Ruby - Bug #18164

Segfault after spawn when using modified ENV

09/13/2021 04:08 PM - Fryguy (Jason Frey)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 3.0.2p107 (2021-07-07 revision 0db68f0233) [x86_64-darwin20]	**Backport:**	2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN

Description

The attached segfault.rb causes a segfault on Ruby 3.0.2 (also on 2.7.2+). This is the smallest reproducer we could get.

```
ENV = {}
spawn({}, "true")
ENV.replace({})
```

You can also change the last line to ENV.to_s and it also segfaults.

Note that while this script is the smallest reproducer we could get to, it's unlikely that someone might replace the ENV in this way directly. A more realistic usage scenario (which is how I found this) is using RSpec, having a spec that spawns a subprocess, using stub_const to have an alternate ENV, and using Bundler.with_unbundled_env to ensure that bundler env vars are not passed to the child process. This is demonstrated in the attached segfault_spec.rb. Here, stub_const effectively does the ENV = {} portion, and Bundler.with_unbundled_env does the ENV.replace({}) portion (https://github.com/rubygems/rubygems/blob/b737e1c930aaca15618c702f10553992087e2bc4/bundler/lib/bundler.rb#L693)

Associated revisions

Revision 57d315c937e79199af2b77f21f5eecaca85ffac8 - 09/14/2021 02:55 PM - jeremyevans (Jeremy Evans)

Handle overwriting Object::ENV in spawn

Instead of looking for Object::ENV (which can be overwritten),
directly look for the envtbl variable. As that is static in hash.c,
and the lookup code is in process.c, add a couple non-static
functions that will return envtbl (or envtbl#to_hash).

Fixes [Bug #18164]

Revision 57d315c937e79199af2b77f21f5eecaca85ffac8 - 09/14/2021 02:55 PM - jeremyevans (Jeremy Evans)

Handle overwriting Object::ENV in spawn

Instead of looking for Object::ENV (which can be overwritten),
directly look for the envtbl variable. As that is static in hash.c,
and the lookup code is in process.c, add a couple non-static
functions that will return envtbl (or envtbl#to_hash).

Fixes [Bug #18164]

Revision 57d315c9 - 09/14/2021 02:55 PM - jeremyevans (Jeremy Evans)

Handle overwriting Object::ENV in spawn

Instead of looking for Object::ENV (which can be overwritten),
directly look for the envtbl variable. As that is static in hash.c,
and the lookup code is in process.c, add a couple non-static
functions that will return envtbl (or envtbl#to_hash).

Fixes [Bug #18164]

History

#1 - 09/13/2021 04:38 PM - djberg96 (Daniel Berger)

On my Mac (Big Sur 11.5.2) using Ruby 3.0.2, I can duplicate the segfault with just the first two lines:

```
ENV = {}
```

```
spawn({}, "true")
```

However, this does NOT segfault:

```
ENV.replace({})
spawn({}, "true")
```

#2 - 09/13/2021 04:55 PM - Fryguy (Jason Frey)

This seems to not segfault

```
ENV = {}
spawn({}, "true", :unsetenv_others => true)
ENV.replace({})
```

which leads me to believe that these lines are in play:
https://github.com/ruby/ruby/blob/ebad1e829316de48f212cd57f88639fa5ac55ee4/process.c#L2981-L2982

#3 - 09/13/2021 05:01 PM - Fryguy (Jason Frey)

Speculation ahead, but I'm at the point where I don't understand the Ruby C code anymore...but I think what's happening is, roughly:

https://github.com/ruby/ruby/blob/ebad1e829316de48f212cd57f88639fa5ac55ee4/process.c#L2980-L2984

```
        else {
            envtbl = rb_const_get(rb_cObject, id_ENV);
            envtbl = rb_to_hash_type(envtbl);
        }
        hide_obj(envtbl);
```

I think, in the "normal" ENV case, that the rb_to_hash_type is essentially creating a copy, and so hide_obj is essentially hiding a copy, so no big deal. But in the case of an ENV that is a Hash, rb_to_hash_type is returning the original reference and hiding that instead. Then after, the original object cannot be accessed.

I'll admit, I don't know what hide_obj actually does, and I'm having trouble following it.

#4 - 09/13/2021 06:35 PM - jeremyevans0 (Jeremy Evans)

Fryguy (Jason Frey) wrote in #note-3:

> Speculation ahead, but I'm at the point where I don't understand the Ruby C code anymore...but I think what's happening is, roughly:
>
> https://github.com/ruby/ruby/blob/ebad1e829316de48f212cd57f88639fa5ac55ee4/process.c#L2980-L2984
>
> ```
>         else {
>             envtbl = rb_const_get(rb_cObject, id_ENV);
>             envtbl = rb_to_hash_type(envtbl);
>         }
>         hide_obj(envtbl);
> ```

I agree, that is a bug. It should not be looking for Object::ENV, it should be accessing the internal envtbl static variable. I'm guessing the reason it doesn't is that envtbl is static in hash.c. We probably need to add a non-static function in hash.c to return envtbl as a hash (e.g. rb_env_to_hash), and have the process.c code call that.

#5 - 09/13/2021 07:37 PM - jeremyevans0 (Jeremy Evans)

jeremyevans0 (Jeremy Evans) wrote in #note-4:

> Fryguy (Jason Frey) wrote in #note-3:
>
> > Speculation ahead, but I'm at the point where I don't understand the Ruby C code anymore...but I think what's happening is, roughly:
> >
> > https://github.com/ruby/ruby/blob/ebad1e829316de48f212cd57f88639fa5ac55ee4/process.c#L2980-L2984
> >
> > ```
> >         else {
> >             envtbl = rb_const_get(rb_cObject, id_ENV);
> >             envtbl = rb_to_hash_type(envtbl);
> >         }
> >         hide_obj(envtbl);
> > ```
>
> I agree, that is a bug. It should not be looking for Object::ENV, it should be accessing the internal envtbl static variable. I'm guessing the reason it doesn't is that envtbl is static in hash.c. We probably need to add a non-static function in hash.c to return envtbl as a hash (e.g. rb_env_to_hash), and have the process.c code call that.

I submitted a pull request that takes this approach to fixing the issue: https://github.com/ruby/ruby/pull/4834

#6 - 09/14/2021 02:55 PM - jeremyevans (Jeremy Evans)

- Status changed from Open to Closed

Applied in changeset git|57d315c937e79199af2b77f21f5eecaca85ffac8.

Handle overwriting Object::ENV in spawn

Instead of looking for Object::ENV (which can be overwritten),
directly look for the envtbl variable. As that is static in hash.c,
and the lookup code is in process.c, add a couple non-static
functions that will return envtbl (or envtbl#to_hash).

Fixes [Bug #18164]

Files

segfault.rb	43 Bytes	09/13/2021	Fryguy (Jason Frey)
segfault_spec.rb	255 Bytes	09/13/2021	Fryguy (Jason Frey)