Freeze hash literals embedded in duphash instructions

Previously, these hash literals were not frozen, and thus could be modified by ObjectSpace, resulting in undesired behavior. Example (run with --disable-gems):

```ruby
require 'objspace'
def a(b={0=>1,1=>4,2=>17})
  b
end
p a # => {0=>1, 1=>4, 2=>17}
ObjectSpace.each_object(Hash) do |a|
end
p a # => {0=>1, 1=>4, 2=>17, 3=>8}
```

Attached is a patch that freezes such hashes, so attempting to modify the object will raise a FrozenError.

It may be desirable to hide such hashes from ObjectSpace, since they are internal, but I'm not sure how to do that.

Associated revisions:

Revision 2a70f68c - 12/20/2018 07:17 AM - ko1 (Koichi Sasada)
hide iseq operand object for duphash. [Bug #15440]

- compile.c (compile_array): hide source Hash object.
- hash.c (rb_hash_resurrect): introduced to dup Hash object using rb_cHash.

Revision 66466 - 12/20/2018 07:17 AM - ko1 (Koichi Sasada)
hide iseq operand object for duphash. [Bug #15440]

- compile.c (compile_array): hide source Hash object.
- hash.c (rb_hash_resurrect): introduced to dup Hash object using rb_cHash.

Revision 66466 - 12/20/2018 07:17 AM - ko1 (Koichi Sasada)
hide iseq operand object for duphash. [Bug #15440]

- compile.c (compile_array): hide source Hash object.
hash.c (rb_hash_resurrect): introduced to dup Hash object using rb_cHash.

History
#1 - 12/20/2018 07:18 AM - ko1 (Koichi Sasada)
- Status changed from Open to Closed

Applied in changeset trunk|r66466.

hide iseq operand object for duphash. [Bug #15440]

- compile.c (compile_array): hide source Hash object.
- hash.c (rb_hash_resurrect): introduced to dup Hash object using rb_cHash.

Files
0001-Freeze-hash-literals-embedded-in-duphash-instruction.patch 2.25 KB 12/20/2018 jeremyevans0 (Jeremy Evans)