Automatically Find Memory Leaks in Native Gems

Peter Zhu

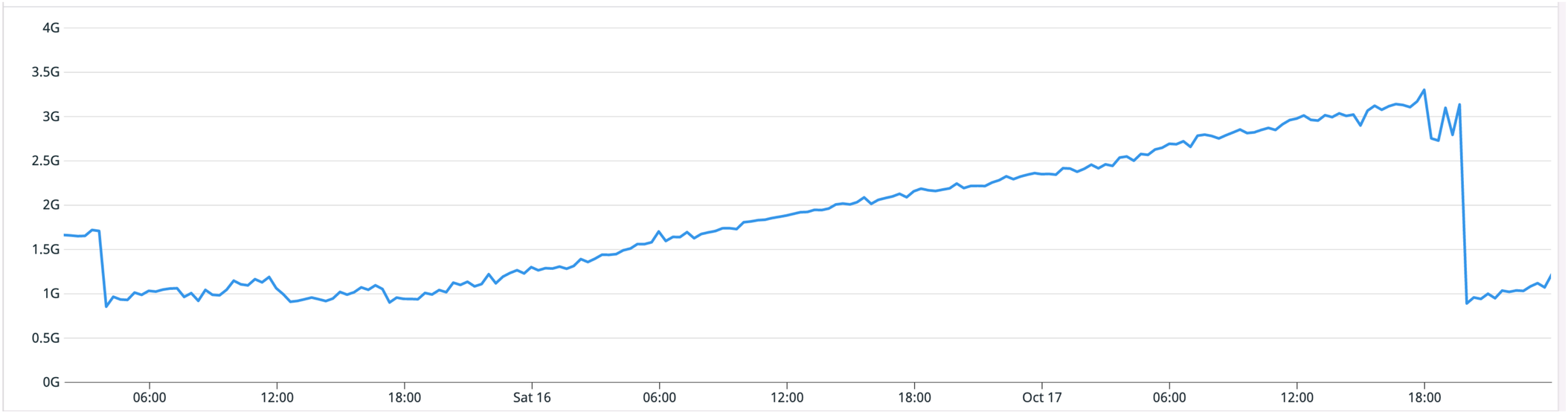
Ruby Core Committer
Senior Developer, Shopify

shopify

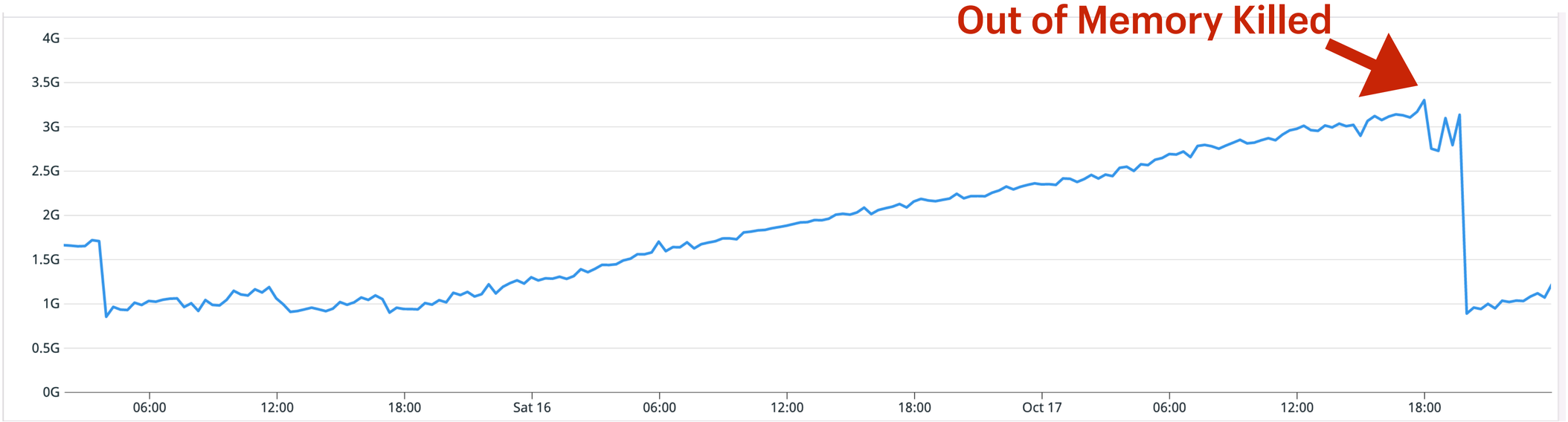
What are native gems?

What’s a memory leak?

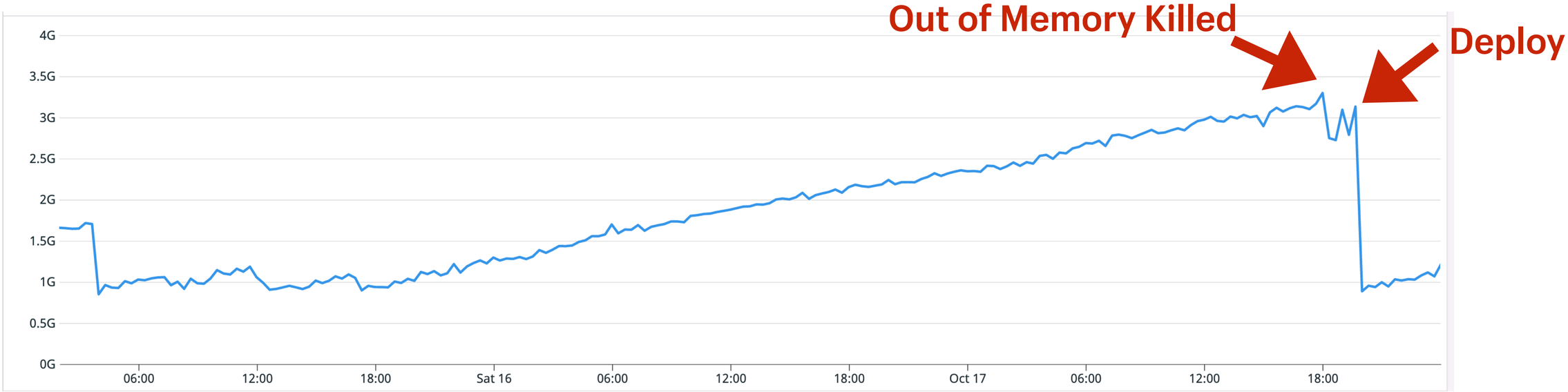
Memory leak in production



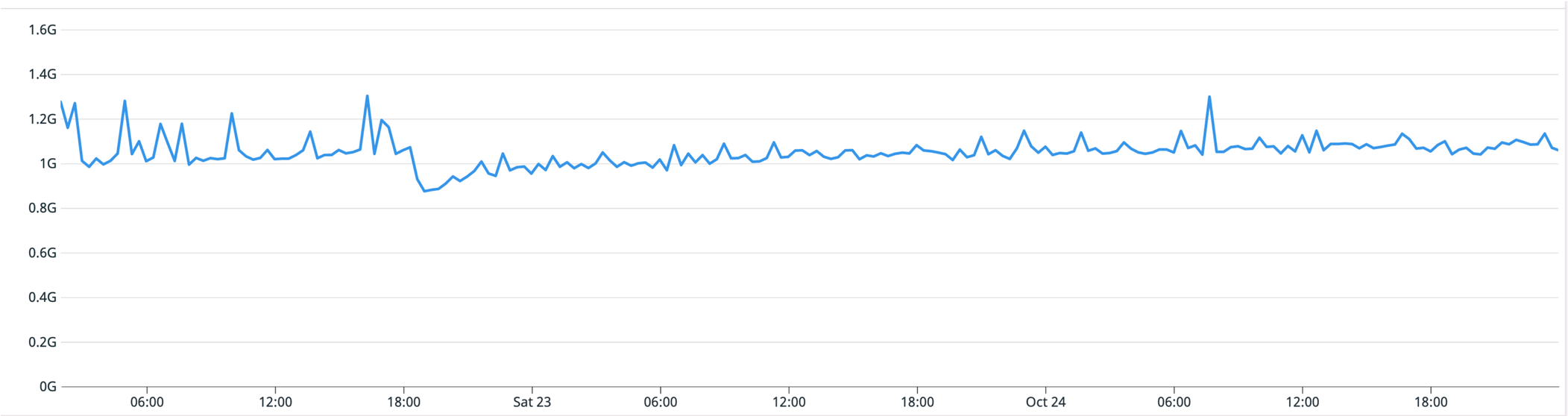
Memory leak in production



Memory leak in production



Fixing the memory leak



How was the memory leak fixed?

Fix memory leak for filters with keyword arguments #157

New issue

Merged peterzhu2118 merged 1 commit into `master` from `pz-kw-arg-mem-leak` on Oct 12, 2021

Conversation 3 Commits 1 Checks 8 Files changed 1 +1 −4

peterzhu2118 commented on Oct 8, 2021 Member

The `push_keywords_obj` (which is a TypedData object) has its data pointer set to NULL and is never freed, which leaks memory. The following script demonstrates the issue:

```
require "liquid"
require_relative "lib/liquid/c"

1_000_000.times do
  Liquid::Template.parse("{{ value | default: 'None', allow_false: true }}")
end

puts `ps -o rss -p #{$$}`.chomp.split("\n").last.to_i
```

Before this patch, the output is `199620`, after this patch, the output is `124160`.

We shouldn't be using `rb_gc_force_recycle` anyways because it doesn't clean up resources and we should let the GC decide when to free objects (rather than forcing the GC to reclaim the object).

 Fix memory leak for filters with keyword arguments ✓ 11f800e

 **peterzhu2118** requested a review from **dylanahsmith** 10 months ago

Reviewers

 **dylanahsmith** ✓

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

Liquid

Safe, customer-facing template language for flexible web apps.

Download View on GitHub

Liquid is an open-source template language created by Shopify and written in Ruby. It is the backbone of Shopify themes and is used to load dynamic content on storefronts.

Liquid has been in production use at Shopify since 2006 and is now used by many other hosted web applications.

Used by

...and many more


jekyll

Shopify / liquid-c Public

master | 45 branches | 7 tags | Go to file | Add file | Code

dylanahsmith Merge pull request #144 from Shopify/inline-comment ...	629377c on Apr 28	370 commits
.github/workflows	Add CI step for Valgrind	9 months ago
ext/liquid_c	Add support for inline comments	3 months ago
lib/liquid	bump gem version to 4.1.0	5 months ago
performance	update rubocop	6 months ago
rakelib	update rubocop	6 months ago
test	Fix test using Ruby 2.7 positional arguments	5 months ago
.gitignore	Generate *.dSYM debug files for MacOS in development for C line n...	2 years ago
.rubocop.yml	update rubocop	6 months ago
Gemfile	update rubocop	6 months ago
LICENSE.txt	Fix copyright header in LICENSE.txt to have Shopify.	8 years ago
README.md	Update benchmark results in README.md	10 months ago
Rakefile	update rubocop	6 months ago
liquid-c.gemspec	Add allowed_push_host to gemspec	5 months ago
performance.rb	update rubocop	6 months ago

README.md

Liquid::C

About

Liquid performance extension in C.

- Readme
- MIT license
- Code of conduct
- 100 stars
- 444 watching
- 15 forks

Releases 2

v4.1.0 Latest
on Feb 2

+ 1 release

Packages

No packages published
Publish your first package

Used by 1.9k

Contributors 18

How could I automatically find memory leaks?

Valgrind memcheck

```
$ valgrind --num-callers=50 --error-limit=no --undef-value-errors=no --leak-check=full --show-leak-kinds=definite ruby -e ""

==2504== Memcheck, a memory error detector
==2504== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==2504== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==2504== Command: ruby -e
==2504==
==2504== Warning: client switching stacks?  SP change: 0x1ffe8020e0 --> 0x1fff000010
==2504==          to suppress, use: --max-stackframe=8380208 or greater
==2504==
==2504== HEAP SUMMARY:
==2504==     in use at exit: 3,285,702 bytes in 20,867 blocks
==2504==   total heap usage: 67,572 allocs, 46,705 frees, 17,742,444 bytes allocated
==2504==
==2504== 8 bytes in 1 blocks are definitely lost in loss record 390 of 11,569
==2504==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==2504==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==2504==    by 0x3C781E: ibf_load_alloc (compile.c:10827)
==2504==    by 0x3C781E: ibf_load_param_keyword (compile.c:11280)
==2504==    by 0x3C781E: ibf_load_iseq_each (compile.c:11856)
==2504==    by 0x3C781E: rb_ibf_load_iseq_complete (compile.c:12748)
==2504==    by 0x3CC001: ibf_load_iseq.isra.0 (compile.c:12803)
==2504==    by 0x3CC4B7: ibf_load_code.isra.0 (compile.c:11166)
==2504==    by 0x3C8A1A: ibf_load_iseq_each (compile.c:11866)
==2504==    by 0x3C8A1A: rb_ibf_load_iseq_complete (compile.c:12748)
==2504==    by 0x3CC001: ibf_load_iseq.isra.0 (compile.c:12803)
```

```
==2504==    by 0x1A7A48: rb_require_string (load.c:1223)
==2504==    by 0x1A7A48: rb_f_require (load.c:904)
==2504==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==2504==    by 0x325E88: vm_call_method_each_type (vm_insnhelper.c:3639)
==2504==    by 0x326753: vm_call_method (vm_insnhelper.c:3750)
==2504==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==2504==    by 0x333A50: vm_exec_core (insns.def:778)
==2504==    by 0x324C72: rb_vm_exec (vm.c:2211)
==2504==    by 0x28F524: ruby_init_prelude (ruby.c:1617)
==2504==    by 0x28F524: ruby_opt_init.part.0 (ruby.c:1642)
==2504==    by 0x29384C: ruby_opt_init (ruby.c:1626)
==2504==    by 0x29384C: process_options (ruby.c:2106)
==2504==    by 0x294578: ruby_process_options (ruby.c:2731)
==2504==    by 0x13C2BC: ruby_options (eval.c:118)
==2504==    by 0x1370B6: main (main.c:47)
==2504==
==2504== LEAK SUMMARY:
==2504==    definitely lost: 543,766 bytes in 4,591 blocks
==2504==    indirectly lost: 856,612 bytes in 9,373 blocks
==2504==      possibly lost: 1,052,616 bytes in 20 blocks
==2504==    still reachable: 832,708 bytes in 6,883 blocks
==2504==         suppressed: 0 bytes in 0 blocks
==2504== Reachable blocks (those to which a pointer was found) are not shown.
==2504== To see them, rerun with: --leak-check=full --show-leak-kinds=all
==2504==
==2504== For lists of detected and suppressed errors, rerun with: -s
```

Valgrind is unusable on Ruby

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

Let’s implement this

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

Valgrind *--xml=yes* flag

```
<error>
  <unique>0×55a3</unique>
  <tid>1</tid>
  <kind>Leak_DefinitelyLost</kind>
  <xwhat>
    <text>5,088 bytes in 53 blocks are definitely lost in loss record 21,924 of 22,145</text>
    <leakedbytes>5088</leakedbytes>
    <leakedblocks>53</leakedblocks>
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

```
  </xwhat>
  <stack>
    <frame>
      <ip>0×483B7F3</ip>
      <obj>/usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so</obj>
      <fn>malloc</fn>
    </frame>
    <frame>
      <ip>0×1609B5</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>objspace_xmalloc0</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>gc.c</file>
      <line>11445</line>
    </frame>
    <frame>
      <ip>0×AC23412</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>vm_assembler_pool_alloc_assembler</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>vm_assembler_pool.c</file>
      <line>59</line>
    </frame>
    <frame>
      <ip>0×AC15942</ip>
      <obj>/home/ubuntu/src/liquid-c/lib/liquid_c.so</obj>
      <fn>block_body_initialize</fn>
      <dir>/home/ubuntu/src/liquid-c/tmp/x86_64-linux/liquid_c/3.1.3/../../../../ext/liquid_c</dir>
      <file>block.c</file>
      <line>108</line>
    </frame>
    <frame>
      <ip>0×32C8A6</ip>
      <obj>/home/ubuntu/.rubies/ruby-3.1.2/bin/ruby</obj>
      <fn>vm_call0_cfunc_with_frame</fn>
      <dir>/home/ubuntu/src/ruby-3.1.2</dir>
      <file>vm_eval.c</file>
      <line>149</line>
    </frame>
...
```

Heuristics ruby_memcheck uses

Heuristics ruby_memcheck uses

Heuristics ruby_memcheck uses

1. Reject leaks that do not contain stack frames from the native gem

Heuristics ruby_memcheck uses

1. Reject leaks that do not contain stack frames from the native gem

Heuristics ruby_memcheck uses

1. Reject leaks that do not contain stack frames from the native gem
2. Reject leaks where the native gem calls back into the Ruby Virtual Machine (e.g. using *rb_funcall* to call a Ruby method, or *rb_raise* to raise an error)

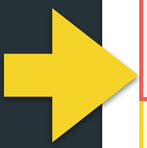
```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

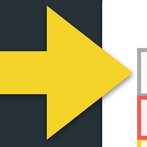


```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 152 (56 direct, 96 indirect) bytes in 1 blocks are definitely lost in loss record 17,613 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0x2A1037: rb_st_init_table_with_size (st.c:539)
==6351==    by 0x2A1037: rb_st_init_table (st.c:577)
==6351==    by 0x176A62: rb_ident_hash_new (hash.c:4433)
==6351==    by 0x16A7E5: rb_hash_aset (hash.c:2937)
==6351==    by 0x336432: vm_opt_aset (vm_insnhelper.c:5401)
==6351==    by 0x336432: vm_exec_core (insns.def:1312)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x330BA9: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x330BA9: rb_funcallv_scope (vm_eval.c:1050)
==6351==    by 0x33967C: rb_funcallv (vm_eval.c:1065)
==6351==    by 0x33967C: rb_funcall (vm_eval.c:1122)
==6351==    by 0xAC0F9CC: context_internal_init (context.c:27)
==6351==    by 0xAC18238: vm_internal_new (vm.c:50)
==6351==    by 0xAC195DD: liquid_vm_render (vm.c:558)
==6351==    by 0xAC0E560: block_body_render_to_output_buffer (block.c:346)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333B6B: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

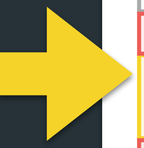
```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```



```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```



```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

```
==6351== 5,184 (5,088 direct, 96 indirect) bytes in 53 blocks are definitely lost in loss record 21,927 of 22,149
==6351==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==6351==    by 0x1609B5: objspace_xmalloc0 (gc.c:11445)
==6351==    by 0xAC1B412: vm_assembler_pool_alloc_assembler (vm_assembler_pool.c:59)
==6351==    by 0xAC0D942: block_body_initialize (block.c:108)
==6351==    by 0x32C8A6: vm_call0_cfunc_with_frame (vm_eval.c:149)
==6351==    by 0x32C8A6: vm_call0_cfunc (vm_eval.c:163)
==6351==    by 0x32C8A6: vm_call0_body (vm_eval.c:209)
==6351==    by 0x32FAAC: vm_call0_cc (vm_eval.c:86)
==6351==    by 0x32FAAC: rb_call0 (vm_eval.c:550)
==6351==    by 0x33056D: rb_call (vm_eval.c:876)
==6351==    by 0x33056D: rb_funcallv_kw (vm_eval.c:1073)
==6351==    by 0x1FA06A: rb_class_new_instance_pass_kw (object.c:1991)
==6351==    by 0x319147: vm_call_cfunc_with_frame (vm_insnhelper.c:3037)
==6351==    by 0x333A50: vm_sendish (vm_insnhelper.c:4751)
==6351==    by 0x333B6B: vm_exec_core (insns.def:759)
==6351==    by 0x324C72: rb_vm_exec (vm.c:2211)
==6351==    by 0x32C672: rb_vm_invoke_proc (vm.c:1521)
==6351==    by 0x232EA4: rb_proc_call_kw (proc.c:991)
==6351==    by 0x232EA4: rb_proc_call (proc.c:1001)
==6351==    by 0x13B74C: exec_end_procs_chain (eval_jump.c:105)
==6351==    by 0x13B74C: rb_ec_exec_end_proc (eval_jump.c:120)
==6351==    by 0x13B964: rb_ec_teardown (eval.c:155)
==6351==    by 0x13BB56: rb_ec_cleanup (eval.c:205)
==6351==    by 0x13C538: ruby_run_node (eval.c:321)
==6351==    by 0x1370BE: main (main.c:47)
```

Heuristics ruby_memcheck uses

1. Reject leaks that do not contain stack frames from the native gem
2. Reject leaks where the native gem calls back into the Ruby Virtual Machine (e.g. using *rb_funcall* to call a Ruby method, or *rb_raise* to raise an error)

Heuristics ruby_memcheck uses

1. Reject leaks that do not contain stack frames from the native gem
2. Reject leaks where the native gem calls back into the Ruby Virtual Machine (e.g. using *rb_funcall* to call a Ruby method, or *rb_raise* to raise an error)
3. Reject leaks in the *Init* function of the native gem

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

```ruby
def skip?
  in_binary = false

  frames.each do |frame|
    fn = frame.fn

    if frame.in_ruby?
      # If a stack from from the binary was encountered first, then this
      # memory leak did not occur from Ruby
      unless in_binary
        # Skip this stack because it was called from Ruby
        return true if configuration.skipped_ruby_functions.any? { |r| r.match?(fn) }
      end
    elsif frame.in_binary?
      in_binary = true

      # Skip the Init function because it is only ever called once, so
      # leaks in it cannot cause memory bloat
      return true if fn == "Init_#{configuration.binary_name}"
    end
  end

  # Skip if the stack was never in the binary because it is very likely
  # not a leak in the native gem
  !in_binary
end
```

Source: https://github.com/Shopify/ruby_memcheck/blob/main/lib/ruby_memcheck/stack.rb

ruby_memcheck finds memory leaks using your test suite

Limitations of ruby_memcheck

Limitations of ruby_memcheck

Limitations of ruby_memcheck

1. Valgrind only works on Linux

Limitations of ruby_memcheck

1. Valgrind only works on Linux

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown
3. Coverage is only as good as your test suite

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown
3. Coverage is only as good as your test suite

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown
3. Coverage is only as good as your test suite
4. It may miss memory leaks in the Ruby Virtual Machine

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown
3. Coverage is only as good as your test suite
4. It may miss memory leaks in the Ruby Virtual Machine

Limitations of ruby_memcheck

1. Valgrind only works on Linux
2. 20-50x slowdown
3. Coverage is only as good as your test suite
4. It may miss memory leaks in the Ruby Virtual Machine
5. It may miss memory leaks where allocation occurred in Ruby

Does this even work?

Shopify / liquid-c Public

Notifications | Fork 15 | Star 100

Code | Issues 9 | Pull requests 12 | Actions | Projects | Wiki | Security | Insights

Fix memory leak for filters with keyword arguments #157

New issue

Merged peterzhu2118 merged 1 commit into `master` from `pz-kw-arg-mem-leak` on Oct 12, 2021

Conversation 3 | Commits 1 | Checks 8 | Files changed 1 +1 −4

peterzhu2118 commented on Oct 8, 2021 Member

The `push_keywords_obj` (which is a TypedData object) has its data pointer set to NULL and is never freed, which leaks memory. The following script demonstrates the issue:

```
require "liquid"
require_relative "lib/liquid/c"

1_000_000.times do
  Liquid::Template.parse("{{ value | default: 'None', allow_false: true }}'
end

puts `ps -o rss -p #{$$}`.chomp.split("\n").last.to_i
```

Before this patch, the output is `199620`, after this patch, the output is `124160`.

We shouldn't be using `rb_gc_force_recycle` anyways because it doesn't clean up resources and we should let the GC decide when to free objects (rather than forcing the GC to reclaim the object).

Reviewers

dylanahsmith ✓

Assignees

No one assigned

Labels

None yet

Projects

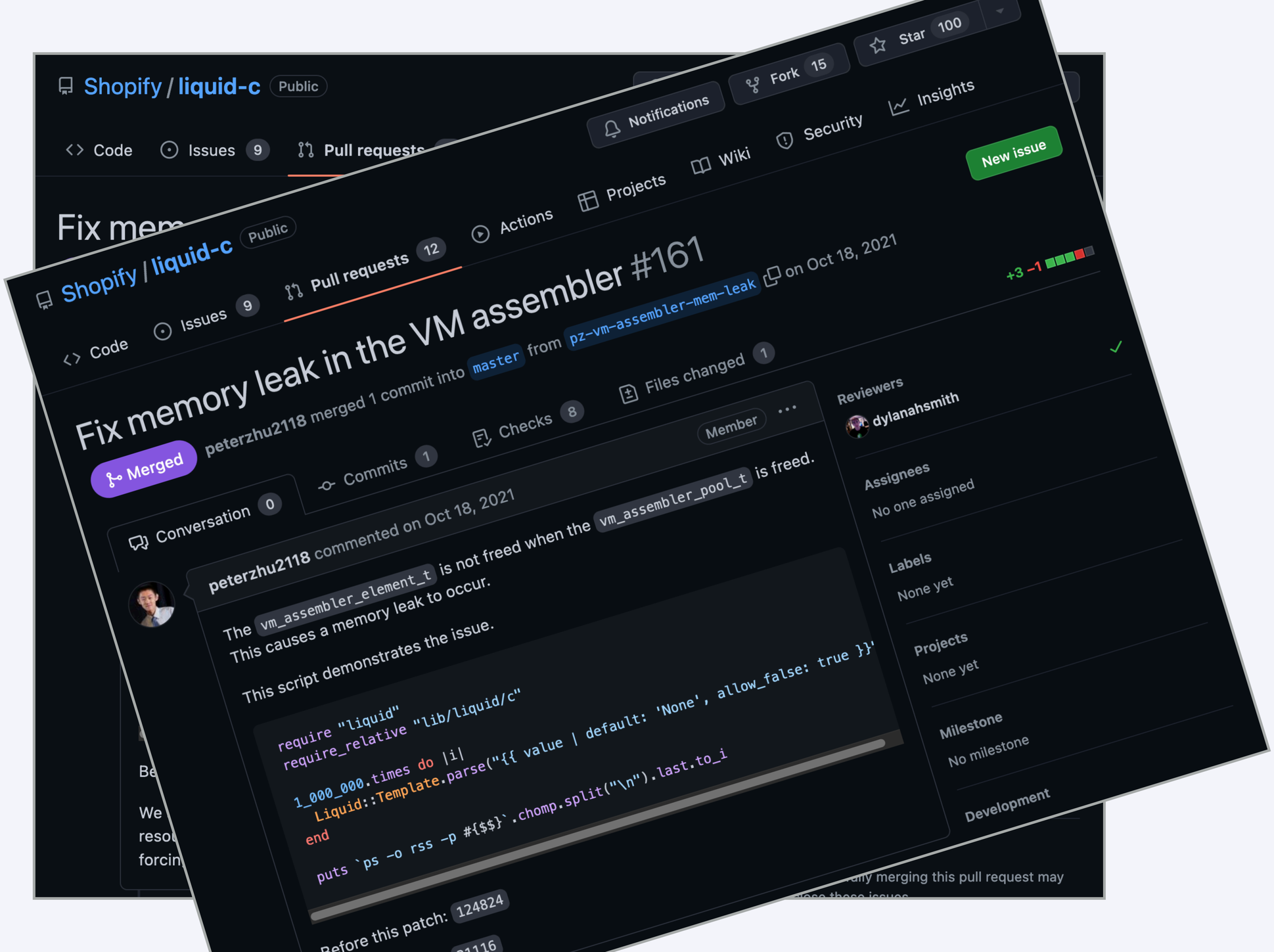
None yet

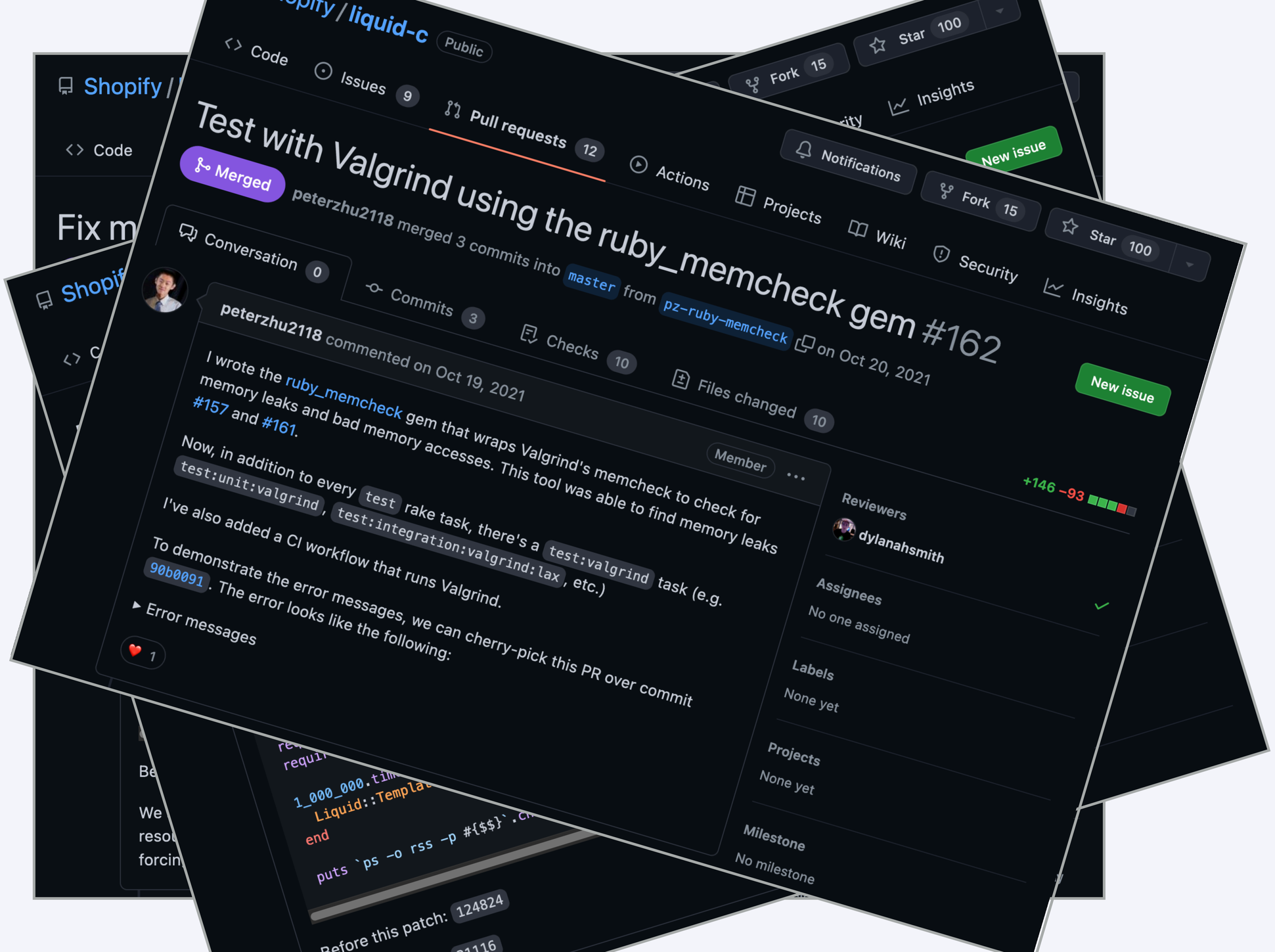
Milestone

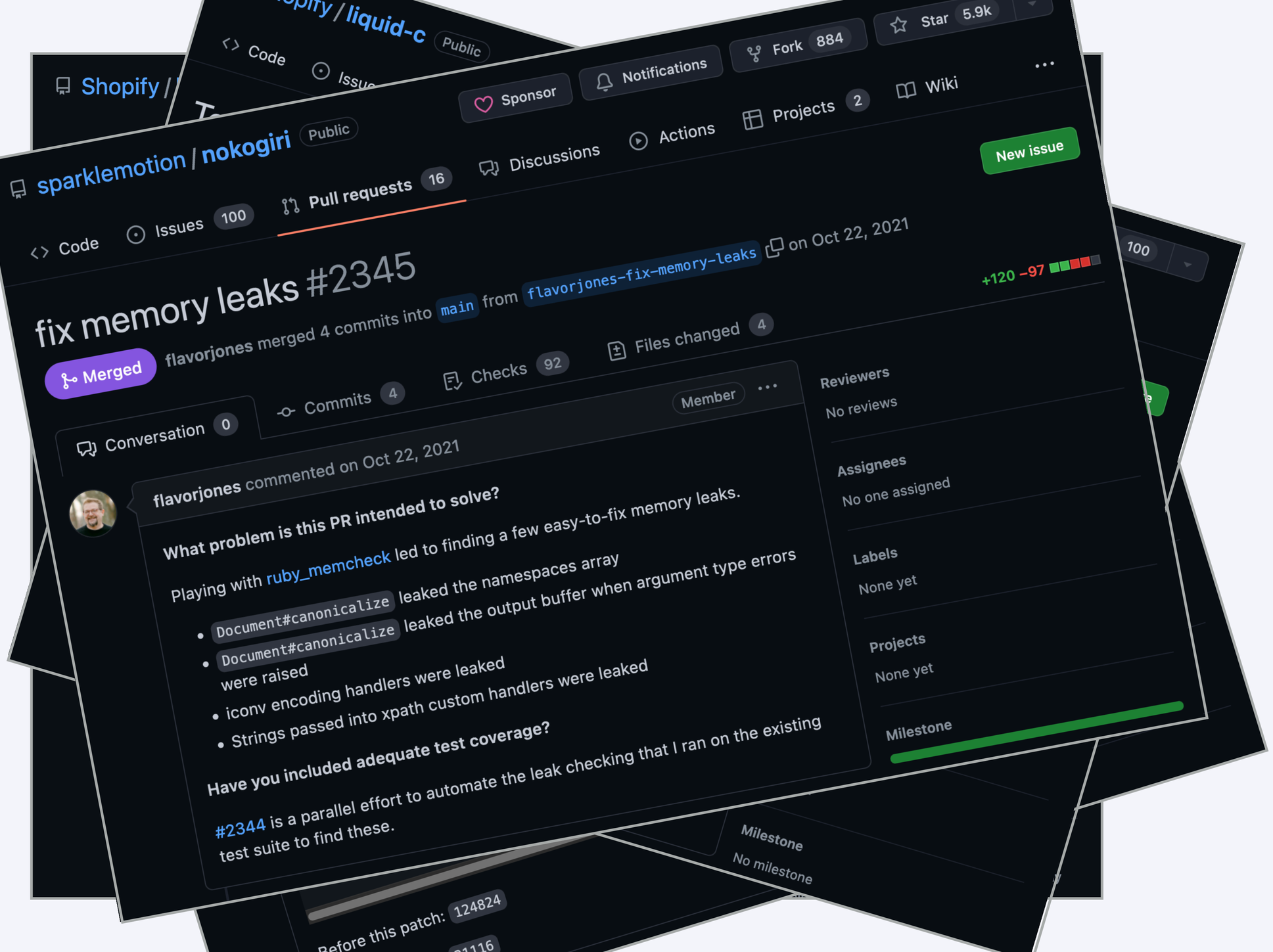
No milestone

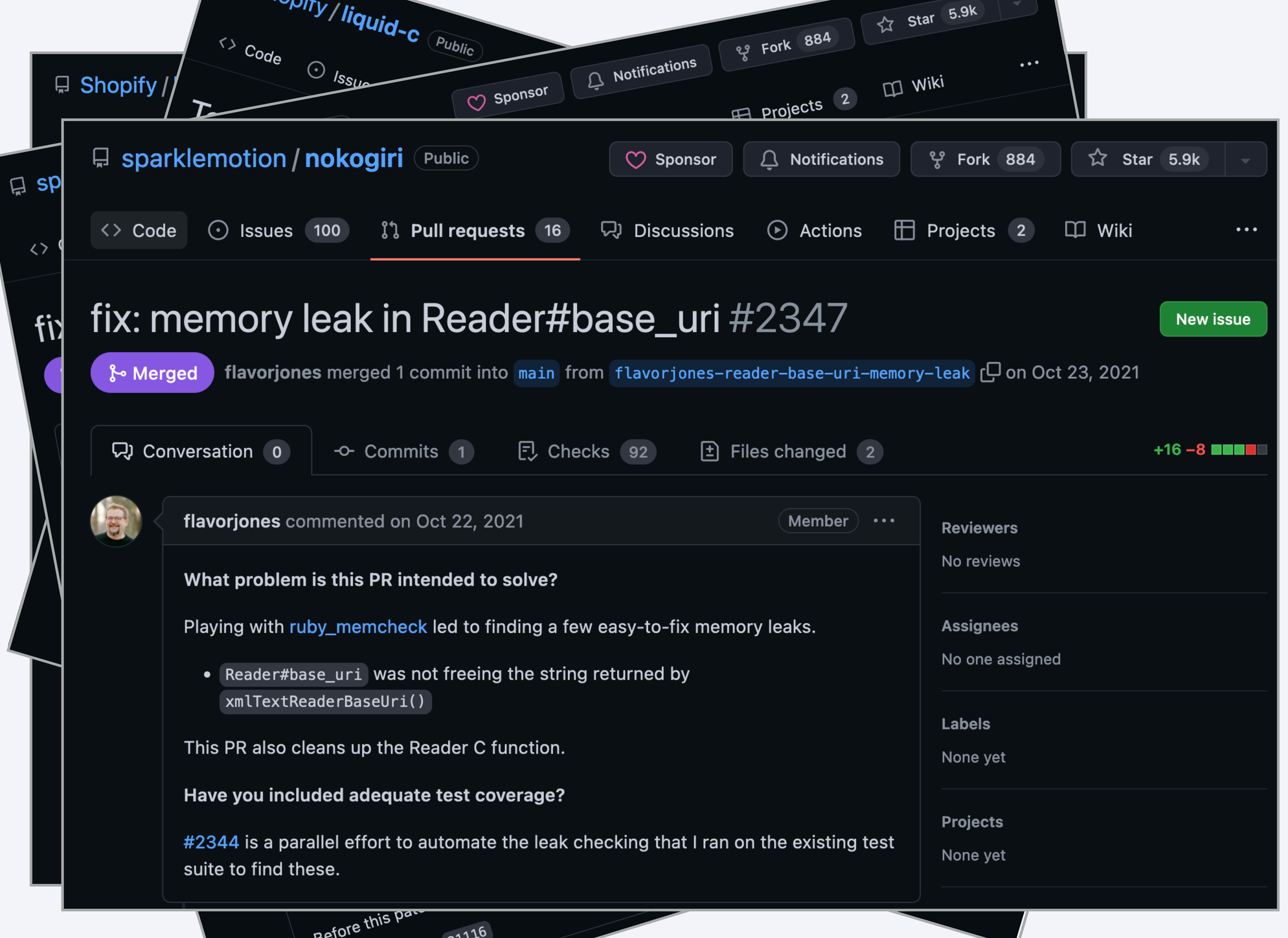
Development

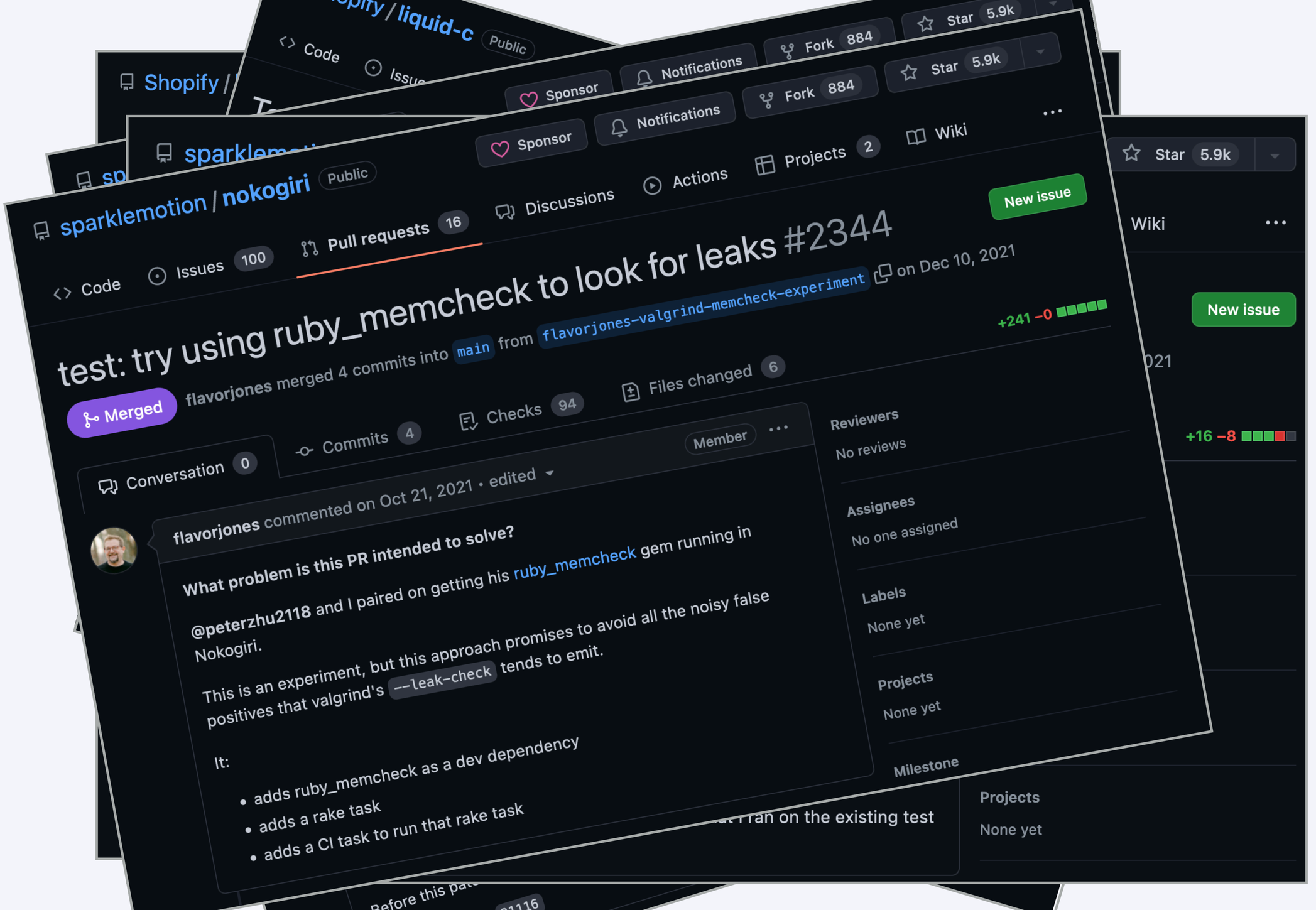
Successfully merging this pull request may close these issues.


Shopify / liquid-c
Public
Notifications
Fork 15
Star 100
Code
Issues 9
Pull requests 12
Actions
Projects
Wiki
Security
Insights
New issue
Fix memory leak in the VM assembler #161
Merged
peterzhu2118 merged 1 commit into master from pz-vm-assembler-mem-leak on Oct 18, 2021
+3 -1
Conversation 0
Commits 1
Checks 8
Files changed 1
peterzhu2118 commented on Oct 18, 2021
Member
The vm_assembler_element_t is not freed when the vm_assembler_pool_t is freed.
This causes a memory leak to occur.
This script demonstrates the issue.
require "liquid"
require_relative "lib/liquid/c"
1_000_000.times do |i|
Liquid::Template.parse("{{ value | default: 'None', allow_false: true }}'
end
puts `ps -o rss -p #{$$}`.chomp.split("\n").last.to_i
Before this patch: 124824
Reviewers
dylanahsmith
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
No milestone
Development


Shopify / liquid-c
Public
Code
Issues 9
Pull requests 12
Actions
Projects
Wiki
Security
Insights
Notifications
Fork 15
Star 100
New issue
Test with Valgrind using the ruby_memcheck gem #162
Merged
peterzhu2118 merged 3 commits into master from pz-ruby-memcheck on Oct 20, 2021
Conversation 0
Commits 3
Checks 10
Files changed 10
peterzhu2118 commented on Oct 19, 2021
Member
I wrote the ruby_memcheck gem that wraps Valgrind's memcheck to check for memory leaks and bad memory accesses. This tool was able to find memory leaks #157 and #161.
Now, in addition to every test rake task, there's a test:valgrind task (e.g. test:unit:valgrind, test:integration:valgrind:lax, etc.)
I've also added a CI workflow that runs Valgrind.
To demonstrate the error messages, we can cherry-pick this PR over commit 90b0091. The error looks like the following:
Error messages
+146 −93
Reviewers
dylanahsmith
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
No milestone
Before this patch: 124824


Shopify / liquid-c
Public
Code
Issues
Sponsor
Notifications
Fork 884
Star 5.9k
sparklemotion / nokogiri
Public
Code
Issues 100
Pull requests 16
Discussions
Actions
Projects 2
Wiki
New issue
fix memory leaks #2345
Merged
flavorjones merged 4 commits into main from flavorjones-fix-memory-leaks on Oct 22, 2021
+120 −97
Conversation 0
Commits 4
Checks 92
Files changed 4
flavorjones commented on Oct 22, 2021
Member
What problem is this PR intended to solve?
Playing with ruby_memcheck led to finding a few easy-to-fix memory leaks.
Document#canonicalize leaked the namespaces array
Document#canonicalize leaked the output buffer when argument type errors were raised
iconv encoding handlers were leaked
Strings passed into xpath custom handlers were leaked
Have you included adequate test coverage?
#2344 is a parallel effort to automate the leak checking that I ran on the existing test suite to find these.
Reviewers
No reviews
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
Milestone
No milestone
Before this patch: 124824


sparklemotion / nokogiri
Public
Sponsor
Notifications
Fork 884
Star 5.9k
Code
Issues 100
Pull requests 16
Discussions
Actions
Projects 2
Wiki
fix: memory leak in Reader#base_uri #2347
New issue
Merged
flavorjones merged 1 commit into main from flavorjones-reader-base-uri-memory-leak on Oct 23, 2021
Conversation 0
Commits 1
Checks 92
Files changed 2
+16 −8
flavorjones commented on Oct 22, 2021
Member
What problem is this PR intended to solve?
Playing with ruby_memcheck led to finding a few easy-to-fix memory leaks.
Reader#base_uri was not freeing the string returned by xmlTextReaderBaseUri()
This PR also cleans up the Reader C function.
Have you included adequate test coverage?
#2344 is a parallel effort to automate the leak checking that I ran on the existing test suite to find these.
Reviewers
No reviews
Assignees
No one assigned
Labels
None yet
Projects
None yet


Shopify / liquid-c
Public
Code
Issues
Pull requests
Discussions
Actions
Projects
Wiki
Security
Insights
Sponsor
Notifications
Fork 884
Star 5.9k
sparklemotion / nokogiri
Public
Issues 100
Pull requests 16
Discussions
Actions
Projects 2
Wiki
Security
Insights
Sponsor
Notifications
Fork 884
Star 5.9k
test: try using ruby_memcheck to look for leaks #2344
New issue
Merged
flavorjones merged 4 commits into main from flavorjones-valgrind-memcheck-experiment on Dec 10, 2021
+241 −0
Conversation 0
Commits 4
Checks 94
Files changed 6
flavorjones commented on Oct 21, 2021 • edited
Member
What problem is this PR intended to solve?
@peterzhu2118 and I paired on getting his ruby_memcheck gem running in Nokogiri.
This is an experiment, but this approach promises to avoid all the noisy false positives that valgrind's --leak-check tends to emit.
It:
adds ruby_memcheck as a dev dependency
adds a rake task
adds a CI task to run that rake task
Reviewers
No reviews
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
Projects
None yet
New issue
+16 −8

[Bug #18264] Fix memory leak in TracePoint #5008

Merged

peterzhu2118 merged 1 commit into ruby:master from Shopify:pz-tracepoint-mem-leak on Oct 26, 2021

Conversation 5 · Commits 1 · Checks 94 · Files changed 2

peterzhu2118 commented on Oct 22, 2021 · Member

Redmine ticket: https://bugs.ruby-lang.org/issues/18264

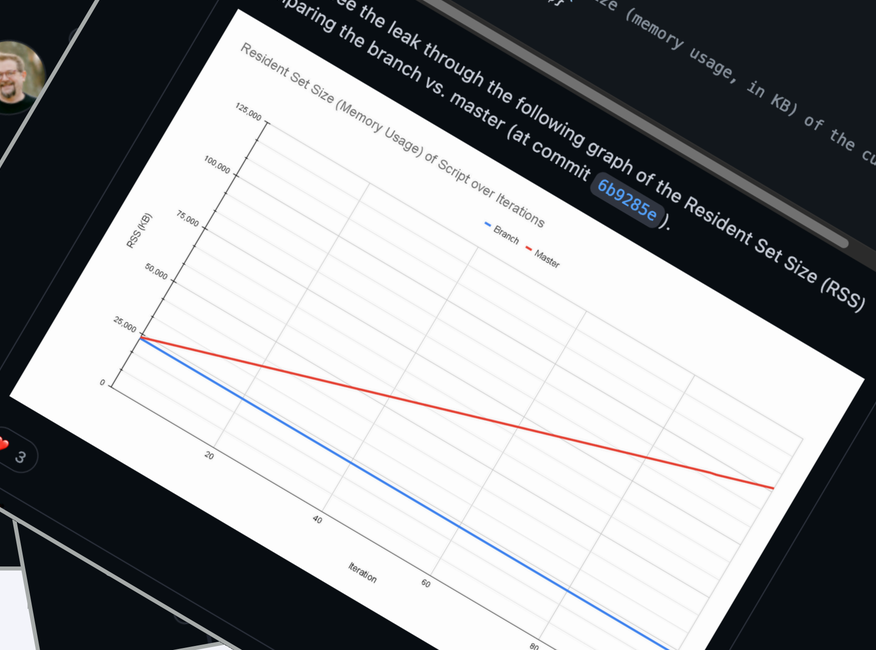
TracePoint leaks memory because it allocates a `rb_tp_t` struct without ever freeing it (it is created with `RUBY_TYPED_NEVER_FREE`). This is reproducible on all maintained Rubies (2.6.8, 2.7.4, 3.0.2, master) on Ubuntu 20.04.

The follow string demonstrates the issue.

```
100.times do
  10000.times do
    TracePoint.new(:line) {}
  end

  # Output the Resident Set Size (memory usage, in KB) of the current Ruby
  puts `ps -o rss= -p #{$$}`
end
```

We can see the leak through the following graph of the Resident Set Size (RSS) comparing the branch vs. master (at commit 6b9285e).



Reviewers: ko1 · Assignees: No one assigned · Labels: None yet · Milestone: No milestone · 4 participants · +11 −1

New issue

Notifications · Fork 6k · Star 19.4k

Pull requests 327 · Actions · Security · Insights

Code · Issues 100 · Merged

test: try using

sparklemotion / nokogiri

Shopify / liquid

Star 5.9k · Fork 884 · Wiki · New issue · +16 −8

Projects: None yet

Shopify / rotoscope Public

Notifications Fork 8 Star 171

Code Issues 3 Pull requests 1 Actions Projects Wiki Security Insights

Add test:valgrind rake task and use it in CI #89

New issue

Merged dylanahsmith merged 4 commits into `main` from `valgrind` on Oct 21, 2021

Conversation 4 Commits 4 Checks 3 Files changed 5 +31 −3

dylanahsmith commented on Oct 21, 2021 Member

To help with detecting memory leaks, I've added a test:valgrind rake task and used it in CI.

- [x] `bin/fmt` was successfully run

Reviewers

jahfer

ChrisBr

Assignees

No one assigned

Projects

None yet

[Bug #18264] Fix memory leak in TracePo

Merged

peterzhu2118 merged 1 commit into `ruby:master` from

Conversation 5 Commits 1

Pull requests 327 Actions Security Insights

Notifications Fork 6k Star 5.9k

sparklemotion / nokogiri

test: try us

Merged

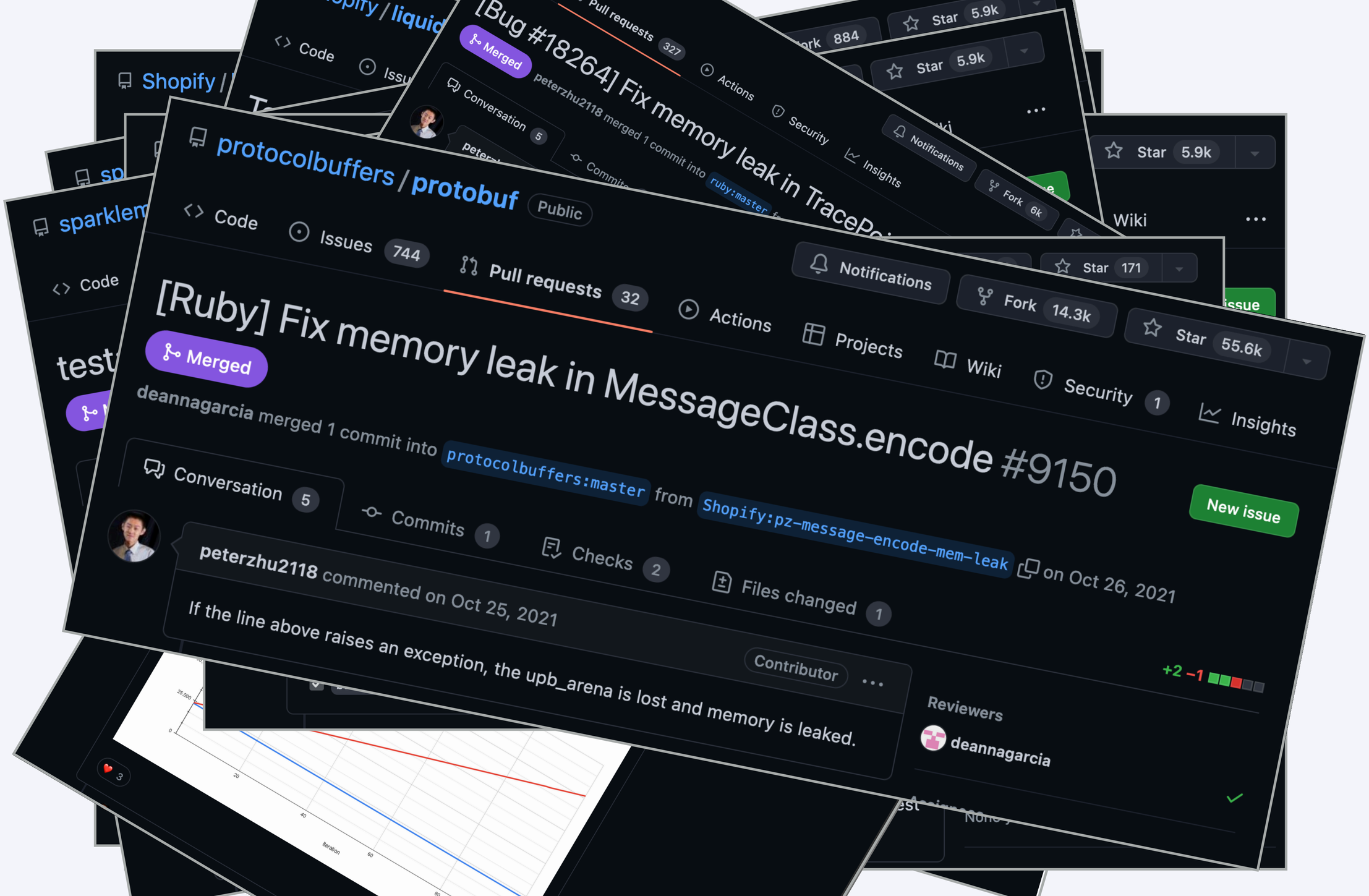
Code Issues

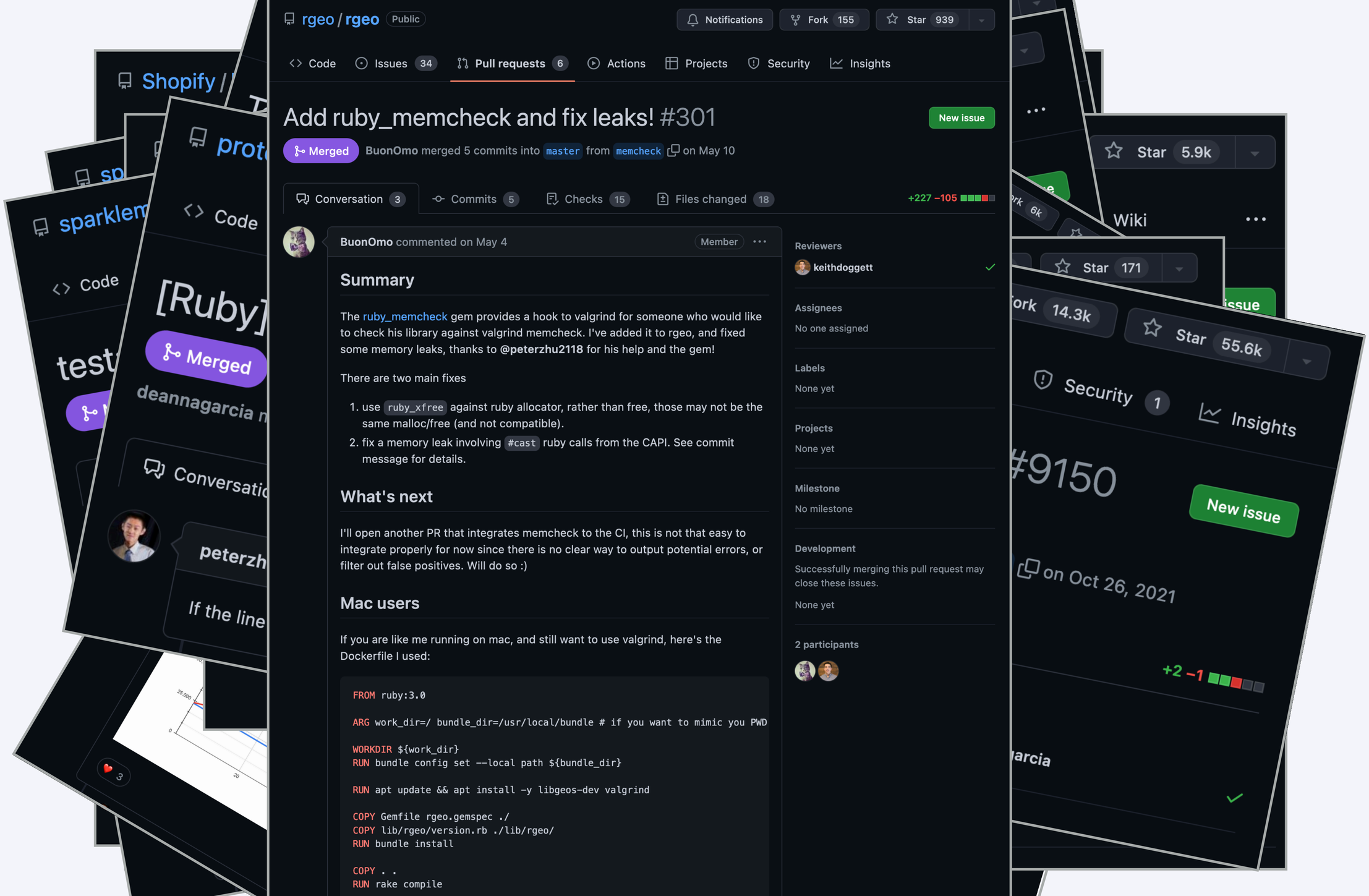
Wiki

Star 5.9k

Fork 884

Star 5.9k


protocolbuffers / protobuf
Public
Code
Issues 744
Pull requests 32
Actions
Projects
Wiki
Security 1
Insights
Notifications
Fork 14.3k
Star 55.6k
[Ruby] Fix memory leak in MessageClass.encode #9150
Merged
deannagarcia merged 1 commit into protocolbuffers:master from Shopify:pz-message-encode-mem-leak on Oct 26, 2021
New issue
Conversation 5
Commits 1
Checks 2
Files changed 1
peterzhu2118 commented on Oct 25, 2021
Contributor
If the line above raises an exception, the upb_arena is lost and memory is leaked.
+2 −1
Reviewers
deannagarcia
[Bug #18264] Fix memory leak in TracePo
Merged
peterzhu2118 merged 1 commit into ruby:master
Pull requests 327
Actions
Security
Insights
Notifications
Fork 6k
Star 5.9k
Wiki
Star 171
Shopify / liquid
Code
Issues
Star 5.9k
Fork 884
Star 5.9k
sparklem
Code
test
Star 5.9k
Wiki
issue
Iteration
25,000
0
20
40
60
80
3


rgeo / rgeo Public
Notifications
Fork 155
Star 939
Code
Issues 34
Pull requests 6
Actions
Projects
Security
Insights
Add ruby_memcheck and fix leaks! #301
New issue
Merged
BuonOmo merged 5 commits into master from memcheck on May 10
Conversation 3
Commits 5
Checks 15
Files changed 18
+227 −105
BuonOmo commented on May 4
Member
Summary
The ruby_memcheck gem provides a hook to valgrind for someone who would like to check his library against valgrind memcheck. I've added it to rgeo, and fixed some memory leaks, thanks to @peterzhu2118 for his help and the gem!
There are two main fixes
1. use ruby_xfree against ruby allocator, rather than free, those may not be the same malloc/free (and not compatible).
2. fix a memory leak involving #cast ruby calls from the CAPI. See commit message for details.
What's next
I'll open another PR that integrates memcheck to the CI, this is not that easy to integrate properly for now since there is no clear way to output potential errors, or filter out false positives. Will do so :)
Mac users
If you are like me running on mac, and still want to use valgrind, here's the Dockerfile I used:
FROM ruby:3.0
ARG work_dir=/ bundle_dir=/usr/local/bundle # if you want to mimic you PWD
WORKDIR ${work_dir}
RUN bundle config set --local path ${bundle_dir}
RUN apt update && apt install -y libgeos-dev valgrind
COPY Gemfile rgeo.gemspec ./
COPY lib/rgeo/version.rb ./lib/rgeo/
RUN bundle install
COPY . .
RUN rake compile
Reviewers
keithdoggett
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Successfully merging this pull request may close these issues.
None yet
2 participants
Star 5.9k
Wiki
Star 171
Fork 14.3k
Star 55.6k
Security 1
Insights
#9150
New issue
on Oct 26, 2021
+2 −1
[Ruby]
Merged
Code
Conversation
If the line
sparklem
Code
Shopify /

Try ruby_memcheck on your native gem!

Thank You

github.com/Shopify/ruby_memcheck

blog.peterzhu.ca/ruby-memcheck/

@peterzhu2118

peter@peterzhu.ca