Ruby trunk - Bug #12731

rb_path_to_class should call custom const_defined? methods (take 2)

09/07/2016 12:01 AM - oggy (George Ogata)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.3.1	**Backport:**	2.1: UNKNOWN, 2.2: UNKNOWN, 2.3: UNKNOWN

Description

(This is a continuation of #3511, as I don't see a way to reopen the ticket, as Matz requested there.)

I'm sorry to be a pain, but I've given this more thought and I think this is still worth considering.

My current use case is a Rails application that stores serialized objects to a queue for background processing. The queue data is stored as YAML, and then a separate worker process deserializes the YAML object to process it. This is a common pattern, as used in popular job queue libraries like resque, sidekiq, qless, etc.

Serializing arbitrary objects in the job data is less common, but can still be handy. In our most frequent use case, we push an arbitrary receiver & method name to the queue as a convenient way of pushing a method call to the background. The class of these receivers are often lazy loaded in development.

If I understand the problems/danger you're referring to, this happens when you lazily load constants in parallel threads. My responses would be:

- There are lots of single-threaded rails applications out there for which this risk is not an issue. And even when running multithreaded in production, one might still want to run single-threaded in development to take advantage of lazy loading. I think it should up to the author to understand the risks and solutions here.
- The same risks are present with autoload. I think this is/was under consideration for removal, although for the reasons above I think it should stay, and unless it's going to be removed soon, I think the patch should be applied to make the constant loading behavior of autoload and deserialization *consistent*.
- It's not technically running const_missing that's dangerous here (which is all this patch does), but modifying global state (defining a constant) in separate threads. On that ground, if const_missing exists, it seems wrong to not invoke it when loading a constant from Marshal/YAML.

What do you think?