Ruby master - Feature #8090

resolv.rb checks platform based on RUBY_PLATFORM, which is insufficient for JRuby

03/14/2013 09:06 AM - headius (Charles Nutter)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td>2.6</td>
</tr>
</tbody>
</table>

Description

JRuby shares stdlib with MRI, and as a result we've had to patch a number of things. We'd like to get some of these changes back into MRI's copy.

This issue refers to the following check in resolv.rb:

```ruby
if /mswin|mingw|bccwin/ =~ RUBY_PLATFORM
  require 'win32/resolv'
  DefaultFileName = Win32::Resolv.get_hosts_path
else
  DefaultFileName = '/etc/hosts'
end
```

Because RUBY_PLATFORM on JRuby is always 'java', this check will use the incorrect "hosts" path on Windows. This was reported as an issue in https://github.com/jruby/jruby/issues/580.

We had patched 1.8 stdlib but not 1.9 stdlib to use the following line for checking platform:

```ruby
if /mswin|mingw|bccwin/ =~ RUBY_PLATFORM || ::RbConfig::CONFIG['host_os'] =~ /mswin/
```

I would like to commit this change back to MRI for 1.9.3, 2.0, and 2.1. It does not produce a behavioral change on MRI, since there's no supported platforms that will match /mswin/ for host_os, but it means we won't have to maintain a diff.

Associated revisions

Revision 8046ddf6 - 03/28/2013 02:11 PM - akr (Akira Tanaka)


git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@39988 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 39988 - 03/28/2013 02:11 PM - akira (Akira Tanaka)


Revision 39988 - 03/28/2013 02:11 PM - akira (Akira Tanaka)


Revision 39988 - 03/28/2013 02:11 PM - akira (Akira Tanaka)


Revision 39988 - 03/28/2013 02:11 PM - akira (Akira Tanaka)


Revision 39988 - 03/28/2013 02:11 PM - akr (Akira Tanaka)


History

#1 - 03/14/2013 09:22 AM - nobu (Nobuyoshi Nakada)
Why not try to require 'win32/resolv' and rescue?

#2 - 03/14/2013 10:55 AM - headius (Charles Nutter)
nobu (Nobuyoshi Nakada) wrote:

    Why not try to require 'win32/resolv' and rescue?

    Yep, that works too. The original logic does not do that, so I just expanded on the original condition. I don't have a strong preference for one way versus the other.

#3 - 03/14/2013 02:31 PM - nobu (Nobuyoshi Nakada)
=begin
Because we haven't considered the case (((RUBY_PLATFORM))) means somethings different from CPU and OS.

If you check by (((RbConfig))), (((RUBY_PLATFORM))) is useless, so it should be:

require 'rbconfig'    # loaded by rubygems in default
if /mswin|mingw|bccwin/ =~ ::RbConfig::CONFIG['target_os']
You should use '((%target_os%))' to see the running platform.
=end

#4 - 03/16/2013 02:01 AM - headius (Charles Nutter)
I thought host_os is the OS that we're actually running on, and target_os was the OS name Ruby was built for. Was I mistaken?

I like checking RbConfig::CONFIG alone, if we can decide which key to use.

#5 - 03/28/2013 04:29 PM - nobu (Nobuyoshi Nakada)
"host" has meanings when you build a cross-compiler. You build a compiler on "build" platform, which runs on "host" platform and generates code for "target" platform. So we don't need "host" almost, and often we tweaks "target" in configure.in.

#6 - 03/28/2013 10:09 PM - jonforums (Jon Forums)
nobu (Nobuyoshi Nakada) wrote:

    "host" has meanings when you build a cross-compiler. You build a compiler on "build" platform, which runs on "host" platform and generates code for "target" platform. So we don't need "host" almost, and often we tweaks "target" in configure.in.

I have a different interpretation of the following docs, and also believe host_os, not target_os should be used (at runtime) to determine which OS Ruby is running on.

http://sourceware.org/autobook/autobook/autobook_260.html#SEC260
http://sourceware.org/autobook/autobook/autobook_261.html#SEC261

#7 - 03/28/2013 11:11 PM - akr (Akira Tanaka)
- Status changed from Open to Closed
- % Done changed from 0 to 100

This issue was solved with changeset r39988.
Charles, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.