Ruby - Bug #21187

Strings concatenated with `\` getting frozen with literal hashes (PRISM only)

03/17/2025 09:10 PM - LocoDelAssembly (Hernán Pereira)

Status:	Closed		
Priority:	Normal		
Assignee:	prism		
Target version:			
ruby -v:	ruby 3.4.2 (2025-02-15 revision d2930f8e7a) +PRISM [x86_64-linux]	**Backport:**	3.1: UNKNOWN, 3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: REQUIRED

Description

When the first elements of a literal hash are strings that are concatenated with \, those elements are flagged with PM_NODE_FLAG_STATIC_LITERAL and a special optimization that I believe was introduced in https://github.com/ruby/ruby/commit/8080de04be8e99e71309745822a9d436cc4ae37c causes the strings to be frozen.

Reproduction

test.rb

```
a = {
  a: 'one' \
     'two',
  b: 'three' \
     'four',
  c: 'five',
  d: 'six' \
     'seven'
}

b = {
  a: 'one',
  b: 'two' \
     'three'
}

puts "a = #{a.map { |k,v| {k => v.frozen?} }}"
puts "b = #{b.map { |k,v| {k => v.frozen?} }}"
```

With prism:

```
$ ruby test.rb
a = [{a: true}, {b: true}, {c: false}, {d: false}]
b = [{a: false}, {b: false}]
```

With parse.y:

```
$ ruby --parser=parse.y test.rb
a = [{a: false}, {b: false}, {c: false}, {d: false}]
b = [{a: false}, {b: false}]
```

(Notice b hash is unaffected in both parsers)

Not sure if this is just part of undefined behavior or this is indeed a bug. Assigning a string concatenated with \ to a variable doesn't make it frozen, to the best of my knowledge this seems to be hash-specific.

Associated revisions

Revision a1403fb7 - 07/22/2025 03:10 PM - tenderlovemaking (Aaron Patterson)

Interpolated strings must not be frozen

Strings concatenated with backslash may end up being frozen when they shouldn't be. This commit fixes the issue. It required a change

upstream in Prism, but also a change to the Prism compiler in CRuby.

https://github.com/ruby/prism/pull/3606

[Bug #21187]

Revision 77aaa6ab - 10/01/2025 07:22 PM - Earlopain (Earlopain _)

Interpolation with only string literals must not be frozen

Basically a redo of https://github.com/ruby/ruby/commit/a1403fb7cbd1fe0df97c932be9814c86081783dc
but respecting the frozen string literal magic comment

Fixes [Bug #21187]

History

#1 - 04/14/2025 07:46 AM - alanwu (Alan Wu)

- Assignee set to prism

#2 - 05/12/2025 11:16 PM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Assigned

#3 - 07/22/2025 03:11 PM - tenderlovemaking (Aaron Patterson)

- Status changed from Assigned to Closed

Applied in changeset git|a1403fb7cbd1fe0df97c932be9814c86081783dc.

Interpolated strings must not be frozen

Strings concatenated with backslash may end up being frozen when they
shouldn't be. It required a change
upstream in Prism, but also a change to the Prism compiler in CRuby.

https://github.com/ruby/prism/pull/3606

[Bug #21187]

#4 - 09/13/2025 01:58 PM - kddnewton (Kevin Newton)

- Status changed from Closed to Open

I think this might not be fixed. I'm reopening to keep investigating.

#5 - 09/30/2025 05:12 PM - tenderlovemaking (Aaron Patterson)

@kddnewton (Kevin Newton) any updates, or reproduction issues? I'm confident I fixed the issue OP reported.

#6 - 09/30/2025 05:53 PM - kddnewton (Kevin Newton)

Yeah, it broke CI on macOS https://github.com/ruby/ruby/actions/runs/16430139182/job/46429723627#step:9:179

#7 - 10/01/2025 10:40 AM - Earlopain (Earlopain _)

So this was partially reverted via https://github.com/ruby/ruby/commit/7dbd9c26361719a45fa39838f46a76d67dc3c2e9. It reverts the ruby/prism changes but kept the compiler changes from https://github.com/ruby/ruby/pull/13966.

I think the macos issue might have stemmed from the fact that all flags were cleared. That includes the one from frozen_string_literal: false which rbconfig.rb has. Seems like something that could cause this, and indeed the mutable flag got cleared from "foo""bar" which rbconfig.rb contains on macos

You added a test that "'a' 'b'" should not have PM_NODE_FLAG_STATIC_LITERAL but that doesn't look correct to me? At least from the description about that flag from the docs (A flag to indicate that the value that the node represents is a value that can be determined at parse-time). It does not mean frozen is my understanding.

I feel like the prism side is already correct, just some compiler optimization is missing some extra condition somewhere to not apply. "'a' 'b'" is static, "a" "b#{c}" is not, which is my understanding of how it should be behave.

(For some reason I thought the compiler changes were enough by themselves but I tested now on 3.4.6 and master, and the behaviour OP describes is still present). So, indeed not fixed.

#8 - 10/01/2025 06:26 PM - Earlopain (Earlopain _)

Turns out the prism commit was reverted before the compiler change was merged. Then the ruby/ruby PR was merged, which contained the reverted code again, essentially fixing this issue.

Then https://github.com/ruby/ruby/commit/e74524616013c616744ebf8168f1ad57eee74a05 came along and "broke" it again, since it gave the impression of being reverted during 3.4.6 release. A bit of a mess to be honest.

I did notice an issue with the original approach where these concatinated strings no longer warn about chilled strings. It has to preserve frozen-ness/mutability when concatinating string literals like "foo""bar". I have https://github.com/ruby/prism/pull/3667 for the prism side and https://github.com/ruby/ruby/pull/14697 for the compiler.

#9 - 10/01/2025 07:23 PM - Earlopain (Earlopain _)

- Status changed from Open to Closed

Applied in changeset git|77aaa6ab0a475b8a57b1de4aa1f1210e4f85a803.

Interpolation with only string literals must not be frozen

Basically a redo of https://github.com/ruby/ruby/commit/a1403fb7cbd1fe0df97c932be9814c86081783dc
but respecting the frozen string literal magic comment

Fixes [Bug #21187]

#10 - 10/01/2025 08:29 PM - k0kubun (Takashi Kokubun)

> Turns out the prism commit was reverted before the compiler change was merged.
> A bit of a mess to be honest.

I think the maintainers understand this, but the thing is

- A Prism change that needs a compiler fix should be merged to ruby/ruby first, and then reverse-synced to ruby/prism.
  - When you merge such a change to ruby/prism first and it breaks ruby/ruby CI, the change will be reverted, at least on the ruby/ruby side. This started the out-of-sync situation.
  - If you don't want this to happen, you probably want to have prism_compile.{c,h} in ruby/prism as well. cruby-bindings.yml would test it, so it's not impossible.
- When something is changed on ruby/ruby master (in this case, a sync from ruby/prism was reverted in ruby/ruby), it should be reverse-synced to ruby/prism as soon as possible.
  - There's nothing that prevents you from leaving the out-of-sync ruby/prism at the moment. This time, they were out of sync for a while without anybody noticing it.
  - Perhaps cruby-bindings.yml should fail at the tool/sync_default_gems.rb step when there's a diff not made by the Prism PR itself. It should let you notice ruby/prism is out of sync (due to a revert in ruby/ruby, for example) when it fails there. There's no point in running the CRuby test when it's not going to be the ruby/ruby code after merging the ruby/prism PR.

#11 - 10/01/2025 08:59 PM - Earlopain (Earlopain _)

It was my bad, I did not look at the history and only noticed what happened in what order now. From what I gathered someone was notified of the revert.

Thank you for taking the time to explain the process in such detail. I was uncertain what should happen in which order. ruby/ruby first and clean up after makes sense to me.

#12 - 10/08/2025 10:24 AM - Earlopain (Earlopain _)

I think this should be marked for backport on 3.4. When doing so, cherry-pick changes to prism_compile.c from https://github.com/ruby/ruby/commit/a1403fb7cbd1fe0df97c932be9814c86081783dc and https://github.com/ruby/ruby/commit/77aaa6ab0a475b8a57b1de4aa1f1210e4f85a803.

Then, pulling in the next prism release should make the fix effective. The first commit on its own with only prism_compile.c changes was benign but I can't tell if it would also be the case for the second commit. So I would prefer to backport it all in one go and not start with something incomplete, since my understanding is that prism gem changes should not be cherry-picked.

#13 - 10/08/2025 01:02 PM - kddnewton (Kevin Newton)

- Backport changed from 3.1: UNKNOWN, 3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: UNKNOWN to 3.1: UNKNOWN, 3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: REQUIRED

#14 - 10/08/2025 11:39 PM - k0kubun (Takashi Kokubun)

Thanks for the instructions.

Could we just cut a release of Prism now? I think we could do more releases of Prism, which would make these backport tasks simpler.

I've heard that @kddnewton (Kevin Newton) is okay with automating the gem release with trusted publishing. Once we automate that along with crate/npm releases, I can cut a release myself whenever I need to backport the latest Prism.

#15 - 10/09/2025 01:34 PM - kddnewton (Kevin Newton)

Just released v1.5.2