Ruby master - Bug #14870

Both TracePoint's :call and :c_call filters seem to skip a lot of builtin methods

06/26/2018 05:39 AM - yarmiganosca (Chris Hoffman)

Status:	Open		
Priority:	Normal		
Assignee:	nobu (Nobuyoshi Nakada)		
Target version:			
ruby -v:	2.5.1p57	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

It looks like a lot of builtin methods (Array#<< and Integer#+, for example) aren't hooked when using the :call or :c_call filters for TracePoint.

```
 irb
 :001 > [RUBY_VERSION, RUBY_PATCHLEVEL]
 => ["2.5.1", 57]
 :002 > TracePoint.new(:call, :c_call) { |tp| puts tp.method_id if tp.method_id == :<< }.enable {
[] << 4 }
 => [4]
 :003 > TracePoint.new(:call, :c_call) { |tp| puts tp.method_id if tp.method_id == :concat }.enabl
e { [].concat([4]) }
concat
 => [4]
 :004 > TracePoint.new(:call, :c_call) { |tp| puts tp.method_id if tp.method_id == :+ }.enable { 1
 + 1 }
 => 2
 :005 > TracePoint.new(:call, :c_call) { |tp| puts tp.method_id if tp.method_id == :+ }.enable { 1
.0 + 1.0 }
 => 2.0
```

I can understand if TracePoint being able to hook every method call against every object would be prohibitive from a performance perspective, but the TracePoint documentation currently doesn't indicate that it ignores entire classes of methods, so either the implementation should support every method call (maybe through other event types, if necessary) or the documentation should be clear about what TracePoint filters will and won't hook into.

History

#1 - 06/26/2018 02:26 PM - marcandre (Marc-Andre Lafortune)

- Assignee set to nobu (Nobuyoshi Nakada)

Mmm, that's indeed a bug that should be fixed imo.

This indeed happens because of optimizations for basic operators (in insns.def):

```
TracePoint.new(:call, :c_call) { |tp| puts tp.method_id }.enable { 42 % 2 } # => No printing
# These don't go through the optimizations of insns.def:
TracePoint.new(:call, :c_call) { |tp| puts tp.method_id }.enable { 42.send(:%, 2) } # => %
TracePoint.new(:call, :c_call) { |tp| puts tp.method_id }.enable { 42.modulo(2) } # => modulo
```

I'm confident that Nobu will find the most efficient way to fix this.