Talos Vulnerability Report

TALOS-2021-1398

Google Chrome MediaStreamTrackGenerator use after free vulnerability

JANUARY 27, 2022

CVE NUMBER

CVE-2021-38008

Summary

A potential code execution vulnerability exists in the MediaStreamTrackGenerator functionality of Google Chrome 94.0.4606.81 (Stable) and 97.0.4674.1 (Canary). A specially-crafted web page can lead to use-after-free. An attacker can provide a malicious web site to trigger this vulnerability.

Tested Versions

Google Chrome 94.0.4606.81 (Stable)
Google Chrome 97.0.4674.1 (Canary)

Product URLs

Chrome - https://www.google.com/chrome/

CVSSv3 Score

8.3 - CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:L

CWE

CWE-416 - Use After Free

Details

Google Chrome is a cross-platform web browser, developed by Google.

The vulnerability exists in object `MediaStreamTrackGenerator` which is responsible for creating streams of audio/video.

`MediaStreamTrackGenerator` inherits from `MediaStreamTrack`. Thanks to this inheritance the `MediaStreamTrackGenerator` object can be cloned/copied to new object. See function below:

```
 1:      MediaStreamTrack* MediaStreamTrack::cloneMediaStreamTrack::clone(ScriptState* script_state) {
 2:      SendLogMessage(String::Format("%s()", __func__));
 3:      MediaStreamComponent* cloned_component = Component()->Clone();
 4:      MediaStreamTrack* cloned_track = MakeGarbageCollected<MediaStreamTrack>(
 5:          ExecutionContext::From(script_state), cloned_component, ready_state_,
 6:          base::DoNothing());
 7:      DidCloneMediaStreamTrack(Component(), cloned_component);
 8:      if (image_capture_) {
 9:          cloned_track->image_capture_ = image_capture_->Clone();
10:      }
11:      return cloned_track;
12:      }
```

The above function is resposible for making the copy of the `MediaStreamTrackGenerator` object. The allocation of new memory happens at line 4. By definition when using `MakeGarbageCollected`, the new object wont be deleted by `delete` but using Olipan, which is the Chromium garbage collector.

```
You can create an instance of your class through MakeGarbageCollected<T>, while you may not free the object with delete,
as Oilpan is responsible for deallocating the object once it determines the object is unreachable.
```

During streaming we can abort the stream using the JS function `abort`, which corresponds to the C++ code shown below.

```
1:      ScriptPromise MediaStreamAudioTrackUnderlyingSink::abort(
2:          ScriptState* script_state,
3:          ScriptValue reason,
4:          ExceptionState& exception_state) {
5:      DCHECK_CALLED_ON_VALID_SEQUENCE(sequence_checker_);
6:      Disconnect();
7:      return ScriptPromise::CastUndefined(script_state);
8:      }
```

The interesting part happens at line 6, where the stream gets disconnected, which triggers destructor of the `MediaStreamTrackGenerator` object. However we still have a copy of this object, which was marked by Olipan. This process can be interrupted using Events. One of those events is `onended`, which is triggered when the status of the stream is changed. During this event we can try to free once again the cloned `MediaStreamTrackGenerator` which would lead to use-after-free vulnerability.

With proper manipulation of objects, when the `onended` event is executed, this vulnerability could lead to control over freed memory and ultimately arbitrary code execution.

Crash Information

```
=================================================================
==18564==ERROR: AddressSanitizer: use-after-poison on address 0x7e80003dc378 at pc 0x7ff63031244d bp 0x00a2070fccd0 sp 0x00a2070fcd18
READ of size 8 at 0x7e80003dc378 thread T0
    #0 0x7ff63031244c in v8::internal::ObjectVisitor::VisitCustomWeakPointers(class v8::internal::HeapObject, class
v8::internal::CompressedObjectSlot, class v8::internal::CompressedObjectSlot) (D:\src\out\95.0.4638.49\content_shell.exe+0x14066244c)
    #1 0x7ff639d1f9a7 in base::OnceCallback<void ()>::Run D:\src\base\callback.h:100
    #2 0x7ff639d1f9a7 in blink::MediaStreamSource::SetReadyState(enum blink::MediaStreamSource::ReadyState)
D:\src\third_party\blink\renderer\platform\mediastream\media_stream_source.cc:181:36
    #3 0x7ff639c88ceb in blink::WebPlatformMediaStreamSource::FinalizeStopSource(void)
D:\src\third_party\blink\renderer\platform\exported\mediastream\web_platform_media_stream_source.cc:37:13
    #4 0x7ff64548150c in blink::PushableMediaStreamAudioSource::Broker::StopSourceOnMain
D:\src\third_party\blink\renderer\modules\breakout_box\pushable_media_stream_audio_source.cc:93
    #5 0x7ff64548150c in blink::PushableMediaStreamAudioSource::Broker::StopSource(void)
D:\src\third_party\blink\renderer\modules\breakout_box\pushable_media_stream_audio_source.cc:53:5
    #6 0x7ff6454846dd in blink::MediaStreamAudioTrackUnderlyingSink::Disconnect
D:\src\third_party\blink\renderer\modules\breakout_box\media_stream_audio_track_underlying_sink.cc:116
    #7 0x7ff6454846dd in blink::MediaStreamAudioTrackUnderlyingSink::abort(class blink::ScriptState *, class blink::ScriptValue, class
blink::ExceptionState &) D:\src\third_party\blink\renderer\modules\breakout_box\media_stream_audio_track_underlying_sink.cc:92:3
    #8 0x7ff639e3ba9f in blink::`anonymous namespace'::v8_underlying_sink_base::AbortOperationCallback
D:\src\out\95.0.4638.49\gen\third_party\blink\renderer\bindings\core\v8\v8_underlying_sink_base.cc:103:23
    #9 0x7ff632f2c9d7 in v8::internal::FunctionCallbackArguments::Call(class v8::internal::CallHandlerInfo) D:\src\v8\src\api\api-arguments-
inl.h:152:3
    #10 0x7ff632f29da4 in v8::internal::`anonymous namespace'::HandleApiCallHelper<0> D:\src\v8\src\builtins\builtins-api.cc:112:36
    #11 0x7ff632f27e42 in v8::internal::Builtins::InvokeApiFunction(class v8::internal::Isolate *, bool, class v8::internal::Handle<class
v8::internal::HeapObject>, class v8::internal::Handle<class v8::internal::Object>, int, class v8::internal::Handle<class
v8::internal::Object> *const, class v8::internal::Handle<class v8::internal::HeapObject>) D:\src\v8\src\builtins\builtins-api.cc:226:16
    #12 0x7ff63327090b in v8::internal::`anonymous namespace'::Invoke D:\src\v8\src\execution\execution.cc:283:20
    #13 0x7ff63326efd1 in v8::internal::Execution::Call(class v8::internal::Isolate *, class v8::internal::Handle<class
v8::internal::Object>, class v8::internal::Handle<class v8::internal::Object>, int, class v8::internal::Handle<class v8::internal::Object>
*const) D:\src\v8\src\execution\execution.cc:470:10
    #14 0x7ff632e245d8 in v8::Function::Call(class v8::Local<class v8::Context>, class v8::Local<class v8::Value>, int, class
v8::Local<class v8::Value> *const) D:\src\v8\src\api\api.cc:5179:7
    #15 0x7ff63d4af89e in blink::PromiseCall(class blink::ScriptState *, class v8::Local<class v8::Function>, class v8::Local<class
v8::Object>, int, class v8::Local<class v8::Value> *const) D:\src\third_party\blink\renderer\core\streams\miscellaneous_operations.cc:485:15
    #16 0x7ff63d4b1bee in blink::`anonymous namespace'::JavaScriptStreamAlgorithmWithoutExtraArg::Run
D:\src\third_party\blink\renderer\core\streams\miscellaneous_operations.cc:139:12
    #17 0x7ff639c059d8 in blink::WritableStreamDefaultController::AbortSteps(class blink::ScriptState *, class v8::Local<class v8::Value>)
D:\src\third_party\blink\renderer\core\streams\writable_stream_default_controller.cc:60:41
    #18 0x7ff639c01fd7 in blink::WritableStream::FinishErroring(class blink::ScriptState *, class blink::WritableStream *)
D:\src\third_party\blink\renderer\core\streams\writable_stream.cc:574:55
    #19 0x7ff639c06e71 in blink::WritableStreamDefaultController::AdvanceQueueIfNeeded(class blink::ScriptState *, class
blink::WritableStreamDefaultController *) D:\src\third_party\blink\renderer\core\streams\writable_stream_default_controller.cc:447:5
    #20 0x7ff639c09730 in blink::WritableStreamDefaultController::SetUp::ResolvePromiseFunction::CallWithLocal
D:\src\third_party\blink\renderer\core\streams\writable_stream_default_controller.cc:163:7
    #21 0x7ff632f2c9d7 in v8::internal::FunctionCallbackArguments::Call(class v8::internal::CallHandlerInfo) D:\src\v8\src\api\api-
arguments-inl.h:152:3
    #22 0x7ff632f29da4 in v8::internal::`anonymous namespace'::HandleApiCallHelper<0> D:\src\v8\src\builtins\builtins-api.cc:112:36
    #23 0x7ff632f271db in v8::internal::Builtin_Impl_HandleApiCall D:\src\v8\src\builtins\builtins-api.cc:142:5
    #24 0x7ff632f2652c in v8::internal::Builtin_HandleApiCall(int, unsigned __int64 *, class v8::internal::Isolate *)
D:\src\v8\src\builtins\builtins-api.cc:130:1
    #25 0x7ec9000c113b  (<unknown module>)

Address 0x7e80003dc378 is a wild pointer inside of access range of size 0x000000000008.
SUMMARY: AddressSanitizer: use-after-poison (D:\src\out\95.0.4638.49\content_shell.exe+0x14066244c) in
v8::internal::ObjectVisitor::VisitCustomWeakPointers(class v8::internal::HeapObject, class v8::internal::CompressedObjectSlot, class
v8::internal::CompressedObjectSlot)
Shadow bytes around the buggy address:
  0x1214ed1fb810: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x1214ed1fb820: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x1214ed1fb830: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x1214ed1fb840: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x1214ed1fb850: 00 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7
=>0x1214ed1fb860: f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7[f7]
  0x1214ed1fb870: f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7
  0x1214ed1fb880: f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7
  0x1214ed1fb890: f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7
  0x1214ed1fb8a0: f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7
  0x1214ed1fb8b0: f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7 f7
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
  Left alloca redzone:     ca
  Right alloca redzone:    cb
==18564==ABORTING
```

Timeline

2022-01-27 - Public Release

CREDIT

Discovered by Marcin Towalski of Cisco Talos.