Ruby master - Bug #8159

Build failure introduced by Rinda changes

03/24/2013 10:53 PM - luislavena (Luis Lavena)

Status:	Closed	
Priority:	Normal	
Assignee:	usa (Usaku NAKAMURA)	
Target version:	2.1.0	
ruby -v:	ruby 2.1.0dev (2013-03-24 trunk 39905) [x64-mingw32]	**Backport:**

Description

=begin

Seems latest Rinda-related changes affected build under Windows:

http://ci.rubyinstaller.org/job/ruby-trunk-x64-test-all/936/console

2) Error:
test_take_bug_8215(Rinda::TupleSpaceProxyTest):
NotImplementedError: fork() function is unimplemented on this machine
C:/Users/Worker/Jenkins/workspace/ruby-trunk-x64-build/test/rinda/test_rinda.rb:486:in fork'
C:/Users/Worker/Jenkins/workspace/ruby-trunk-x64-build/test/rinda/test_rinda.rb:486:intest_take_bug_8215'

3) Error:
test_make_socket_ipv4_multicast(Rinda::TestRingServer):
Errno::EADDRNOTAVAIL: The requested address is not valid in its context. - bind(2)
C:/Users/Worker/Jenkins/workspace/ruby-trunk-x64-build/lib/rinda/ring.rb:117:in bind'
C:/Users/Worker/Jenkins/workspace/ruby-trunk-x64-build/lib/rinda/ring.rb:117:inmake_socket'
C:/Users/Worker/Jenkins/workspace/ruby-trunk-x64-build/test/rinda/test_rinda.rb:542:in `test_make_socket_ipv4_multicast'

r39895 seems to have introduced a test that is not skipping on non-fork() platforms.

=end

Related issues:

Related to Ruby master - Feature #8073: Add multicast support to Rinda::Ring*	**Closed**	**03/11/2013**	
Blocks Ruby master - Misc #8288: Ruby 2.1.0 release engeneering	**Closed**	**04/18/2013**	**12/25/2013**

Associated revisions

Revision 77b88526 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@39922 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 39922 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

Revision 39922 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

Revision 39922 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

Revision 39922 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

Revision 39922 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

Revision 39922 - 03/24/2013 10:01 PM - drbrain (Eric Hodel)

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

Revision a850dc62 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@39930 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 39930 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

Revision 39930 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

Revision 39930 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

Revision 39930 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

Revision 39930 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

Revision 39930 - 03/25/2013 06:50 PM - naruse (Yui NARUSE)

Use more general approach to get scope_id see #8159

Revision 93ed9f08 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional arguments for multicast interface.
- test/rinda/test_rinda.rb (TestRingFinger#test_ring_server_ipv4_multicast, TestRingFinger#test_ring_server_ipv6_multicast): add tests for above change.
- test/rinda/test_rinda.rb (TestRingServer#test_make_socket_ipv4_multicast, TestRingServer#test_make_socket_ipv6_multicast): change bound interface address because multicast address is not allowed on Linux or Windows.
  [ruby-core:53692] [Bug #8159]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@40472 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 40472 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional arguments for multicast interface.
- test/rinda/test_rinda.rb

(TestRingFinger#test_ring_server_ipv4_multicast, TestRingFinger#test_ring_server_ipv6_multicast): add tests for above change.

- test/rinda/test_rinda.rb (TestRingServer#test_make_socket_ipv4_multicast, TestRingServer#test_make_socket_ipv6_multicast): change bound interface address because multicast address is not allowed on Linux or Windows.
  [ruby-core:53692] [Bug #8159]

Revision 40472 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional arguments for multicast interface.
- test/rinda/test_rinda.rb (TestRingFinger#test_ring_server_ipv4_multicast, TestRingFinger#test_ring_server_ipv6_multicast): add tests for above change.
- test/rinda/test_rinda.rb (TestRingServer#test_make_socket_ipv4_multicast, TestRingServer#test_make_socket_ipv6_multicast): change bound interface address because multicast address is not allowed on Linux or Windows.
  [ruby-core:53692] [Bug #8159]

Revision 40472 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional arguments for multicast interface.
- test/rinda/test_rinda.rb (TestRingFinger#test_ring_server_ipv4_multicast, TestRingFinger#test_ring_server_ipv6_multicast): add tests for above change.
- test/rinda/test_rinda.rb (TestRingServer#test_make_socket_ipv4_multicast, TestRingServer#test_make_socket_ipv6_multicast): change bound interface address because multicast address is not allowed on Linux or Windows.
  [ruby-core:53692] [Bug #8159]

Revision 40472 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional arguments for multicast interface.
- test/rinda/test_rinda.rb (TestRingFinger#test_ring_server_ipv4_multicast, TestRingFinger#test_ring_server_ipv6_multicast): add tests for above change.

- test/rinda/test_rinda.rb
  (TestRingServer#test_make_socket_ipv4_multicast,
  TestRingServer#test_make_socket_ipv6_multicast): change bound
  interface address because multicast address is not allowed on Linux
  or Windows.
  [ruby-core:53692] [Bug #8159]

Revision 40472 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array
  arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional
  arguments for multicast interface.
- test/rinda/test_rinda.rb
  (TestRingFinger#test_ring_server_ipv4_multicast,
  TestRingFinger#test_ring_server_ipv6_multicast): add tests for
  above change.
- test/rinda/test_rinda.rb
  (TestRingServer#test_make_socket_ipv4_multicast,
  TestRingServer#test_make_socket_ipv6_multicast): change bound
  interface address because multicast address is not allowed on Linux
  or Windows.
  [ruby-core:53692] [Bug #8159]

Revision 40472 - 04/25/2013 03:43 PM - shirosaki

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array
  arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional
  arguments for multicast interface.
- test/rinda/test_rinda.rb
  (TestRingFinger#test_ring_server_ipv4_multicast,
  TestRingFinger#test_ring_server_ipv6_multicast): add tests for
  above change.
- test/rinda/test_rinda.rb
  (TestRingServer#test_make_socket_ipv4_multicast,
  TestRingServer#test_make_socket_ipv6_multicast): change bound
  interface address because multicast address is not allowed on Linux
  or Windows.
  [ruby-core:53692] [Bug #8159]

History

#1 - 03/24/2013 10:57 PM - luislavena (Luis Lavena)

- Status changed from Open to Assigned

#2 - 03/25/2013 07:02 AM - drbrain (Eric Hodel)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r39922.
Luis, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- test/rinda/test_rinda.rb: Skip IPv6 tests if no IPv6 addresses exist. Skip fork-dependent test if fork is not available. [ruby-trunk - Bug #8159]

#3 - 03/25/2013 07:04 AM - drbrain (Eric Hodel)

- Status changed from Closed to Assigned

- % Done changed from 100 to 50

I have skipped the test using fork and the IPv6 tests when IPv6 addresses are missing.

I'm unsure what to do about the IPv4 multicast test, does something special need to be done for multicast on windows?

#4 - 03/25/2013 06:10 PM - naruse (Yui NARUSE)

I create an experimental patch as following for failures on Linux:
http://u64.rubyci.org/~chkbuild/ruby-trunk/log/20130324T210202Z.log.html.gz

```
diff --git a/ext/socket/socket.c b/ext/socket/socket.c
index 1ddaea9..44703c1 100644
--- a/ext/socket/socket.c
+++ b/ext/socket/socket.c
@@ -1585,7 +1585,9 @@ socket_s_ip_address_list(VALUE self)
     for (p = ifp; p; p = p->ifa_next) {
         if (p->ifa_addr != NULL && IS_IP_FAMILY(p->ifa_addr->sa_family)) {
             struct sockaddr *addr = p->ifa_addr;
-            rb_ary_push(list, sockaddr_obj(addr, sockaddr_len(addr)));
+            ai = sockaddr_obj(addr, sockaddr_len(addr));
+            rb_ivar_set(ai, rb_intern("@ifa_name"), rb_str_new_cstr(p->ifa_name));
+            rb_ary_push(list, ai);
         }
     }

@@ -1856,6 +1858,12 @@ socket_s_ip_address_list(VALUE self)
 #define socket_s_ip_address_list rb_f_notimplement
 #endif

+VALUE
+sock_s_if_nametoindex(VALUE klass, VALUE name)
+{
+    return UINT2NUM(if_nametoindex(StringValueCStr(name)));
+}
+
 void
 Init_socket()
 {
@@ -1999,6 +2007,7 @@ Init_socket()
     rb_define_singleton_method(rb_cSocket, "gethostbyaddr", sock_s_gethostbyaddr, -1);
     rb_define_singleton_method(rb_cSocket, "getservbyname", sock_s_getservbyname, -1);
     rb_define_singleton_method(rb_cSocket, "getservbyport", sock_s_getservbyport, -1);
+    rb_define_singleton_method(rb_cSocket, "if_nametoindex", sock_s_if_nametoindex, 1);
     rb_define_singleton_method(rb_cSocket, "getaddrinfo", sock_s_getaddrinfo, -1);
     rb_define_singleton_method(rb_cSocket, "getnameinfo", sock_s_getnameinfo, -1);
     rb_define_singleton_method(rb_cSocket, "sockaddr_in", sock_s_pack_sockaddr_in, 2);
diff --git a/lib/rinda/ring.rb b/lib/rinda/ring.rb
index 9b3e273..b87800c 100644
--- a/lib/rinda/ring.rb
+++ b/lib/rinda/ring.rb
@@ -59,6 +59,8 @@ module Rinda
       end
     end

+    attr_accessor :multicast_interface
+
     ##
     # Advertises +ts+ on the given +addresses+ at +port+.
     #
@@ -84,6 +86,7 @@ module Rinda

       @w_services = write_services
       @r_service  = reply_service
+      @multicast_interface = 0
     end

     ##
@@ -111,6 +114,9 @@ module Rinda
           mreq = IPAddr.new(addrinfo.ip_address).hton + [0].pack('I')
```

```
            socket.setsockopt(:IPPROTO_IPV6, :IPV6_JOIN_GROUP, mreq)
+           sa = addrinfo.to_sockaddr
+           sa[-4, 4] = [@multicast_interface].pack('I')
+           addrinfo = Addrinfo.new(sa)
          end
        end

diff --git a/test/rinda/test_rinda.rb b/test/rinda/test_rinda.rb
index 65df228..9aaeb66 100644
--- a/test/rinda/test_rinda.rb
+++ b/test/rinda/test_rinda.rb
@@ -525,7 +525,24 @@ class TupleSpaceProxyTest < Test::Unit::TestCase
   @server = DRb.primary_server || DRb.start_service
 end

+module RingIPv6
+  def prepare_ipv6(r)
+    Socket.ip_address_list.any? do |addrinfo|
+      if addrinfo.ipv6? && ipv6_global_unicast?(addrinfo)
+        r.multicast_interface = Socket.if_nametoindex(addrinfo.instance_variable_get(:@ifa_name))
+        return
+      end
+    end
+    skip 'IPv6 not available'
+  end
+
+  def ipv6_global_unicast?(ai)
+    (ai.ip_address[0].to_i & 0b1110) == 0b0010
+  end
+end
+
 class TestRingServer < Test::Unit::TestCase
+  include RingIPv6

   def setup
     @port = Rinda::Ring_PORT
@@ -560,14 +577,8 @@ class TestRingServer < Test::Unit::TestCase
   end

   def test_make_socket_ipv6_multicast
-    skip 'IPv6 not available' unless
-      Socket.ip_address_list.any? { |addrinfo| addrinfo.ipv6? }
-
-    begin
-      v6mc = @rs.make_socket('ff02::1')
-    rescue Errno::EADDRNOTAVAIL
-      return # IPv6 address for multicast not available
-    end
+    prepare_ipv6(@rs)
+    v6mc = @rs.make_socket('ff02::1')

     if Socket.const_defined?(:SO_REUSEPORT) then
       assert v6mc.getsockopt(:SOCKET, :SO_REUSEPORT).bool
@@ -575,7 +586,7 @@ class TestRingServer < Test::Unit::TestCase
       assert v6mc.getsockopt(:SOCKET, :SO_REUSEADDR).bool
     end

-    assert_equal('ff02::1',  v6mc.local_address.ip_address)
+    assert_match(/\Aff02::1(?:%|\z)/,  v6mc.local_address.ip_address)
     assert_equal(@port, v6mc.local_address.ip_port)
   end

@@ -588,22 +599,10 @@ class TestRingServer < Test::Unit::TestCase
 end

 class TestRingFinger < Test::Unit::TestCase
+  include RingIPv6

   def setup
     @rf = Rinda::RingFinger.new
-    ifindex = nil
-    10.times do |i|
-      begin
-        addrinfo = Addrinfo.udp('ff02::1', Rinda::Ring_PORT)
-        soc = Socket.new(addrinfo.pfamily, addrinfo.socktype, addrinfo.protocol)
```

```
-        soc.setsockopt(:IPPROTO_IPV6, :IPV6_MULTICAST_IF,
-                       [i].pack('I'))
-        ifindex = i
-        break
-      rescue
-      end
-    end
-    @rf.multicast_interface = ifindex
   end

   def test_make_socket_unicast
@@ -620,26 +619,23 @@ class TestRingFinger < Test::Unit::TestCase
   end

   def test_make_socket_ipv6_multicast
-    skip 'IPv6 not available' unless
-      Socket.ip_address_list.any? { |addrinfo| addrinfo.ipv6? }
-
+    prepare_ipv6(@rf)
     v6mc = @rf.make_socket('ff02::1')

     assert_equal(1, v6mc.getsockopt(:IPPROTO_IPV6, :IPV6_MULTICAST_LOOP).int)
     assert_equal(1, v6mc.getsockopt(:IPPROTO_IPV6, :IPV6_MULTICAST_HOPS).int)
   end

-  def test_make_socket_multicast_hops
+  def test_make_socket_ipv4_multicast_hops
     @rf.multicast_hops = 2

     v4mc = @rf.make_socket('239.0.0.1')

     assert_equal(2, v4mc.getsockopt(:IPPROTO_IP, :IP_MULTICAST_TTL).int)
+  end

-    return unless Socket.ip_address_list.any? { |addrinfo| addrinfo.ipv6? }
-
+  def test_make_socket_ipv6_multicast_hops
+    prepare_ipv6(@rf)
+    @rf.multicast_hops = 2
     v6mc = @rf.make_socket('ff02::1')

     assert_equal(2, v6mc.getsockopt(:IPPROTO_IPV6, :IPV6_MULTICAST_HOPS).int)
   end
```

#5 - 03/25/2013 07:23 PM - akr (Akira Tanaka)

2013/3/25 naruse (Yui NARUSE) naruse@airemix.jp:

> Issue #8159 has been updated by naruse (Yui NARUSE).
>
> I create an experimental patch as following for failures on Linux:
> http://u64.rubyci.org/~chkbuild/ruby-trunk/log/20130324T210202Z.log.html.gz
>
> diff --git a/ext/socket/socket.c b/ext/socket/socket.c
> index 1ddaea9..44703c1 100644
> --- a/ext/socket/socket.c
> +++ b/ext/socket/socket.c
> @@ -1585,7 +1585,9 @@ socket_s_ip_address_list(VALUE self)
> for (p = ifp; p; p = p->ifa_next) {
> if (p->ifa_addr != NULL && IS_IP_FAMILY(p->ifa_addr->sa_family)) {
> struct sockaddr *addr = p->ifa_addr;
>
> - rb_ary_push(list, sockaddr_obj(addr, sockaddr_len(addr)));
> - ai = sockaddr_obj(addr, sockaddr_len(addr));
> - rb_ivar_set(ai, rb_intern("@ifa_name"), rb_str_new_cstr(p->ifa_name));
> - rb_ary_push(list, ai); } }

It seems we need Socket.getifaddrs.
--
Tanaka Akira

#6 - 03/25/2013 07:59 PM - naruse (Yui NARUSE)

akr (Akira Tanaka) wrote:

2013/3/25 naruse (Yui NARUSE) naruse@airemix.jp:

> Issue #8159 has been updated by naruse (Yui NARUSE).
>
> I create an experimental patch as following for failures on Linux:
> http://u64.rubyci.org/~chkbuild/ruby-trunk/log/20130324T210202Z.log.html.gz
>
> diff --git a/ext/socket/socket.c b/ext/socket/socket.c
> index 1ddaea9..44703c1 100644
> --- a/ext/socket/socket.c
> +++ b/ext/socket/socket.c
> @@ -1585,7 +1585,9 @@ socket_s_ip_address_list(VALUE self)
> for (p = ifp; p; p = p->ifa_next) {
> if (p->ifa_addr != NULL && IS_IP_FAMILY(p->ifa_addr->sa_family)) {
> struct sockaddr *addr = p->ifa_addr;
>
> - rb_ary_push(list, sockaddr_obj(addr, sockaddr_len(addr)));
> - ai = sockaddr_obj(addr, sockaddr_len(addr));
> - rb_ivar_set(ai, rb_intern("@ifa_name"), rb_str_new_cstr(p->ifa_name));
> - rb_ary_push(list, ai); } }

It seems we need Socket.getifaddrs.

Sure, but what is the type of its return value?
Extended Addrinfo or a new object like Ifaddr?

#7 - 03/26/2013 01:23 AM - akr (Akira Tanaka)

2013/3/25 naruse (Yui NARUSE) naruse@airemix.jp:

> It seems we need Socket.getifaddrs.

Sure, but what is the type of its return value?
Extended Addrinfo or a new object like Ifaddr?

I think it should have new class.

Addrinfo has bunch of methods for a struct sockaddr.
But struct ifaddrs has multiple pointers to struct sockaddr.
It is difficult to deal with them fairly.
--
Tanaka Akira

#8 - 03/26/2013 01:44 AM - drbrain (Eric Hodel)

I want to update #8075 to use getifaddrs so it will contain at least the interface name, IP address and netmask for the interface. With this information we can fix this bug and have RFC-compliant one-shot mDNS support in Resolv (as you only want to accept packets from the local network, see #8089).

I propose the class Socket::Interface for this data along with Socket.interface_list.

I have not had the time to update my patch in #8075, though, as retreiving the netmask seems difficult on windows.

#9 - 03/26/2013 03:59 AM - naruse (Yui NARUSE)

After some inspection, I came to doubt RingServer#make_socket is buggy.

Actually at least on FreeBSD and darwin test_make_socket_ipv4_multicast and test_make_socket_ipv6_multicast don't fail,
but on Linux and Windows.
It seems because the given address for bind is ff02::1.
I think it must be the client's ip address.
I don't know the detail of multicast and cannot fix it correctly.
So could you confirm them?

#10 - 03/26/2013 08:23 AM - akr (Akira Tanaka)

2013/3/26 drbrain (Eric Hodel) drbrain@segment7.net:

> Issue #8159 has been updated by drbrain (Eric Hodel).
>
> I want to update #8075 to use getifaddrs so it will contain at least the interface name, IP address and netmask for the interface. With this

> information we can fix this bug and have RFC-compliant one-shot mDNS support in Resolv (as you only want to accept packets from the local network, see #8089).
>
> I propose the class Socket::Interface for this data along with Socket.interface_list.

I'm not sure it is good idea.

The result of getifaddrs() has no one-to-one mapping to the result of
if_nameindex().

Also, if_nameindex() is standardized by POSIX (and RFC 3493) but getifaddrs()
is not standardized.
So it is good to separate the features to be if_nameindex() is usable even on
a platform which doesn't support getifaddrs().
--
Tanaka Akira

#11 - 03/27/2013 10:53 AM - akr (Akira Tanaka)

2013/3/26 Tanaka Akira akr@fsij.org:

> > Sure, but what is the type of its return value?
> > Extended Addrinfo or a new object like Ifaddr?
>
> I think it should have new class.
>
> Addrinfo has bunch of methods for a struct sockaddr.
> But struct ifaddrs has multiple pointers to struct sockaddr.
> It is difficult to deal with them fairly.

One more point for new class:

Addrinfo is possible to marshal now but
struct ifaddrs is impossible to marshal
because we don't know about internal of ifa_data.
--
Tanaka Akira

#12 - 03/29/2013 07:23 PM - akr (Akira Tanaka)

2013/3/25 Tanaka Akira akr@fsij.org

> It seems we need Socket.getifaddrs.

I implemented it in
https://github.com/akr/ruby/tree/getifaddrs
--
Tanaka Akira

#13 - 04/01/2013 05:22 PM - naruse (Yui NARUSE)

On Linux, IPv6 UDP multicast with link-local IP address (FF02::1 is link-local) requires scope_id specified.
http://d.hatena.ne.jp/torutk/20080520/p1
http://stackoverflow.com/questions/3851061/listening-for-ipv6-multicasts-on-linux
http://stackoverflow.com/questions/3068231/ipv6-link-local-multicasting

So at first you must decide the meaning of Ringserver.new "ff02::1" on Linux.
If it means only one interface, the interface must be specified.
If it means all interfaces, make_socket must get interface by itself, and make related sockets for all interfaces.

#14 - 04/02/2013 06:02 AM - drbrain (Eric Hodel)

It is odd, considering IPV6_MULTICAST_IF, but I will make these updates.

Thank you for researching this for me while I have been busy.

#15 - 04/06/2013 06:15 PM - h.shirosaki (Hiroshi Shirosaki)

- File 0001-Fix-multicast-of-rinda.patch added

I created a patch to fix test errors.

It seems a socket should be binded to another non multicast address and join a multicast group.

I got it from the following examples.
http://ntrg.cs.tcd.ie/undergrad/4ba2/multicast/antony/example.html
http://publib.boulder.ibm.com/infocenter/iseries/v5r3/index.jsp?topic=%2Frzab6%2Frzab6xmulticast.htm

#16 - 04/06/2013 06:34 PM - naruse (Yui NARUSE)

h.shirosaki (Hiroshi Shirosaki) wrote:

> I created a patch to fix test errors.
> It seems a socket should be binded to another non multicast address and join a multicast group.
>
> I got it from the following examples.
> http://ntrg.cs.tcd.ie/undergrad/4ba2/multicast/antony/example.html
> http://publib.boulder.ibm.com/infocenter/iseries/v5r3/index.jsp?topic=%2Frzab6%2Frzab6xmulticast.htm

Yeah, I think it is correct.
But your patch doesn't specify what multicast group the socket join.
Current make_socket's argument is only one and it specifies both binding address and joining multicast group,
so I think the API must be modified.

#17 - 04/09/2013 06:26 PM - h.shirosaki (Hiroshi Shirosaki)

- File 0002-Fix-multicast-of-rinda.patch added

I've updated a patch which includes API change. Users can specify network interface by optional arguments. But I'm not sure about API spec.
If #8075 is applied, we will be able to specify an interface for IPv6 by name.

#18 - 04/18/2013 05:49 AM - naruse (Yui NARUSE)

Could you temporally commit the patch because it causes test failure?

#19 - 04/26/2013 12:43 AM - Anonymous

- Status changed from Assigned to Closed

- % Done changed from 50 to 100

This issue was solved with changeset r40472.
Luis, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

---

ring.rb: specify multicast interface

- lib/rinda/ring.rb (Rinda::RingServer#initialize): accept array arguments of address to specify multicast interface.
- lib/rinda/ring.rb (Rinda::RingServer#make_socket): add optional arguments for multicast interface.
- test/rinda/test_rinda.rb (TestRingFinger#test_ring_server_ipv4_multicast, TestRingFinger#test_ring_server_ipv6_multicast): add tests for above change.
- test/rinda/test_rinda.rb (TestRingServer#test_make_socket_ipv4_multicast, TestRingServer#test_make_socket_ipv6_multicast): change bound interface address because multicast address is not allowed on Linux or Windows.
  [ruby-core:53692] [Bug #8159]

#20 - 04/26/2013 12:50 AM - h.shirosaki (Hiroshi Shirosaki)

Committed at r40472 to fix test failure. If there are problems, please fix it.

#21 - 04/26/2013 04:17 PM - naruse (Yui NARUSE)

- Status changed from Closed to Assigned

- Target version changed from 2.6 to 2.1.0

#22 - 05/20/2013 03:40 AM - luislavena (Luis Lavena)

- Assignee changed from drbrain (Eric Hodel) to usa (Usaku NAKAMURA)

- % Done changed from 100 to 80

Rinda failure now seems to be related to ifindex() being missing:

http://ci.rubyinstaller.org/job/ruby-trunk-x86-test-all/lastFailedBuild/console

#23 - 08/04/2013 05:25 AM - luislavena (Luis Lavena)

- Status changed from Assigned to Closed

issue shown here no longer present in CI.

Thank you.

Files

0001-Fix-multicast-of-rinda.patch	2.37 KB	04/06/2013	h.shirosaki (Hiroshi Shirosaki)
0002-Fix-multicast-of-rinda.patch	6.13 KB	04/09/2013	h.shirosaki (Hiroshi Shirosaki)