Ruby master - Feature #13613

Prefer that require/require_relative/load to tell us permission error if the target file is unreadable

05/30/2017 11:25 AM - sonots (Naotoshi Seo)

Status:	Feedback	
Priority:	Normal	
Assignee:		
Target version:		

Description

Background

https://github.com/google/google-api-ruby-client/issues/205

We've ever met a situation that read-permissions of files in released google-api-client gem were lost as:

```
$ ls -l ~/.rbenv/versions/2.1.3/lib/ruby/gems/2.1.0/gems/google-api-client-0.8.3/lib/google
-rw-r----- 1 sonots sonots 27249  March 24 18:32 2015 api_client.rb
```

The error message was

```
in `require': cannot load such file -- api_client.rb (LoadError)
```

At that time, it took 30 minutes for me to find the reason why we get LoadError with the gem because the file exists.
I looked the source codes load.c, and finally I found the reason as it is because of permission with my intuition.

What I Want

If require tells us PermissionError in addition to LoadError, I could figure out the reason soon.
I think the additional information is helpful for finding such issues.

Related issues:

Related to Ruby master - Feature #5980: Trying to Load File When Too Many Fil...	**Rejected**

History

#1 - 05/31/2017 12:45 AM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Feedback

There's no such thing like PermissionError. Do you want to introduce one?
Or perhaps you mean Errno::EPERM? Then it sounds slightly odd to me, because other Errno errors such as Errno::ENOENT can be thought of.

#2 - 05/31/2017 01:08 AM - sonots (Naotoshi Seo)

> you mean Errno::EPERM?

Ah, Errno:: EACCES

> other Errno errors such as Errno::ENOENT can be thought of.

Right, I think other errno also should be given.

#3 - 05/31/2017 01:18 AM - sonots (Naotoshi Seo)

- Subject changed from Prefer that require/require_relative/load to tell us PermissionError if the target file is unreadable to Prefer that require/require_relative/load to tell us permission error if the target file is unreadable

#4 - 05/31/2017 02:00 AM - sonots (Naotoshi Seo)

I am thinking to apply a patch like:

```
diff --git a/load.c b/load.c
```

```
index 75ac4df..a8175ca 100644
--- a/load.c
+++ b/load.c
@@ -708,7 +708,7 @@ rb_f_load(int argc, VALUE *argv)
     path = rb_find_file(fname);
     if (!path) {
        if (!rb_file_load_ok(RSTRING_PTR(fname)))
-           load_failed(orig_fname);
+           rb_load_fail(orig_fname, strerror(errno));
        path = fname;
     }
     rb_load_internal(path, RTEST(wrap));
@@ -1049,7 +1049,7 @@ rb_require_safe(VALUE fname, int safe)
        JUMP_TAG(result);
     }
     if (result < 0) {
-       load_failed(fname);
+       rb_load_fail(fname, strerror(errno));
     }

     return result ? Qtrue : Qfalse;
```

WIth this patch, I cloud get an error message for load as:

```
in `load': Permission denied -- foo.rb (LoadError)
```

However, it was not sufficient because require 'foo' first tries to load foo.rb, and if it does not exist or not readable (or any other errors), load foo.so next.
So, we get ENOENT in errno although I want to get EACCESS in this situation.

Any advice is welcome.

#5 - 05/31/2017 02:05 AM - shyouhei (Shyouhei Urabe)

Mmm, this is my private feeling but it seems introducing whole Errno::Efoobar set into require seems to be a drastic change. If we do such change, a formely-legal script like this:

```
begin
  require 'foo'
rescue LoadError
  puts 'cannot load such file'
end
```

should then be rewritten as:

```
begin
  require 'foo'
rescue Errno::ENOENT
  puts 'no such file or directory'
rescue Errno::EISDIR
  puts 'is a directory'
rescue Errno::EACCESS
  puts 'access denied'
rescue Errno::ELOOP
  puts 'too many levels of symbolic links'
rescue ...
  ...
end
```

And it's quite annoying.

#6 - 05/31/2017 02:07 AM - sonots (Naotoshi Seo)

It is not correct. The error class is still LoadError, only error message is changed.

#7 - 05/31/2017 02:11 AM - shyouhei (Shyouhei Urabe)

sonots (Naotoshi Seo) wrote:

> It is not correct. The error class is still LoadError, only error message is changed.

OK, I see what you want.

#8 - 05/31/2017 09:35 AM - shevegen (Robert A. Heiler)

Would be nice - the better error messages are, the easier it may be to solve problems related from that. We had the did_you_mean gem from Yuki Nishijima, perhaps we will one day have a do_you_want_to_fix_this_problem_that_way, like in a circular require warning or rubuocop autocorrection method. :D

So I think the more accurate error messages are, the better.

IMO an error such as "Permission denied" is better than the error "cannot load such file", even though the latter may be the more correct one from a technical point of view.

If necessary one could always slightly extend the error in the latter case such as "cannot load such file - permission denied" or something like that.

#9 - 06/01/2017 04:04 AM - nobu (Nobuyoshi Nakada)

require searches a library across load paths.
When many unreadable files matched, do you want all of them to be reported?
I think it would not make sense unless reporting the full path name.
Maybe warnings for existent but unloadable files?

#10 - 07/04/2017 02:14 PM - rhenium (Kazuki Yamaguchi)

- Related to Feature #5980: Trying to Load File When Too Many Files Open Should Raise Something Other Than Plain LoadError added

#11 - 08/30/2017 10:07 AM - sonots (Naotoshi Seo)

I propose following specification: require 'foobar'

If ENOENT occurs for all paths searched, raise LoadError with messages like below:

```
in `require': cannot load such file -- No such file or directory -- foobar (LoadError)
```

If other errors such as EACCESS are detected, stop searching any more and raise LoadError at that timing with messages like below:

```
in `require': cannot load such file -- Permission denied -- foobar (LoadError)
```

(Permission denied is just an example for EACCESS)

#12 - 09/22/2017 05:42 AM - sonots (Naotoshi Seo)

I tried to write a patch which just adding strerror(errno) to error message.
https://github.com/sonots/ruby/commit/4af8960b65aec1c4a82d81431ed812bcebcdf7f8

But, I noticed that it was very fragile.

```
$:.replace([IO::NULL])
require 'foo'
```

This tells cannot load such file -- Not a directory -- foo (LoadError) rather than cannot load such file -- No such file or directory -- foo (LoadError).
The error message depends on orders of calling syscalls internally, which is very fragile.

#13 - 09/25/2017 12:16 PM - shyouhei (Shyouhei Urabe)

We looked at this issue in a developer meeting today. It seems we need to define "fatality" of each situations for requiring a library. For instance, ENOENT seems less fatal than EACCESS. For ENOENT situation we can silently ignore the error and continue searching in the next LOAD_PATH. However on EACCESS (which means there do exist a file but not readable), it might be worth raising exceptions.

We did not reach consensus whether we should raise then, or to just print warning and continue searching.