Ruby - Feature #10911

IPAddr.new should ignore zone identifiers

02/26/2015 11:36 PM - postmodern (Hal Brodigan)

Status:	Closed	
Priority:	Normal	
Assignee:	knu (Akinori MUSHA)	
Target version:		

Description

Link local IPv6 addresses may have a zone identifier suffix:

```
fe80::1%lo0
```

IPAddr.new currently does not ignore the zone identifier and raises IPAddr::InvalidAddressError.

Associated revisions

Revision bd6e1a0f0883dba7b02f30cefe5ebec96d02cb90 - 10/07/2021 09:22 AM - jeremyevans (Jeremy Evans)

[ruby/ipaddr] Support zone identifiers in IPv6 addresses

These are supported by Ruby's socket library if the operating system
supports zone indentifiers, so they should be supported by ipaddr.
See RFCs 4007 and 6874 for additional information.

Implements Ruby Feature #10911

https://github.com/ruby/ipaddr/commit/09a6408fb2

Revision bd6e1a0f0883dba7b02f30cefe5ebec96d02cb90 - 10/07/2021 09:22 AM - jeremyevans (Jeremy Evans)

[ruby/ipaddr] Support zone identifiers in IPv6 addresses

These are supported by Ruby's socket library if the operating system
supports zone indentifiers, so they should be supported by ipaddr.
See RFCs 4007 and 6874 for additional information.

Implements Ruby Feature #10911

https://github.com/ruby/ipaddr/commit/09a6408fb2

Revision bd6e1a0f - 10/07/2021 09:22 AM - jeremyevans (Jeremy Evans)

[ruby/ipaddr] Support zone identifiers in IPv6 addresses

These are supported by Ruby's socket library if the operating system
supports zone indentifiers, so they should be supported by ipaddr.
See RFCs 4007 and 6874 for additional information.

Implements Ruby Feature #10911

https://github.com/ruby/ipaddr/commit/09a6408fb2

History

#1 - 03/20/2019 12:40 PM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Assigned

- Assignee set to knu (Akinori MUSHA)

#2 - 10/31/2019 04:23 PM - jeremyevans0 (Jeremy Evans)

- File ipaddr-ipv6-zone-id-10911.patch added

- Tracker changed from Bug to Feature

- ruby -v deleted (ruby 2.1.5p273 (2014-11-13 revision 48405) [x86_64-linux])

- Backport deleted (2.0.0: UNKNOWN, 2.1: UNKNOWN, 2.2: UNKNOWN)

I don't think this is a bug. The decision to not support zone identifiers seems deliberate as there are tests that using a zone identifier raises an exception:

```
assert_raise(IPAddr::InvalidAddressError) { IPAddr.new("fe80::1%fxp0") }
```

Still, I think this would be a useful feature to add. We should not ignore the zone identifier, as it a property of the address. Attached is a patch that implements support for it, so that to_s and inspect will show the zone identifier. It doesn't consider the zone identifier when determining equality, just as the mask_addr is not currently considered for that. That may be a bug (#11531), but at least it is consistent.

#3 - 10/31/2019 06:37 PM - Dan0042 (Daniel DeLorme)

Looks like this testcase was in the original IPAddr commit from 2002:

```
commit 9ec0a96ad4235f2054976eab6c04efbe62b3c703
Author: knu <knu@b2dd03c8-39d4-4d8f-98ff-823fe69b080e>
Date:   Mon Dec 23 17:07:49 2002 +0000

    * MANIFEST, lib/README, lib/ipaddr.rb: Add ipaddr.rb from rough.
```

Maybe @knu (Akinori MUSHA) remembers why? I'm sure there must have been a reason.

But I think this could break the contract in other places that accept a IPv6 address but not a zone identifier. ex:

```
Socket.getaddrinfo("fe80::1%fxp0", nil) #=> SocketError (getaddrinfo: Name or service not known)
Socket.getaddrinfo("fe80::1", nil)
#=> [["AF_INET6", 0, "fe80::1", "fe80::1", 10, 1, 6], ["AF_INET6", 0, "fe80::1", "fe80::1", 10, 2, 17], ["AF_I
NET6", 0, "fe80::1", "fe80::1", 10, 3, 0]]

TCPSocket.new("fe80::1%fxp0", 42) #=> SocketError (getaddrinfo: Name or service not known)
TCPSocket.new("fe80::1", 42)      #=> Errno::EINVAL (Invalid argument - connect(2) for "fe80::1" port 42)
```

So "fe80::1%fxp0" is not even recognized as an IP address. This could result in unpleasantness in cases like this:

```
str = "fe80::1%fxp0"
ip = IPAddr.new(str) rescue nil   #validate IP address... ok!
TCPSocket.new(ip.to_s, 42) if ip  #and connect... fail!?
```

#4 - 10/31/2019 07:02 PM - jeremyevans0 (Jeremy Evans)

Dan0042 (Daniel DeLorme) wrote:

> So "fe80::1%fxp0" is not even recognized as an IP address. This could result in unpleasantness in cases like this:
>
> ```
> str = "fe80::1%fxp0"
> ip = IPAddr.new(str) rescue nil   #validate IP address
> TCPSocket.new(ip.to_s, 42) if ip  #and connect
> ```

Good point. We would probably want to fix socket to support this before adding support to ipaddr. I think it is reasonable for socket to support it, as tools dealing with IPv6 should handle zone identifiers. I tried ping6 and traceroute6:

```
$ ping6 fe80::1%lo0
PING fe80::1%lo0 (fe80::1%lo0): 56 data bytes
64 bytes from fe80::1%lo0: icmp_seq=0 hlim=64 time=0.630 ms
$ traceroute6 fe80::1
traceroute6 to fe80::1%lo (fe80::1%lo0), 64 hops max, 60 byte packets
 1  fe80::1%lo0 (fe80::1%lo0)  0.162 ms  0.098 ms  0.091 ms
```

Ignoring the zone identifier is not valid:

```
$ ping6 fe80::1
PING fe80::1 (fe80::1): 56 data bytes
ping6: sendmsg: Network is unreachable
$ traceroute6 fe80::1
traceroute6 to fe80::1 (fe80::1), 64 hops max, 60 byte packets
traceroute6: sendto: Network is unreachable
```

#5 - 06/02/2020 05:13 PM - jeremyevans0 (Jeremy Evans)

jeremyevans0 (Jeremy Evans) wrote in #note-4:

> Dan0042 (Daniel DeLorme) wrote:
>
> > So "fe80::1%fxp0" is not even recognized as an IP address. This could result in unpleasantness in cases like this:

```
str = "fe80::1%fxp0"
ip = IPAddr.new(str) rescue nil   #validate IP address
TCPSocket.new(ip.to_s, 42) if ip  #and connect
```

Good point. We would probably want to fix socket to support this before adding support to ipaddr.

It turns out that the socket library already supports this, at least on my operating system (and I would guess other operating systems supporting IPv6 zone identifiers). If I get the IPv6 loopback address with the zone identifier, and use nc to create a listening socket:

```
$ ifconfig lo0 | fgrep fe80
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
$ nc -l fe80::1%lo0 1042
```

I can use that address with Ruby to write to it, as well as get the addrinfo:

```
require 'socket'
t = TCPSocket.new("fe80::1%lo0", 1042)
t.write("foo\n")
t.close

Socket.getaddrinfo("fe80::1%lo0", nil)
# => [["AF_INET6", 0, "fe80::1%lo0", "fe80::1%lo0", 24, 2, 17], ["AF_INET6", 0, "fe80::1%lo0", "fe80::1%lo0",
24, 1, 6]]
```

And foo gets printed to the stdout of the nc process. I've tested and this works back to Ruby 1.8.

So I don't think we need changes to socket, and I think it we should apply the patch to ipaddr (it still applies without conflicts).

#6 - 07/13/2020 04:47 PM - jeremyevans0 (Jeremy Evans)

I've submitted a pull request to the ipaddr repository for this: https://github.com/ruby/ipaddr/pull/24

#7 - 07/21/2020 07:04 PM - jeremyevans0 (Jeremy Evans)

- Status changed from Assigned to Closed

Files

ipaddr-ipv6-zone-id-10911.patch	5.17 KB	10/31/2019	jeremyevans0 (Jeremy Evans)