Ruby trunk - Bug #14353

$SAFE should stay at least thread-local for compatibility

01/13/2018 07:33 PM - Eregon (Benoit Daloze)

Status:	Open		
Priority:	Normal		
Assignee:	matz (Yukihiro Matsumoto)		
Target version:			
ruby -v:		**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

In #14250 $SAFE changed from a frame+thread-local variable to a process-wide global variable.

This feels wrong and breaks the most common usage of $SAFE in tests:

```
Thread.new {
  $SAFE = 1
  sth that should be checked to work under $SAFE==1
}.join
```

It is very clear this is incompatible given how many files (33!) had to be changed in r61510.
And it has wide ranging confusing side-effects, one example: https://travis-ci.org/ruby/spec/jobs/328524568

I agree frame-local is too much for $SAFE.

But removing thread-local seems to only introduce large incompatibilities.

It also makes it impossible to use it in a thread-safe way.
The common pattern (not necessarily for $SAFE, more often for $VERBOSE):

```
begin
  old = $SAFE
  $SAFE = 1
  something under SAFE==1
ensure
  $SAFE = old
end
```

is unsafe if two threads run it concurrently (The last thread executing $SAFE = old might restore to 1 even though it should be 0).

(Actually I believe most built-in variables (e.g. $VERBOSE) should be thread-local and not process-wide due to this)

Since $SAFE is being deprecated and removed, I don't see any reason to make it more incompatible than needed.

ko1 (Koichi Sasada) Can we switch it back to thread-local for compatibility, avoiding headaches and keeping it usable with multiple threads?

Related issues:

Related to Ruby trunk - Feature #14250: Make `$SAFE` process global state and...	**Closed**

History

#1 - 01/13/2018 07:33 PM - Eregon (Benoit Daloze)

- Assignee set to ko1 (Koichi Sasada)

#2 - 01/13/2018 07:33 PM - Eregon (Benoit Daloze)

- Target version set to 2.6

#3 - 01/13/2018 08:00 PM - Eregon (Benoit Daloze)

- Related to Feature #14250: Make `$SAFE` process global state and allow to set 0 again added

#4 - 01/13/2018 10:23 PM - shevegen (Robert A. Heiler)

I have seen $VERBOSE used in ways very similar as you described though and have used it myself; most commonly I used it e. g. when I would use syck for yaml files rather than psych, in order to suppress warnings I'd get during loading. It never felt very elegant though to use the re-assignment trick. Can't think of a more elegant solution either.

#5 - 01/16/2018 03:14 AM - ko1 (Koichi Sasada)

- Assignee changed from ko1 (Koichi Sasada) to matz (Yukihiro Matsumoto)

Of course, I proposed $SAFE as Fiber local.

However, Matz said process global is enough.
akr also said nobody use $SAFE correctly, so that such incompatibility is not a matter.

I'm neutral. Matz, please decide it.

#6 - 01/16/2018 05:14 PM - Eregon (Benoit Daloze)

> Of course, I proposed $SAFE as Fiber local.

:)

If $SAFE had no effect at all, then I think it would be fine to make $SAFE a normal process-wide global variable.
But this is not the case, $SAFE still causes SecurityError.
Therefore any library/code setting $SAFE will potentially break other threads, which is quite incompatible.

I think a more compatible way to remove $SAFE is:

- Warn that $SAFE is deprecated.
- Make it Fiber-local or Thread-local. (because very few people expect it frame-local and it incurs a cost)
- Warn that $SAFE is has no longer any effect (or maybe even raise when $SAFE is set, like for levels 2-4) and remove $SAFE checks, so the value of $SAFE has no effect.
- Remove the variable entirely or let it be process global.

Something like

```
ruby -e 'Thread.new{ $SAFE=1 }.join; Thread.new{ load "test.rb".taint }.join'
```

fails, prints no warning and could happen in any big application if $SAFE is set to 1 in some Thread.
This doesn't seem worth breaking. Even more so without a warning.

#7 - 12/10/2018 07:08 AM - naruse (Yui NARUSE)

- Target version deleted (2.6)

#8 - 01/23/2019 09:25 AM - vo.x (Vit Ondruch)

Just FTR, as per this discussion:

https://lists.fedoraproject.org/archives/list/ruby-sig@lists.fedoraproject.org/message/V234THAZ2ETTFVLJZXJYPH2QO5W7A3KL/

The global $SAFE breaks gettext testsuite:

https://github.com/ruby-gettext/gettext/commit/49b9f4ca66583395ddfa91503afd7593f069de18

As well as there are issues in postgresql-plruby:

https://github.com/devrimgunduz/postgresql-plruby/blob/master/extconf.rb#L21