Rewrite exclusion statistics: Jul 2021

[PUBLIC DOC: commiters@chromium.org can comment, anyone with a link can view]

Author: lukasza@chromium.org
Date: July 2021

This document provides a summary of the initial, automatically-generated raw_ptr<T> (aka CheckedPtr / aka BackupRefPtr) exclusions.

Summary

Assumptions:

- It is fair to ignore `const char*` pointers and `union`s. Rationale:
  - We plan to exclude them from anti-raw-ptr-field compile-time checks.
  - The public guidance will exclude `const char*` fields (along function pointers)
  - The public guidance will be to avoid using raw pointers in fields of structs and classes (and unions are technically neither a struct nor a class).

Summary of results:

- Most common long-term exclusions (after filtering out `const char*` and `union` exclusions):
  - 381 constexpr exclusions
  - 76 global-scope exclusions
  - Only 17 reinterpret_case-related exclusions remain after the fix
- Most common manually fixable exclusions:
  - 516 in-out-param / addr-of exclusions (168 in non-test code)
- Ignoring `const char*` avoids the need for 1439 more explicit exclusions

Repro steps

On July the 9th 2021, the following steps were taken:

1. `git fetch` to get the most recent, ~ToT Chromium sources
2. Apply the following changes to the rewriter:
   a. https://crrev.com/c/3017409: Only generate exclusions for `reinterpret_cast` of *trivial* types.
      i. Rationale: This change is needed to drastically reduce the scope of the “reinterpret-cast-struct” exclusion that has been reported to account for majority of exclusions in recent rewrites.
   b. https://crrev.com/c/3018575: Separate, explicit exclusion for fields in sources "test" in their path.

        i. Rationale: This change allows analysing 1) which of the old exclusions apply to test code (and checking if some of the exclusions appear mostly in test code) and 2) checking how many of the rewritten fields are in test code.
3. Build the rewriter
4. Invoke the rewriter
    a. One delta from the `tools/clang/rewrite_raw_ptr_fields/rewrite.sh` script is that the invocation also used --exclude-fields=.../manual-fields-to-ignore.txt cmdline argument.
        i. Rationale: This avoids double-counting exclusions.

The generated `automated-fields-to-ignore.txt` has been saved here.

Detailed results

Overall

```
# Exclusion numbers grouped by category
# (Excluding “test-code” because this is not a real exclusion - it has been artificially included
# - see repro step 2b)
# (Cases that should be fixable via a manual rewrite are colored green below)
$ cat automated-fields-to-ignore.txt | cut -d '#' -f 2 | sed -e 's/, test-code//' | sed -e 's/test-code//' |
grep -v const-char | grep -v union | grep '
[^ ]' | sort | uniq -c | sort -n
      1  addr-of, reinterpret-cast-trivial-type
      1  constexpr-ctor-field-initializer, in-out-param-ref
      1  embedder-has-no-operator-new, reinterpret-cast-trivial-type
      2  macro, pointee-has-no-operator-new
      3  constexpr-ctor-field-initializer, pointee-has-no-operator-new
      3  in-out-param-ref
      4  constexpr-ctor-field-initializer, embedder-has-no-operator-new
      4  reinterpret-cast-trivial-type
      5  constexpr-ctor-field-initializer, reinterpret-cast-trivial-type
      5  constexpr-var-initializer, global-scope
      5  overlapping
      8  addr-of, constexpr-ctor-field-initializer
      8  embedder-has-no-operator-new, pointee-has-no-operator-new
     10  addr-of, macro
     11  constexpr-ctor-field-initializer, macro
     30  macro
     34  constexpr-ctor-field-initializer, global-scope
     37  global-scope
     51  pointee-has-no-operator-new
```

```
 76  embedder-has-no-operator-new
179  addr-of
310  constexpr-ctor-field-initializer

# Overall number of exclusions (excluding “test-code” when it occurs *alone*,
# because this is not a real exclusion - it has been artificially included - see repro step 2b)
$ cat automated-fields-to-ignore.txt | grep -v '# test-code$' | grep -v const-char | grep -v union |
wc -l
788
```

reinterpret_case-related exclusions

```
$ cat automated-fields-to-ignore.txt | grep reinterpret | wc -l
17
```

global-scope

Example:

```
struct MyStruct {
  raw_ptr<int> ptr_field;
};

MyStruct g_struct;
```

Error (because raw_ptr<T> has a non-trivial / user-defined destructor):

```
.../test.cc:11915:10: error: declaration requires an exit-time
destructor [-Werror,-Wexit-time-destructors]
MyStruct g_struct;
         ^
```

Sometimes the “global-scope” problem can be avoided altogether, but the fix might not always be possible or might be icky (`base::NoDestructor`?). So, in general these exclusions are probably difficult to fix and will likely remain present forever.

```
# global scope:
$ cat automated-fields-to-ignore.txt | grep '# .*\bglobal-scope\b' | wc -l
938

# global scope, excluding const-char and unions (which will be ignored by the
anti-raw-ptr-compile-checks):
$ cat automated-fields-to-ignore.txt | grep '# .*\bglobal-scope\b' | grep -v '# .*\bconst-char\b' |
grep -v '# .*\bunion\b' |  wc -l
76
```

# global-scope in test-code:
$ cat automated-fields-to-ignore.txt | grep '# .*\bglobal-scope\b' | grep '# .*\btest\b' | wc -l
582

# global-scope in const-char:
$ cat automated-fields-to-ignore.txt | grep '# .*\bglobal-scope\b' | grep '# .*\bconst-char\b' | wc -l
856

# 10 random examples (**green** color indicates the example was understood and expected):
$ cat automated-fields-to-ignore.txt | grep '# .*\bglobal-scope\b' | grep -v 'const-char' | grep -v '# .*union' | cut -d '#' -f 1 | sort | uniq | sort -
-random-sort | head -10 | sort
autofill::(anonymous namespace)::CategoryOfWords::latin_dictionary **# static scope indeed**
base::FeatureParam<type-parameter-0-0, true>::options
base::Location::program_counter_
device::UsbVendor::products **# static member in Usblds::vendors_**
gpu::GpuControlList::Entry::cr_bugs **# gpu_control_list_..._autogen.h**
gpu::GpuControlList::MachineModelInfo::machine_model_names
net::TransportSecurityStateSource::Pinset::accepted_pins **# transport_security_state_static_unittest...h**
net::TransportSecurityStateSource::Pinset::rejected_pins
net::TransportSecurityStateSource::pinsets
tracing::MessageInfo::sub_messages

constexpr

Example:

```
struct MyStruct {
  raw_ptr<int> ptr_field;
};

int g_int = 123;
constexpr MyStruct g_struct = { &g_int };
```

Error:

```
constexpr MyStruct g_struct = { &g_int };
                    ^          ~~~~~~~~~~
../../base/memory/checked_ptr_unittest.cc:846:33: note: non-constexpr
constructor 'raw_ptr' cannot be used in a constant expression
constexpr MyStruct g_struct = { &g_int };
                                ^
../../base/memory/raw_ptr_ptr.h:342:17: note: declared here
  ALWAYS_INLINE raw_ptr(T* p) noexcept
```

Sometimes `constexpr` can be avoided altogether, but in general these exclusions are difficult to fix and will likely remain present forever.

```
# constexpr:
$ cat automated-fields-to-ignore.txt | grep '# .*\bconstexpr\b' | wc -l
1015

# constexpr, excluding const-char and unions (which will be ignored by the
anti-raw-ptr-compile-checks):
$ cat automated-fields-to-ignore.txt | grep '# .*\bconstexpr\b' | grep -v '# .*\bconst-char\b' | grep
-v '# .*\bunion\b' | wc -l
381

# constexpr in test code:
$ cat automated-fields-to-ignore.txt | grep '# .*\bconstexpr\b' | grep '# .*\btest\b' |  wc -l
530

# constexpr in const-char:
$ cat automated-fields-to-ignore.txt | grep '# .*\bconstexpr\b' | grep '# .*\bconst-char\b' |  wc -l
614

# 10 random examples:
$ cat automated-fields-to-ignore.txt | grep '# .*\bconstexpr\b' | grep -v 'const-char' | grep -v '#
.*union' | cut -d '#' -f 1 | sort | uniq | sort --ra
ndom-sort | head -10 | sort
blink::(anonymous namespace)::ConversionContext::EffectBoundsInfo::transform
content::CacheQueryResult::characteristic
enterprise_connectors::(anonymous namespace)::TestParam::expected_settings
gpu::InProcessCommandBuffer::InitializeOnGpuThreadParams::image_factory
net::NetLogWithSource::non_null_net_log_
password_manager::(anonymous namespace)::SQLTableBuilders::sync_model_metadata
TabGroupChange::VisualsChange::old_visuals
views::AnimatingLayoutManager::LayoutFadeInfo::child_view
views::ViewHierarchyChangedDetails::parent
web_app::InstallFinalizer::os_integration_manager_
```

in-out and addr-of exclusions

```
# All in-out and addr-of exclusions
$ cat automated-fields-to-ignore.txt | grep '# .*\baddr-of\b\|\bin-out-param-ref\b' | wc -l
516

# All are assumed to be rewritable manually, but the ones important for security are
# in non-test code:
```

```
$ cat automated-fields-to-ignore.txt | grep '# .*\baddr-of\b\|\bin-out-param-ref\b' | grep -v '#
.*\btest\b' | wc -l
168

# 10 random examples:
$ cat automated-fields-to-ignore.txt | grep '# .*\baddr-of\b\|\bin-out-param-ref\b' | grep -v
'const-char' | grep -v '# .*union' | cut -d '#' -f 1 | sort | uniq | sort --random-sort | head -10 | sort
aura::WindowOcclusionTracker::target_occlusion_window_
autofill::AddressField::city_
autofill::(anonymous namespace)::FirstTwoLastNamesField::honorific_prefix_
google::protobuf::FieldDescriptorProto::options_
google::protobuf::FileDescriptorProto::options_
url::URLComponentSource::query
url::URLComponentSource::username
views::examples::DialogExample::cancel_button_label_
views::examples::DialogExample::persistent_bubble_
views::LayoutManager::view_setting_visibility_on_
```

const-char

```
# Helps avoid 1439 additional exclusions
$ cat automated-fields-to-ignore.txt | grep -v '# test-code$' | sed -e 's/# test-code, /# /g' | sed -e
's/, test-code//g' | grep -v '# const-char$' | grep '#.*\bconst-char\b' | wc -l
1439

# Breakdown of the avoided exclusions:
$ cat automated-fields-to-ignore.txt | grep -v '# test-code$' | sed -e 's/# test-code, /# /g' | sed -e
's/, test-code//g' | grep -v '# const-char$' | gr
ep '#.*\bconst-char\b' | cut -f 2 -d '#' | sort | uniq -c | sort -n
      1  addr-of, const-char, union
      1  const-char, constexpr-ctor-field-initializer, embedder-has-no-operator-new
      1  const-char, constexpr-ctor-field-initializer, global-scope, union
      1  const-char, constexpr-var-initializer, global-scope, reinterpret-cast-trivial-type
      1  const-char, embedder-has-no-operator-new, in-out-param-ref
      1  const-char, global-scope, reinterpret-cast-trivial-type
      1  const-char, reinterpret-cast-trivial-type
      2  const-char, constexpr-ctor-field-initializer, constexpr-var-initializer, global-scope, union
      3  const-char, constexpr-ctor-field-initializer, constexpr-var-initializer
      3  const-char, constexpr-ctor-field-initializer, global-scope, in-out-param-ref
      3  const-char, constexpr-ctor-field-initializer, global-scope, reinterpret-cast-trivial-type
      3  const-char, constexpr-ctor-field-initializer, union
      5  const-char, constexpr-var-initializer, in-out-param-ref
      7  const-char, global-scope, overlapping
      7  const-char, global-scope, union
```

```
 13  const-char, constexpr-ctor-field-initializer, constexpr-var-initializer, global-scope
 13  const-char, embedder-has-no-operator-new
 15  const-char, macro
 20  const-char, union
 22  const-char, constexpr-ctor-field-initializer, in-out-param-ref
 26  addr-of, const-char
 31  const-char, constexpr-var-initializer
 33  const-char, global-scope, in-out-param-ref
 72  const-char, constexpr-var-initializer, global-scope
219  const-char, constexpr-ctor-field-initializer
222  const-char, in-out-param-ref
235  const-char, constexpr-ctor-field-initializer, global-scope
478  const-char, global-scope
```

union

Example:

```
union MyUnion {
  raw_ptr<int> ptr_field;
  intptr_t intptr_field;
};
```

Error #1:

```
.../test.cc:11917:11: error: attempt to use a deleted function
  MyUnion u = { .intptr_field = 0 };
          ^
.../test.cc:11912:19: note: destructor of 'MyUnion' is implicitly
deleted because variant field 'ptr_field' has a non-trivial destructor
  raw_ptr<int> ptr_field;
                  ^
```

Error #2:

```
.../test.cc:11917:11: error: call to implicitly-deleted default
constructor of 'base::MyUnion'
  MyUnion u;
          ^
.../test.cc:11912:19: note: default constructor of 'MyUnion' is
implicitly deleted because variant field 'ptr_field' has a non-trivial
default constructor
  raw_ptr<int> ptr_field;
                  ^
```

```
# unions
$ cat automated-fields-to-ignore.txt | grep '# .*\bunion\b' | wc -l
181
```

```
# excluded fields in unions (excluding ones that would have been excluded by const-char rule)
$ cat automated-fields-to-ignore.txt | grep '# .*\bunion\b' | grep -v '# .*\bconst-char\b' | wc -l
147
```