Ruby trunk - Bug #13769

IPAddr#ipv4_compat incorrect behavior

07/26/2017 04:45 AM - arkadiyt (Arkadiy Tetelman)

Status:	Closed		
Priority:	Normal		
Assignee:	knu (Akinori MUSHA)		
Target version:	2.5		
ruby -v:	ruby 2.2.1p85 (2015-02-26 revision 49769) [x86_64-darwin14]	**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

To ease transition from IPv4 to IPv6, there exist "ipv4-compatible" and "ipv4-mapped" addresses, which are ipv6 addresses that embed an ipv4 address inside them.

Ruby's IPAddr defines several helper functions related to this:
IPAddr#ipv4_mapped? -> return true if the ipaddr is an ipv4-mapped ipv6 address
IPAddr#ipv4_compat? -> return true if the ipaddr is an ipv4-compatible ipv6 address

These 2 formats are defined in RFC4291 section 2.5.5, here:
https://tools.ietf.org/html/rfc4291#section-2.5.5

Notably for ipv4-compatible addresses, it says the following:

```
The "IPv4-Compatible IPv6 address" was defined to assist in the IPv6
   transition.  The format of the "IPv4-Compatible IPv6 address" is as
   follows:

   |                80 bits                | 16 |      32 bits        |
   +---------------------------------------+----+---------------------+
   |0000..............................0000|0000|    IPv4 address     |
   +---------------------------------------+----+---------------------+

   Note: The IPv4 address used in the "IPv4-Compatible IPv6 address"
   must be a globally-unique IPv4 unicast address.
```

But this is not the behavior of IPAddr#ipv4_compat?, defined here:
https://github.com/ruby/ruby/blob/trunk/lib/ipaddr.rb#L267-L274

Given an ipv6 address it checks that the top 96 bits are 0, but then also that the last 32 bits are not equal to 0 or 1, so we have:

```
2.2.1 :002 > IPAddr.new('::0.0.0.0').ipv4_compat?
 => false
2.2.1 :003 > IPAddr.new('::0.0.0.1').ipv4_compat?
 => false
```

It seems like those should return true.

Or:

Perhaps this is related to the last sentence of the RFC: "The IPv4 address used in the "IPv4-Compatible IPv6 address must be a globally-unique IPv4 unicast address.". Since 0.0.0.0 and 0.0.0.1 are not globally-unique unicast addresses, that might justify the false return value. Under this reasoning the function is still wrong in that it *only* returns false for those 2 IPv4 addresses, when there are many other non-globally-unique unicast addresses - see here for some:
https://en.wikipedia.org/wiki/IPv4#Special-use_addresses

It's not clear why 0.0.0.0 and 0.0.0.1 are given this special treatment. The commit was made in 2002 and predates Ruby 1.9.3:
https://github.com/ruby/ruby/commit/9ec0a96ad4235f2054976eab6c04efbe62b3c703

Associated revisions

Revision 3e95ecd8 - 10/21/2017 01:38 PM - knu (Akinori MUSHA)

Fix the issue reference to [Bug #13769], handled in r60270

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@60272 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 60272 - 10/21/2017 01:38 PM - knu (Akinori MUSHA)

Fix the issue reference to [Bug #13769], handled in r60270

Revision 60272 - 10/21/2017 01:38 PM - knu (Akinori MUSHA)

Fix the issue reference to [Bug #13769], handled in r60270

Revision 60272 - 10/21/2017 01:38 PM - knu (Akinori MUSHA)

Fix the issue reference to [Bug #13769], handled in r60270

History

#1 - 08/29/2017 08:02 AM - Glass_saga (Masaki Matsushita)

- Target version set to 2.5

RFC4291 says: '"IPv4-Compatible IPv6 address" is now deprecated.'
We can remove #ipv4_compat? or mark it as deprecated.

Section 2.5.5.1.

The "IPv4-Compatible IPv6 address" is now deprecated because the
current IPv6 transition mechanisms no longer use these addresses.
New or updated implementations are not required to support this
address type.

#2 - 08/29/2017 08:25 AM - arkadiyt (Arkadiy Tetelman)

I'd vote for marking it deprecated & fixing the bug in the original ticket. Despite the fact that the addresses are now deprecated, it may be useful for ruby code to interact with or check for legacy situations - that's how I found the bug

#3 - 08/29/2017 09:06 AM - Glass_saga (Masaki Matsushita)

- Assignee set to knu (Akinori MUSHA)

- Status changed from Open to Assigned

#4 - 08/31/2017 08:58 AM - knu (Akinori MUSHA)

Agreed with deprecating it. Other libraries like the ipaddress gem do not seem to have this, so I'll remove it by 2.5.

#5 - 09/02/2017 06:58 AM - knu (Akinori MUSHA)

I guess :: and ::1 are the only exceptions listed because they are the only IPv6 addresses with the 80+16 bit zero prefix that already have special, conflicting meanings as IPv6 address, so I consider it was practically reasonable enough not to bother with all the other possible non-public IPv4 addresses.

#6 - 10/21/2017 07:07 AM - knu (Akinori MUSHA)

While I do understand the need for dealing with legacy situations, changing the current behavior can cause a subtle bug with rarely maintained code without being noticed immediately.

I'd rather just remove it so that breakage is noticeable because one would soon get a NoMethodError.

#7 - 10/21/2017 07:50 AM - knu (Akinori MUSHA)

I'll mark IPAddr#ipv4_compat and #ipv4_compat? as deprecated in Ruby 2.5 / ipaddr 1.x and remove them in Ruby 2.6 / ipaddr 2.x.

#8 - 10/21/2017 01:37 PM - knu (Akinori MUSHA)

Depreation warning added in r60270. Thanks!

#9 - 10/21/2017 01:38 PM - knu (Akinori MUSHA)

- Status changed from Assigned to Closed

Applied in changeset trunk|r60272.

Fix the issue reference to [Bug #13769], handled in r60270

#10 - 10/23/2017 02:19 AM - arkadiyt (Arkadiy Tetelman)

Thanks!