Ruby master - Bug #14479

Exceptions raised from a :call tracepoint can sometimes be "rescued" inside the method

02/15/2018 03:09 AM - dazuma (Daniel Azuma)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.5.0p0 (2017-12-25 revision 61468) [x86_64-linux]	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

This is a Ruby 2.5 regression.

If you raise an exception from a :call tracepoint, it can, in certain circumstances, be caught by a rescue block inside the called method. Here is an illustration:

```
####
def foo
begin
puts "hi"
rescue => e
puts "In rescue"
end
end

TracePoint.trace :call do |tp|
raise "kaboom" if tp.method_id == :foo
end

foo
####
```

In Ruby 2.4.3, this results in the exception as expected.
In Ruby 2.5.0, this results in "in rescue" being printed to the console. The rescue block inside method "foo" is catching the exception.

This is highly dependent on the positioning of the rescue block in the method, and may be related to which bytecode is flagged with the trace flag. For example, the following method "foo" raises the exception in Ruby 2.5.0:

```
####
def foo
puts "hi"
begin
puts "hi"
rescue => e
puts "In rescue"
end
end
####
```

Here are three more interesting variants that should be considered:

```
####
def foo
if true
begin
puts "hi"
rescue => e
puts "In rescue"
end
end
end
####
```

Prints "in rescue"

####
def foo
if false
begin
puts "hi"
rescue => e
puts "In rescue"
end
end
end
####
Raises the exception

####
def foo
if false
begin
puts "hi"
rescue => e
puts "In rescue"
end
end
1
end
####
Segfaults!