Ruby master - Feature #14922

Resolv getaddresses ignores AAAA records for IPv6

07/19/2018 07:09 PM - dlampa (Donovan Lampa)

Status:	Assigned
Priority:	Normal
Assignee:	akr (Akira Tanaka)
Target version:	

Description

I'd like some feedback here as I'm not totally convinced this is a bug quite yet. I may have done something silly with my DNS configuration.

I have a local DNS server set up with the following /etc/resolv.conf

```
[root@ip-10-20-0-181 ~]# cat /etc/resolv.conf
# Generated by NetworkManager
search us-east-2.compute.internal
nameserver ::1
```

And the following zone configured in bind

```
[root@ip-10-20-0-181 ~]# cat /var/named/test.net.zone
$TTL    86400

@       IN      SOA     test.net root.test.net (
        2016050204
        3600
        900
        604800
        86400
)

@               IN      NS          testbox
testbox         IN      AAAA        2600:1f16:a82:9b01:5a89:f06f:dde4:7b5e
malware         IN      AAAA        2600:1f16:a82:9b01:cab2:a7c0:e2cb:1162
goodware        IN      A           10.20.0.181
```

It appears that Resolv needs me to explicitly request the AAAA resource for malware.test.net in order to get the IPv6 address in this situation. It doesn't seem to need that for IPv4.

```
irb(main):001:0> require 'resolv'
=> true
irb(main):002:0> resolver = Resolv.new
=> #<Resolv:0x0000000001db3c78 @resolvers=[#<Resolv::Hosts:0x0000000001db3c28 @filename="/etc/host
s", @mutex=#<Thread::Mutex:0x0000000001db3bd8>, @initialized=nil>, #<Resolv::DNS:0x0000000001db3b8
8 @mutex=#<Thread::Mutex:0x0000000001db3b38>, @config=#<Resolv::DNS::Config:0x0000000001db3ae8 @mu
tex=#<Thread::Mutex:0x0000000001db3a48>, @config_info=nil, @initialized=nil, @timeouts=nil>, @init
ialized=nil>]>
irb(main):003:0> resolver.getaddresses("malware.test.net")
=> []
irb(main):004:0> resolver.getaddresses("goodware.test.net")
=> ["10.20.0.181"]
irb(main):005:0> dns_resolver = Resolv::DNS.new
=> #<Resolv::DNS:0x0000000001d2a040 @mutex=#<Thread::Mutex:0x0000000001d29fa0>, @config=#<Resolv::
DNS::Config:0x0000000001d29f78 @mutex=#<Thread::Mutex:0x0000000001d29e60>, @config_info=nil, @init
ialized=nil, @timeouts=nil>, @initialized=nil>
irb(main):006:0> dns_resolver.getresource("malware.test.net", Resolv::DNS::Resource::IN::AAAA)
=> #<Resolv::DNS::Resource::IN::AAAA:0x0000000001cbaf38 @address=#<Resolv::IPv6 2600:1F16:A82:9B01
:CAB2:A7C0:E2CB:1162>, @ttl=86400>
```

Based on Resolv's documentation I would expect to get back the IPv6 address for malware.test.net

History

#1 - 07/24/2018 07:22 PM - eviljoel (evil joel)

Greetings. I work behind Donovan (this bug's reporter) and am also dealing with this same issue. I tracked down the source of this behavior by looking through the Ruby resolv.rb source code. Apparently the DefaultResolver only resolves IPv6 addresses if the interpreter's machine has been assigned a public IPv6 address. Machines configured with loopback or link-local IPv6 addresses will default to not resolving IPv6 addresses. To see the code in question, search for use_ipv6 in https://github.com/ruby/ruby/blob/trunk/lib/resolv.rb .

This behavior is causing us problems testing on an IPv6 test network using link-local addresses. Our test network also has a test DNS server which resolves hostnames to link-local IPv6 addresses. While this is only a test network, a similar network configuration could potentially show up at one of our client sites and we would like to be able to resolve IPv6 addresses without writing IPv6 specific resolution code. Also, other Ruby users might setup similar IPv6 test networks and encounter this inconsistent behavior.

I understand why this IPv6 resolution behavior is desirable for the vast majority of Ruby users. However, there should be a way to override this behavior for those who desire full IPv6 functionality equivalent to IPv4 functionality. I recommend that the default resolution behavior be overridden in two ways:

1) Add a RUBYOPT option named --assume-ipv6 to force full IPv6 support.
2) Add a config_info parameter to the Resolv constructor. Have it take a boolean option named :assumeipv6.

Also, please add documentation to https://ruby-doc.org/stdlib-2.5.1/libdoc/resolv/rdoc/Resolv.html explaining when IPv6 addresses are resolved by default. I could not find this behavior described anywhere!

Thank you.

#2 - 07/25/2018 07:44 AM - shyouhei (Shyouhei Urabe)

- Assignee set to akr (Akira Tanaka)

- Status changed from Open to Assigned

Mmm, I think this is a real bug.

RUBYOPT does not work well in this case because resolve.rb is a library while RUBYOPT is for the process. Adding parameter to the constructor is an option though.

#3 - 08/27/2019 03:38 AM - jeremyevans0 (Jeremy Evans)

- Backport deleted (2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN)

- ruby -v deleted (2.5.1)

- Tracker changed from Bug to Feature

- File resolv-use_ipv6-14922.patch added

I think this is not a bug, but a feature request for IPv6 resolution when no public IPv6 address is present. I think you can only consider this a bug if you think the default behavior should be to return IPv6 addresses in such cases.

I think this is a worthwhile feature to add. Attached is a patch that implements it using a keyword argument to Resolv#initialize:

```
$ ruby -rresolv -e 'p Resolv.new.getaddresses("google.com")'
["216.58.195.78"]
$ ruby -rresolv -e 'p Resolv.new(:use_ipv6=>true).getaddresses("google.com")'
["216.58.195.78", "2607:f8b0:4005:807::200e"]
```

Files

resolv-use_ipv6-14922.patch	2.08 KB	08/27/2019	jeremyevans0 (Jeremy Evans)