mar 25, 2022, 2:09 PM EDT

Cc: syg@chromium.org

Comment 47 by gov...@chromium.org on Fri, Mar 25, 2022, 3:30 PM EDT

Cc: amineer@chromium.org

Comment 48 by gov...@chromium.org on Fri, Mar 25, 2022, 3:34 PM EDT

Cc: amin...@google.com

Comment 49 by sheriffbot on Sun, Mar 27, 2022, 1:40 PM EDT

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

Comment 50 by marja@chromium.org on Mon, Mar 28, 2022, 3:46 AM EDT

Cc: mslekova@chromium.org

Comment 51 by m...@semmle.com on Mon, Mar 28, 2022, 10:41 AM EDT

I had a look at it a bit more and it seems that the `kNativeDataProperty` case have the same problem. This case is under the accessor sub case. If the accessor is not a simple api, then the IC code will try to see if it is a native data property. In this case, it also checks the `lookup_start_object_map` for signature compatibility [1]

```
    if (v8::ToCData<Address>(info->getter()) == kNullAddress ||
        !AccessorInfo::IsCompatibleReceiverMap(info, map) ||
        !holder->HasFastProperties() ||
        (info->is_sloppy() && !receiver->IsJSReceiver())) {
      ...
      return LoadHandler::LoadSlow(isolate());     //<--------- bailout if signature not compatible
    }
```

On the other hand, `receiver` is used when the accessor is called [2]

```
void AccessorAssembler::HandleLoadCallbackProperty(
    const LazyLoadICParameters* p, TNode<JSObject> holder,
    TNode<WordT> handler_word, ExitPoint* exit_point) {
  Comment("native_data_property_load");

  ...
  exit_point->ReturnCallStub(callable, p->context(), p->receiver(), holder,
                             accessor_info);
```

```
                    accessor_info);
}

```

which is the same problem with the simple api case. A quick look at how native data is used (looking at the call graph of `AddNativeDataProperty` [3]) seems to show that in all the use cases, the signature doesn't matter (i.e. empty signatures are passed so the accessors can use any receiver), however, it is probably worth fixing this case also in case the situation changes in the future. The JIT code does not seem affected as it only handles simple api calls [4].

1. https://source.chromium.org/chromium/chromium/src/+/e5eeff20f769c93f074084316a9c51d4ae34fb07:v8/src/ic/ic.cc;l=1112
2. https://source.chromium.org/chromium/chromium/src/+/e5eeff20f769c93f074084316a9c51d4ae34fb07:v8/src/ic/accessor-assembler.cc;l=246
3. https://source.chromium.org/chromium/chromium/src/+/e5eeff20f769c93f074084316a9c51d4ae34fb07:v8/src/api/api-natives.cc;l=662;bpv=0;bpt=1
4. https://source.chromium.org/chromium/chromium/src/+/75c36c7712bea160e69de6b87b864dfcebab239e:v8/src/compiler/access-info.cc;l=565

Comment 52 by adetaylor@chromium.org on Mon, Mar 28, 2022, 10:55 AM EDT

I'm going to report this as a brand new bug to avoid confusing all our merge and release labels... I'll add the number back here after I've done so.

Comment 53 by adetaylor@chromium.org on Mon, Mar 28, 2022, 10:57 AM EDT

Reported as ~~issue 1310790~~.

Comment 54 by amin...@google.com on Mon, Mar 28, 2022, 1:31 PM EDT

Cc: -amineer@chromium.org -amin...@google.com

Comment 55 by amyressler@chromium.org on Mon, Mar 28, 2022, 5:29 PM EDT

Labels: Release-0-M100

Comment 56 by amyressler@google.com on Tue, Mar 29, 2022, 1:14 PM EDT

Labels: CVE-2022-1134 CVE_description-missing

Comment 57 by gmpritchard@google.com on Tue, Mar 29, 2022, 4:47 PM EDT

Labels: -LTS-Merge-Candidate LTS-Merge-Delayed-96

Comment 58 by Git Watcher on Wed, Mar 30, 2022, 9:10 AM EDT

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/185d6116ae7d7771c590f38be8b08530f979d226

commit 185d6116ae7d7771c590f38be8b08530f979d226
Author: Marja Hölttä <marja@chromium.org>

Date: Wed Mar 30 09:40:43 2022

[super IC] Fix API getter related bugs and re-enable super IC

[super IC] Fix API getter related bugs and re-enable super IC

~~Bug: chromium:1308360~~,~~chromium:1309467~~,~~v8:9237~~
Change-Id: I2923e3ee60b4b30c4e2b57b9c8569a030fc7bfbd
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3550588
Reviewed-by: Tobias Tebbi <tebbi@chromium.org>
Reviewed-by: Toon Verwaest <verwaest@chromium.org>
Reviewed-by: Maya Lekova <mslekova@chromium.org>
Commit-Queue: Marja Hölttä <marja@chromium.org>
Cr-Commit-Position: refs/heads/main@{#79676}

[modify]
https://crrev.com/185d6116ae7d7771c590f38be8b08530f979d226/test/cctest/interpreter/bytecode_expectations/ClassAndSuperClass.golden
[modify] https://crrev.com/185d6116ae7d7771c590f38be8b08530f979d226/src/flags/flag-definitions.h
[modify] https://crrev.com/185d6116ae7d7771c590f38be8b08530f979d226/src/compiler/js-native-context-specialization.cc
[modify] https://crrev.com/185d6116ae7d7771c590f38be8b08530f979d226/src/compiler/js-native-context-specialization.h
[modify] https://crrev.com/185d6116ae7d7771c590f38be8b08530f979d226/src/ic/accessor-assembler.cc

Comment 59 by gmpritchard@google.com on Fri, Apr 8, 2022, 10:29 AM EDT

Labels: -LTS-Merge-Review-96 -LTS-Merge-Delayed-96 LTS-Merge-Approved-96

Comment 60 by Git Watcher on Mon, Apr 11, 2022, 9:07 AM EDT

Labels: merge-merged-9.6

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/f546ac11eec75a5f3db797a844a5b8e2322f345f

commit f546ac11eec75a5f3db797a844a5b8e2322f345f
Author: Marja Hölttä <marja@chromium.org>
Date: Thu Mar 24 16:30:16 2022

[M96-LTS][super IC] Turn off super ICs

M96 merge issues:
Conflicts in the bytecode test expectations

They make assumptions which don't hold for API handlers.

(cherry picked from commit c6b68cbfbd49a24bd9d343d7181321253 70da729)

~~Bug: v8:9237~~,~~chromium:1308360~~
No-Try: true
No-Presubmit: true
No-Tree-Checks: true
Change-Id: I9f122c4e75a24d83ef3653cbf7a223ed522e4d13
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3548899
Commit-Queue: Marja Hölttä <marja@chromium.org>
Cr-Original-Commit-Position: refs/heads/main@{#79614}
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3553109
Reviewed-by: Igor Sheludko <ishell@chromium.org>
Commit-Queue: Roger Felipe Zanoni da Silva <rzanoni@google.com>
Cr-Commit-Position: refs/branch-heads/9.6@{#60}
Cr-Branched-From: 0b7bda016178bf438f90b2c02da572ac2663a1f7-refs/heads/9.6.180@{#1}

Cr-Branched-From: 0b7bda01b178bf4381b9b3c93da572ae3663a1f7-refs/heads/9.6.180@{#1}
Cr-Branched-From: 41a5a247d9430b953e38631e88d17790306f7a4c-refs/heads/main@{#77244}

[modify] https://crrev.com/f546ac11eec75a5f3db797a844a5b8e2322f345f/test/cctest/interpreter/bytecode_expectations/ClassAndSuperClass.golden
[modify] https://crrev.com/f546ac11eec75a5f3db797a844a5b8e2322f345f/src/flags/flag-definitions.h

Comment 61 by rzanoni@google.com on Fri, Apr 15, 2022, 11:48 AM EDT

Labels: -LTS-Merge-Approved-96 LTS-Merge-Merged-96

Comment 62 by sheriffbot on Fri, Jul 1, 2022, 1:31 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 63 by Git Watcher on Tue, Jul 5, 2022, 10:18 AM EDT

The following revision refers to this bug:
https://chromium.googlesource.com/v8/v8/+/f3f47a9fef0e5f8c49bb7292327a1de84f33f64c

commit f3f47a9fef0e5f8c49bb7292327a1de84f33f64c
Author: Marja Hölttä <marja@chromium.org>
Date: Tue Jul 05 12:28:30 2022

[super IC] Add tests for a security bug

~~Bug: chromium:1309467~~,~~chromium:1308360~~,~~v8:9237~~
Change-Id: I77b004e263a9bed98a0dfe5936bdad055bde36a6
Reviewed-on: https://chromium-review.googlesource.com/c/v8/v8/+/3745365
Reviewed-by: Igor Sheludko <ishell@chromium.org>
Commit-Queue: Marja Hölttä <marja@chromium.org>
Cr-Commit-Position: refs/heads/main@{#81530}

[add] https://crrev.com/f3f47a9fef0e5f8c49bb7292327a1de84f33f64c/test/mjsunit/regress/regress-crbug-1309467.js
[add] https://crrev.com/f3f47a9fef0e5f8c49bb7292327a1de84f33f64c/test/mjsunit/regress/regress-crbug-1308360.js

Comment 64 by amyressler@google.com on Fri, Jul 22, 2022, 7:36 PM EDT

Labels: CVE_description-submitted -CVE_description-missing

Comment 65 by amyressler@chromium.org on Fri, Jul 29, 2022, 5:26 PM EDT

Labels: -CVE_description-missing --CVE_description-missing