Ruby - Bug #10203

TCPServer.new has strange behaviour when EADDRINUSE without specifying hostname

09/04/2014 03:20 PM - lucas (Lucas Nussbaum)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.1.2p95	**Backport:**	2.0.0: UNKNOWN, 2.1: UNKNOWN

Description

after:
irb(main):003:0> TCPServer::new(10001)
=> #<TCPServer:fd 9>
I get irb listening to port 10001 using IPv4, but not IPv6:
tcp 0 0 0.0.0.0:10001 0.0.0.0:* LISTEN 1000 376068 24437/irb

a second creation also works, but only binds the IPv6 address:
irb(main):004:0> TCPServer::new(10001)
=> #<TCPServer:fd 10>
tcp6 0 0 :::10001 :::* LISTEN 1000 376098 24437/irb

=> I would have expected the first creation to bind to both IPv4 and IPv6, not just IPv4, and the second attempt to fail.

Trying once again, the creation fails with a strange exception:
irb(main):007:0> TCPServer::new(10001)
TypeError: no implicit conversion of nil into String
from (irb):7:in initialize' from (irb):7:in new'
from (irb):7
from /usr/bin/irb:11:in `'

Binding explicitely to 0.0.0.0 avoids this:
irb(main):005:0> TCPServer::new('0.0.0.0', 10002)
=> #<TCPServer:fd 11>
irb(main):006:0> TCPServer::new('0.0.0.0', 10002)
Errno::EADDRINUSE: Address already in use - bind(2) for "0.0.0.0" port 10002
from (irb):6:in initialize' from (irb):6:in new'
from (irb):6
from /usr/bin/irb:11:in `'

History

#1 - 09/05/2014 07:07 AM - naruse (Yui NARUSE)

I know you was Debian porter, but show the actual ruby -v

#2 - 09/05/2014 07:25 AM - lucas (Lucas Nussbaum)

that was from the Debian package indeed. I haven't rebuilt Ruby from source. The Debian package does not carry any Debian-specific patch at the moment. You cannot reproduce it?

#3 - 09/05/2014 09:12 AM - lucas (Lucas Nussbaum)

Also, a colleague using Arch Linux checked (using the same Ruby version). He doesn't have IPv6 enabled.

And he also got the "TypeError: no implicit conversion of nil into String" exception instead of the Errno::EADDRINUSE one.

#4 - 10/13/2014 06:59 AM - akr (Akira Tanaka)

- Status changed from Open to Feedback

It doesn't reproduce on my environment.

```
% lsb_release -idrc
Distributor ID: Debian
Description: Debian GNU/Linux testing (jessie)
```

```
Release: testing
Codename: jessie
% bin/irb
irb(main):001:0> RUBY_VERSION
=> "2.2.0"
irb(main):002:0> require 'socket'
=> true
irb(main):003:0> TCPServer::new(10001)
=> #<TCPServer:fd 9>
irb(main):004:0> TCPServer::new(10001)
Errno::EADDRINUSE: Address already in use - bind(2) for nil port 10001
 from (irb):4:in `initialize'
 from (irb):4:in `new'
 from (irb):4
 from bin/irb:11:in `<main>'
irb(main):005:0>
```

#5 - 10/13/2014 07:10 AM - lucas (Lucas Nussbaum)

It might be fixed in Ruby 2.2... I don't have an easy way to test. Akira, do you have an easy way to test with Ruby 2.1?

If it is indeed fixed, it might be worth backporting to 2.1.

#6 - 10/13/2014 08:40 AM - normalperson (Eric Wong)

lucas@lucas-nussbaum.net wrote:

> It might be fixed in Ruby 2.2... I don't have an easy way to test. Akira, do you have an easy way to test with Ruby 2.1?
>
> If it is indeed fixed, it might be worth backporting to 2.1.

It may be fixed in trunk r44497.
Lucas: can you apply on your end and confirm? Thanks.

commit 5b0fb1aaddad03e111606e0d0eaf8ed9c9f7b0b6
Author: nobu nobu@b2dd03c8-39d4-4d8f-98ff-823fe69b080e
Date: Sat Jan 4 10:15:31 2014 +0000

```
 socket.c: format flags

 * ext/socket/socket.c (rsock_syserr_fail_host_port): use format flags,
   '+' to inspect, ' ' to quote unprintables.
 * ext/socket/socket.c (rsock_syserr_fail_path): ditto.
 * ext/socket/socket.c (rsock_syserr_fail_raddrinfo): ditto.

 git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@44497 b2dd03c8-39d4-4d8f-98ff-823fe69b080e
```

#7 - 10/13/2014 09:01 AM - akr (Akira Tanaka)

Hm.

Ruby 2.1.3 causes the wrong error message:

```
% ./ruby -vrsocket -e 'p TCPServer::new(10001); p TCPServer::new(10001)'
ruby 2.1.3p242 (2014-09-19 revision 47630) [x86_64-linux]
#<TCPServer:fd 7>
-e:1:in `initialize': no implicit conversion of nil into String (TypeError)
 from -e:1:in `new'
 from -e:1:in `<main>'
```

However it doesn't succeed twice.

#8 - 10/13/2014 11:37 AM - lucas (Lucas Nussbaum)

I built Ruby 2.2 from source, and still encountered the same problem as the original report, with the exception that the bogus error message seems fixed:
$ ./ruby -vrsocket -e 'p TCPServer::new(10001); system("netstat -ln|grep 10001"); p TCPServer::new(10001); system("netstat -ln|grep 10001"); p TCPServer::new(10001)'
ruby 2.2.0dev (2014-10-13 trunk 47902) [x86_64-linux]
#<TCPServer:fd 7>
tcp 0 0 0.0.0.0:10001 0.0.0.0:* LISTEN
#<TCPServer:fd 8>
tcp 0 0 0.0.0.0:10001 0.0.0.0:* LISTEN
tcp6 0 0 :::10001 :::* LISTEN

-e:1:in initialize': Address already in use - bind(2) for nil port 10001 (Errno::EADDRINUSE) from -e:1:in new'
from -e:1:in `'

I would have expected the first creation to bind to both IPv4 and IPv6, not just IPv4, and the second attempt to fail.

#9 - 10/13/2014 11:41 AM - lucas (Lucas Nussbaum)

correct copy/paste:

```
$  ./ruby -vrsocket -e 'p TCPServer::new(10001); system("netstat -ln|grep 10001"); p TCPServer::new(10001); sy
stem("netstat -ln|grep 10001"); p TCPServer::new(10001)'
ruby 2.2.0dev (2014-10-13 trunk 47902) [x86_64-linux]
#<TCPServer:fd 7>
tcp        0      0 0.0.0.0:10001           0.0.0.0:*               LISTEN
#<TCPServer:fd 8>
tcp        0      0 0.0.0.0:10001           0.0.0.0:*               LISTEN
tcp6       0      0 :::10001                :::*                    LISTEN
-e:1:in `initialize': Address already in use - bind(2) for nil port 10001 (Errno::EADDRINUSE)
 from -e:1:in `new'
 from -e:1:in `<main>'
```

#10 - 10/13/2014 12:41 PM - akr (Akira Tanaka)

It seems net.ipv6.bindv6only affect the behavior.

```
% sudo sysctl net.ipv6.bindv6only=0
net.ipv6.bindv6only = 0
% ./ruby -vrsocket -e 'p TCPServer::new(10001); system("netstat -ln|grep 10001"); p TCPServer::new(10001); sys
tem("netstat -ln|grep 10001"); p TCPServer::new(10001)'
ruby 2.2.0dev (2014-10-13 trunk 47899) [x86_64-linux]
#<TCPServer:fd 7>
tcp        0      0 0.0.0.0:10001           0.0.0.0:*               LISTEN
-e:1:in `initialize': Address already in use - bind(2) for nil port 10001 (Errno::EADDRINUSE)
 from -e:1:in `new'
 from -e:1:in `<main>'

% sudo sysctl net.ipv6.bindv6only=1
net.ipv6.bindv6only = 1
% ./ruby -vrsocket -e 'p TCPServer::new(10001); system("netstat -ln|grep 10001"); p TCPServer::new(10001); sys
tem("netstat -ln|grep 10001"); p TCPServer::new(10001)'
ruby 2.2.0dev (2014-10-13 trunk 47899) [x86_64-linux]
#<TCPServer:fd 7>
tcp        0      0 0.0.0.0:10001           0.0.0.0:*               LISTEN
#<TCPServer:fd 8>
tcp        0      0 0.0.0.0:10001           0.0.0.0:*               LISTEN
tcp6       0      0 :::10001                :::*                    LISTEN
-e:1:in `initialize': Address already in use - bind(2) for nil port 10001 (Errno::EADDRINUSE)
 from -e:1:in `new'
 from -e:1:in `<main>'
```

This depends on OS configuration.

TCPServer.new doesn't hide this difference because some OS, such as OpenBSD, provides only the latter behavior.

I recommend Socket.tcp_server_sockets.
It returns two sockets for IPv4 and IPv6, regardless of net.ipv6.bindv6only.

```
% sudo sysctl net.ipv6.bindv6only=0
net.ipv6.bindv6only = 0
% ./ruby -rsocket -e 'p Socket.tcp_server_sockets(10001)'
[#<Socket:fd 7>, #<Socket:fd 8>]
% sudo sysctl net.ipv6.bindv6only=1
net.ipv6.bindv6only = 1
% ./ruby -rsocket -e 'p Socket.tcp_server_sockets(10001)'
[#<Socket:fd 7>, #<Socket:fd 8>]
```

#11 - 10/13/2014 04:50 PM - lucas (Lucas Nussbaum)

Indeed, bindv6only=1 seems to change this. However, bindv6only=1 is kind-of the default on all systems (except some Linux distros).

My goal was to implement the following:
try to bind port 10001 ; if already taken, bind port 10002
That's not possible with the current behaviour.

I don't think it's an OS behaviour. I straced the ruby process, and saw:

first TCPServer call:
[pid 16280] bind(7, {sa_family=AF_INET, sin_port=htons(10001), sin_addr=inet_addr("0.0.0.0")}, 16) = 0
second TCPServer call:
[pid 16280] bind(8, {sa_family=AF_INET, sin_port=htons(10001), sin_addr=inet_addr("0.0.0.0")}, 16) = -1 EADDRINUSE (Address already in use)
[pid 16280] bind(8, {sa_family=AF_INET6, sin6_port=htons(10001), inet_pton(AF_INET6, "::", &sin6_addr), sin6_flowinfo=0, sin6_scope_id=0}, 28) = 0
third TCPServer call:
[pid 16280] bind(9, {sa_family=AF_INET, sin_port=htons(10001), sin_addr=inet_addr("0.0.0.0")}, 16) = -1 EADDRINUSE (Address already in use)
[pid 16280] bind(9, {sa_family=AF_INET6, sin6_port=htons(10001), inet_pton(AF_INET6, "::", &sin6_addr), sin6_flowinfo=0, sin6_scope_id=0}, 28) = -1 EADDRINUSE (Address already in use)

So the second bind() on AF_INET port 10001 fails correctly. It seems that Ruby implements a fallback mechanism that makes it bind the corresponding AF_INET6 port. I think that's wrong. But maybe it's just a documentation issue.

#12 - 10/13/2014 05:10 PM - akr (Akira Tanaka)

Ruby uses getaddrinfo() to obtain addresses to bind.

```
% ./ruby -rsocket -e 'p Addrinfo.getaddrinfo(nil, 10001, nil, :STREAM, nil, Socket::AI_PASSIVE)'
[#<Addrinfo: 0.0.0.0:10001 TCP>, #<Addrinfo: [::]:10001 TCP>]
```

TCPServer.new returns the first socket succeed to bind.

If net.ipv6.bindv6only=1, the first TCPServer.new returns IPv4 socket and the second returns IPv6 socket.

If net.ipv6.bindv6only=0, first TCPServer.new returns IPv4 socket and second fails because
IPv6 socket for 10001 needs IPv4 10001 but it is already used, I guess.

The biggest problem here is TCPServer.new returns only one socket.
I think TCP server should listen all addresses obtained by getaddrinfo() but it is impossible with TCPServer.new().

So, my suggestion again:
I recommend Socket.tcp_server_sockets.

#13 - 10/13/2014 05:35 PM - lucas (Lucas Nussbaum)

I was not interested in adding IPv6 support to my application, only in binding a port with a fallback mechanism. I worked around the problem with bind('0.0.0.0', 10001).

But I still think that it is wrong that TCPServer.new implements this fallback mechanism. It would be less surprising if it just tried to bind the first address, and fail if unsuccessful. I can't think of a scenario where this is useful.

#14 - 10/14/2014 12:15 PM - akr (Akira Tanaka)

I'm not sure that ignoring getaddrinfo()'s non-first entries is a good behavior.

#15 - 03/06/2021 12:52 AM - jeremyevans0 (Jeremy Evans)

- Status changed from Feedback to Closed

I think this is not a bug and can be closed. TCPServer.new using first getaddrinfo address that works is reasonable behavior if no address is specified. If you require a specific address, you should specify it. As @akr (Akira Tanaka) mentioned, Socket.tcp_server_sockets is a better way to handle the situation if you want to listen on both IPv4 and IPv6 addresses.