EuroLLVM ’25

Artem Pianykh
Software Engineer @ Meta

Accidentally quadratic in `compiler-rt/asan`

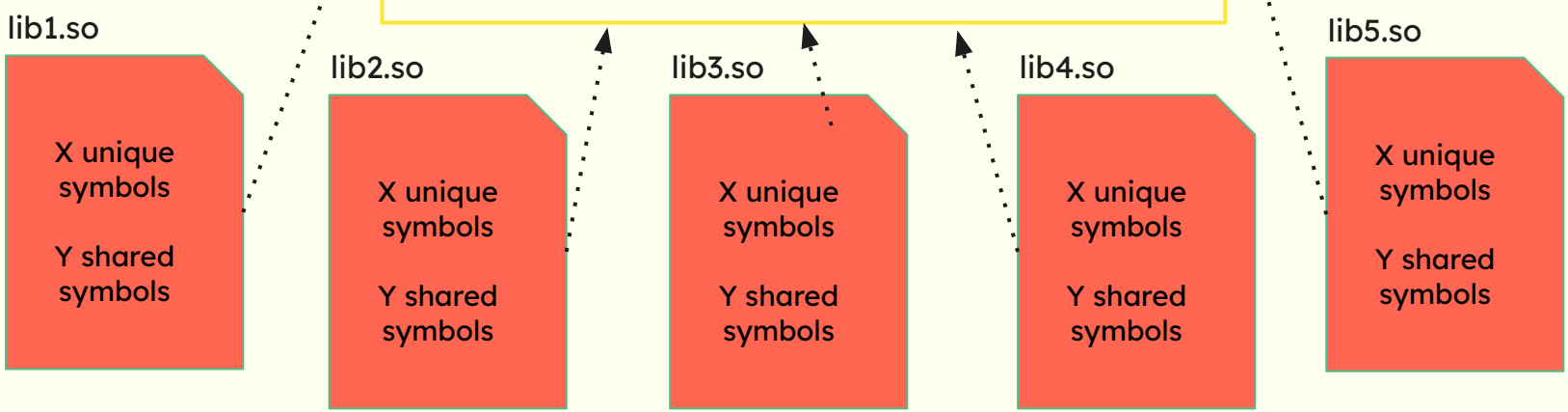
Motivating Example / 1

https://github.com/artempyanykh/eurollvm25

```cpp
#include <iostream>

int main(int argc, char *argv[]) {
   std::cout << "Hello EuroLLVM!" << std::endl;
}

extern "C" const char *__asan_default_options() {
   return "detect_odr_violation=1";
}
```



Motivating Example / 2


```
~/d/eurollvm25 > time ./before_patch
Hello EuroLLVM!


------------------------------------------------------
Executed in    5.70 secs    fish           external
   usr time    5.60 secs    0.00 micros    5.60 secs
   sys time    0.03 secs  518.00 micros    0.03 secs

~/d/eurollvm25 >
```

```
~/d/eurollvm25 > time ./after_patch
Hello EuroLLVM!


------------------------------------------------------
Executed in   96.93 millis    fish           external
   usr time   46.83 millis  263.00 micros   46.57 millis
   sys time   49.78 millis  218.00 micros   49.56 millis

~/d/eurollvm25 >
```

Simplified, but representative of our production

perf report


```
Samples: 22K of event 'cycles:Pu', Event count (approx.): 22956920047
  Children      Self  Command       Shared Object          Symbol
+   99.81%     0.00%  before_patch  ld-linux-x86-64.so.2   [.] _dl_start_user
+   99.42%     0.00%  before_patch  ld-linux-x86-64.so.2   [.] _dl_init
+   99.42%     0.00%  before_patch  ld-linux-x86-64.so.2   [.] call_init
-   99.42%     0.00%  before_patch  before_patch           [.] __asan_register_elf_globals
   - __asan_register_elf_globals
        99.39% __asan_register_globals
+   99.39%    99.10%  before_patch  before_patch           [.] __asan_register_globals
     0.38%     0.00%  before_patch  ld-linux-x86-64.so.2   [.] _dl_start
     0.38%     0.00%  before_patch  ld-linux-x86-64.so.2   [.] _dl_sysdep_start
     0.38%     0.00%  before_patch  ld-linux-x86-64.so.2   [.] dl_main
     0.38%     0.07%  before_patch  ld-linux-x86-64.so.2   [.] _dl_relocate_object
     0.27%     0.10%  before_patch  ld-linux-x86-64.so.2   [.] _dl_lookup_symbol_x
     0.16%     0.15%  before_patch  ld-linux-x86-64.so.2   [.] do_lookup_x
```

ASan is too slow! I better turn it off.

Creator: https://pixy.org/ | Credit: https://pixy.org/530787/
Copyright: CC BY-NC-ND 4.0

~~ASan is too slow! I better turn it off.~~
We need ASan! Let's dig deeper.

Creator: https://pixy.org/ | Credit: https://pixy.org/530787/
Copyright: CC BY-NC-ND 4.0

perf report cont.


```
Samples: 22K of event 'cycles:Pu', 4000 Hz, Event count (approx.): 22956920047
__asan_register_globals  /home/arr/dev/eurollvm25/before_patch [Percent: local period]
Percent
                xorl    %esi,%esi
                movl    $0x0,%ecx
                testq   %rdi,__asan::mu_for_globals
              ↓ je      b41
        7fa:    movq    %r12,%rdi
              → callq   __asan::ReportODRViolation(__asan_global const*, unsigned int, __asa
                nop
   0.02 810:  ┌→movq    0x8(%r15),%r15
   0.55       │ testq   %r15,%r15
   0.14       │↓ je      910
        81d:  │ movq    0x38(%r12),%rcx
   1.28       │ movq    (%r15),%rax
   9.71       │ cmpq    0x38(%rax),%rcx
  88.22       └ jne     810
                movq    $0x54fce0,%rcx
   0.00         cmpl    $0x1,0x64(%rcx)
              ↓ jg      841
                movq    (%rbx),%rcx
                cmpq    0x8(%rax),%rcx
              ↑ je      810
        841:    movq    0x10(%rbx),%rdi
              → callq   __asan::IsODRViolationSuppressed(char const*)
```

Is this a linked list?

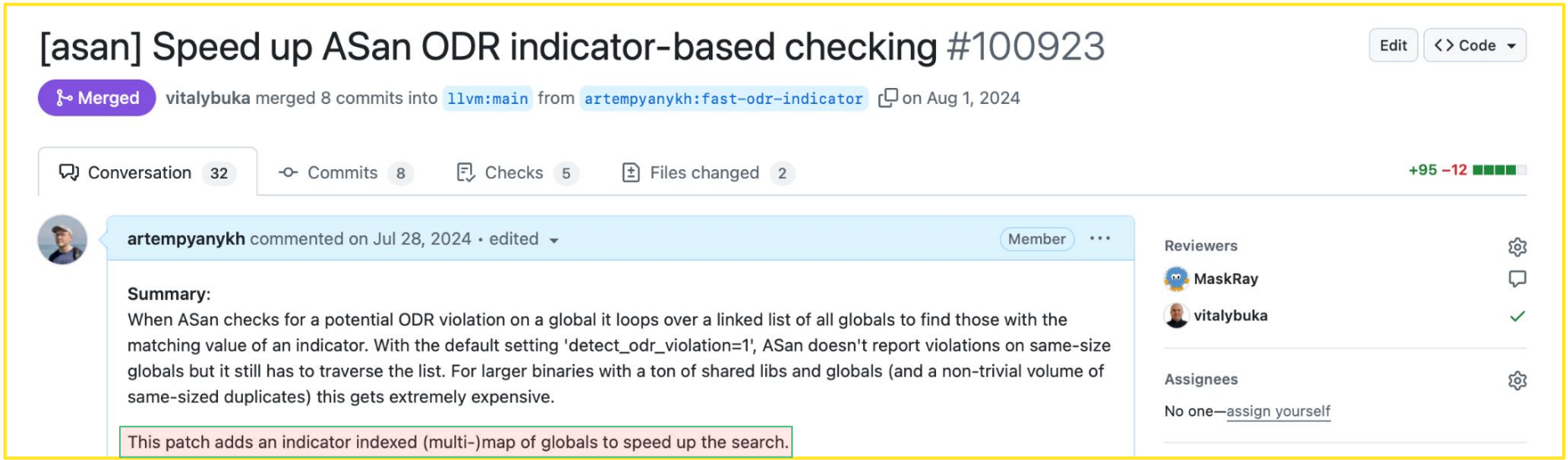
Credits: SVTFOE (c) by Daron Nefcy

Culprit


```
static void CheckODRViolationViaIndicator(const Global *g) {
 …
 // If *odr_indicator is DEFINED …
 for (const auto &l : list_of_all_globals) {
   if (g->odr_indicator == l.g->odr_indicator &&
       (flags()->detect_odr_violation >= 2 || g->size != l.g->size) && !IsODRViolationSuppressed(g->name)) {
     ReportODRViolation(g, FindRegistrationSite(g), l.g,
                        FindRegistrationSite(l.g));
   }
 }
```

*A potential violation triggers iteration over the **list** of **all** globals.*

Fix (list -> map)



[asan] Speed up ASan ODR indicator-based checking #100923

Edit | Code

Merged vitalybuka merged 8 commits into llvm:main from artempyanykh:fast-odr-indicator on Aug 1, 2024

Conversation 32 | Commits 8 | Checks 5 | Files changed 2 | +95 −12

artempyanykh commented on Jul 28, 2024 • edited — Member

Summary:
When ASan checks for a potential ODR violation on a global it loops over a linked list of all globals to find those with the matching value of an indicator. With the default setting 'detect_odr_violation=1', ASan doesn't report violations on same-size globals but it still has to traverse the list. For larger binaries with a ton of shared libs and globals (and a non-trivial volume of same-sized duplicates) this gets extremely expensive.

This patch adds an indicator indexed (multi-)map of globals to speed up the search.

Reviewers
MaskRay
vitalybuka

Assignees
No one—assign yourself

https://github.com/llvm/llvm-project/pull/100923
Thanks to Vitaly Buka for the review and context on the compiler-rt codebase!

Fix (list -> map)


```
38 39 typedef IntrusiveList<GlobalListNode> ListOfGlobals;
.. 40 typedef DenseMap<uptr, ListOfGlobals> MapOfGlobals;
39 41
40 42 static Mutex mu_for_globals;
41 43 static ListOfGlobals list_of_all_globals;
.. 44 static MapOfGlobals map_of_globals_by_indicator;
42 45
43 46 static const int kDynamicInitGlobalsInitialCapacity = 512;
44 47 struct DynInitGlobal {

compiler-rt/lib/asan/asan_globals.cpp --- 3/3 --- C++
149   if (g->odr_indicator == UINTPTR_MAX)
150     return;
...
...
...
...
151   u8 *odr_indicator = reinterpret_cast<u8 *>(g->odr_indicator);
152   if (*odr_indicator == UNREGISTERED) {
153     *odr_indicator = REGISTERED;
154     return;
155   }
156   // If *odr_indicator is DEFINED, some module have already registered
157   // externally visible symbol with the same name. This is an ODR violation.
158   for (const auto &l : list_of_all_globals) {
159     if (g->odr_indicator == l.g->odr_indicator &&
160         (flags()->detect_odr_violation >= 2 || g->size != l.g->size) &&
161         !IsODRViolationSuppressed(g->name)) {
162       ReportODRViolation(g, FindRegistrationSite(g), l.g,
163                          FindRegistrationSite(l.g));
164     }
165   }
...
...
...
...
...
166 }
167

152   if (g->odr_indicator == UINTPTR_MAX)
153     return;
154
155   ListOfGlobals &relevant_globals =
156       map_of_globals_by_indicator[g->odr_indicator];
157
158   u8 *odr_indicator = reinterpret_cast<u8 *>(g->odr_indicator);
159   if (*odr_indicator == REGISTERED) {
...
...
...
160     // If *odr_indicator is REGISTERED, some module have already registered
161     // externally visible symbol with the same name. This is an ODR violation.
162     for (const auto &l : relevant_globals) {
163       if ((flags()->detect_odr_violation >= 2 || g->size != l.g->size) &&
...
164           !IsODRViolationSuppressed(g->name))
165         ReportODRViolation(g, FindRegistrationSite(g), l.g,
166                            FindRegistrationSite(l.g));
...
167     }
168   } else {  // UNREGISTERED
169     *odr_indicator = REGISTERED;
170   }
171
172   AddGlobalToList(relevant_globals, g);
173 }
174
```

Thoughts

Developers' perception

"Sanitizers are too slow" is the most likely reaction. Can we provide better upper bounds and runtime diagnostics?

Benchmarks

We fixed one pathological case. How do we protect from regressions going forward?

ODR checker defaults

The current defaults lead to too many FPs. FNs are also possible. What does "good" mean for the ODR checker?

Thank you!

Artem Pianykh

Presentation reference:
https://github.com/artempyanykh/eurollvm25

Contacts:
artem.pianykh@gmail.com
https://pianykh.com/