This repository | Search Pull requests Issues Marketplace Gist

golang / go Watch 2,265 Star 29,233 Fork 3,935

Code Issues 2,817 Pull requests 0 Wiki Insights

net: Go's DNS resolver fails with no such host #15434

New issue

Closed **fraenkel** opened this issue on Apr 26, 2016 · 9 comments

fraenkel commented on Apr 26, 2016 Contributor

Please answer these questions before submitting your issue. Thanks!

1. What version of Go are you using (`go version`)?
   go 1.6.2
   go version devel +093ac15 Mon Apr 25 06:00:15 2016 +0000 linux/amd64
2. What operating system and processor architecture are you using (`go env`)?
   linux amd64
3. What did you do?
   If possible, provide a recipe for reproducing the error.
   A complete runnable program is good.
   A link on play.golang.org is best.

http://play.golang.org/p/O1SMk6JM1w

1. What did you expect to see?
   success
2. What did you see instead?
   Get http://registry-1.docker.io/v2: dial tcp: lookup registry-1.docker.io on 10.155.248.190:53: no such host

When I set export GODEBUG=netdns=cgo+1, I do succeed but with the default or setting it to go, I always get the no such host.
I have 3 DNS entries in my resolv.conf. The second and third entries can resolve the name.
With strace, I see that the cgo trace shows a request to each DNS resolver, however with Go, it only tries the first and fails.

mdempsky commented on Apr 26, 2016 Member

Can you share the output from running `dig registry-1.docker.io @10.155.248.190` on that machine?

fraenkel commented on Apr 26, 2016 Contributor

root@6515733c-cf49-4443-b7d9-adb073ea8885:~# dig registry-1.docker.io @10.155.248.190

; <<>> DiG 9.9.5-3ubuntu0.8-Ubuntu <<>> registry-1.docker.io @10.155.248.190
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6235
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 2800
;; QUESTION SECTION:
;registry-1.docker.io. IN A

;; Query time: 1 msec
;; SERVER: 10.155.248.190#53(10.155.248.190)
;; WHEN: Mon Apr 25 17:05:55 UTC 2016
;; MSG SIZE rcvd: 49

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone
Go1.7

Notifications
Subscribe
You're not receiving notifications from this thread.

4 participants

mdempsky commented on Apr 26, 2016 Member +😀

Interesting. With strace, can you tell if libc immediately tries the next nameserver, or does it continue waiting a bit longer before sending the next packet (e.g., calling poll(2) again)?

Looking at libresolv's source, I'm wondering if you're hitting this code:

```
if (anhp->rcode == NOERROR && anhp->ancount == 0
    && anhp->aa == 0 && anhp->ra == 0 && anhp->arcount == 0) {
        DprintQ(statp->options & RES_DEBUG,
                (stdout, "referred query:\n"),
                *thisansp,
                (*thisresplenp > *thisanssizp)
                ? *thisanssizp : *thisresplenp);
        goto next_ns;
}
```

except that according to the dig output, anhp->arcount (aka "ADDITIONAL") should be 1.

fraenkel commented on Apr 26, 2016 Contributor +😀

Here is the relevant strace from both

go DNS

```
socket(PF_INET, SOCK_DGRAM|SOCK_CLOEXEC|SOCK_NONBLOCK, IPPROTO_IP) = 5
setsockopt(5, SOL_SOCKET, SO_BROADCAST, [1], 4) = 0
connect(5, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("10.155.248.190")}, 1
epoll_ctl(4, EPOLL_CTL_ADD, 5, {EPOLLIN|EPOLLOUT|EPOLLRDHUP|EPOLLET, {u32=2021793344, u64=1
getsockname(5, {sa_family=AF_INET, sin_port=htons(50722), sin_addr=inet_addr("10.155.248.16
getpeername(5, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("10.155.248.190")
clock_gettime(CLOCK_REALTIME, {1461610393, 390514267}) = 0
clock_gettime(CLOCK_MONOTONIC, {5648, 84800638}) = 0
clock_gettime(CLOCK_MONOTONIC, {5648, 84874068}) = 0
clock_gettime(CLOCK_REALTIME, {1461610393, 390734946}) = 0
write(5, "\310-\1\0\0\1\0\0\0\0\0\0\nregistry-1\6docker\2i"..., 38) = 38
read(5, 0xc8200ee000, 512)              = -1 EAGAIN (Resource temporarily unavailable)
futex(0x704bd0, FUTEX_WAIT, 0, NULLGet http://registry-1.docker.io/v2: dial tcp: lookup reg
```

cgo DNS

```
socket(PF_INET, SOCK_DGRAM|SOCK_NONBLOCK, IPPROTO_IP) = 3
connect(3, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("10.155.248.190")}, 1
gettimeofday({1461610383, 13246}, NULL) = 0
poll([{fd=3, events=POLLOUT}], 1, 0)    = 1 ([{fd=3, revents=POLLOUT}])
sendmmsg(3, {{{msg_name(0)=NULL, msg_iov(1)=[{"\324^\1\0\0\1\0\0\0\0\0\0\nregistry-1\6docke
poll([{fd=3, events=POLLIN}], 1, 5000)  = 1 ([{fd=3, revents=POLLIN}])
ioctl(3, FIONREAD, [38])                = 0
recvfrom(3, "\364\244\201\0\0\1\0\0\0\0\0\0\nregistry-1\6docker\2i"..., 2048, 0, {sa_family
close(3)                                = 0
socket(PF_INET, SOCK_DGRAM|SOCK_NONBLOCK, IPPROTO_IP) = 3
connect(3, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("10.0.80.11")}, 16) =
gettimeofday({1461610383, 14478}, NULL) = 0
poll([{fd=3, events=POLLOUT}], 1, 0)    = 1 ([{fd=3, revents=POLLOUT}])
sendmmsg(3, {{{msg_name(0)=NULL, msg_iov(1)=[{"\324^\1\0\0\1\0\0\0\0\0\0\nregistry-1\6docke
poll([{fd=3, events=POLLIN}], 1, 3000)  = 1 ([{fd=3, revents=POLLIN}])
ioctl(3, FIONREAD, [125])               = 0
recvfrom(3, "\324^\201\200\0\1\0\4\0\0\0\0\nregistry-1\6docker\2i"..., 2048, 0, {sa_family=
gettimeofday({1461610383, 15315}, NULL) = 0
poll([{fd=3, events=POLLIN}], 1, 2999)  = 1 ([{fd=3, revents=POLLIN}])
ioctl(3, FIONREAD, [152])               = 0
recvfrom(3, "\364\244\201\200\0\1\0\1\0\1\0\0\nregistry-1\6docker\2i"..., 65536, 0, {sa_fam
close(3)                                = 0
```

mdempsky commented on Apr 26, 2016 Member +😀

Thanks! Looking at the DNS response header from 10.155.248.190 in the cgo strace output, it's clear that the response *does* actually have `arcount==0` , which explains why libresolv proceeds to try the next name server.

Replicating libresolv's behavior here seems easy, but feels kinda gross as it doesn't seem to be prescribed by the RFCs at all.

FYI though, your netgo strace output looks incomplete though, like you're only tracing one of the OS threads. In particular, it's missing the `write` call that actually produced the "no such host" output, and it's probably missing the `recvmsg` call that actually read the response DNS packet; the one in the output above failed with `EAGAIN`.

I usually use `strace -f` to trace all children.

fraenkel commented on Apr 26, 2016 Contributor

I have a more more complete go trace with some editorials to trim the unwanted stuff.
I am noticing that we connect to 10.155.248.190 twice. Not sure if that is expected.

```
[pid  7617] socket(PF_INET, SOCK_DGRAM|SOCK_CLOEXEC|SOCK_NONBLOCK, IPPROTO_IP <unfinished .
[pid  7621] futex(0x703e20, FUTEX_WAIT, 2, NULL <unfinished ...>
[pid  7617] <... socket resumed> )      = 3
[pid  7617] setsockopt(3, SOL_SOCKET, SO_BROADCAST, [1], 4 <unfinished ...>
[pid  7617] <... setsockopt resumed> )  = 0
[pid  7619] futex(0x703e20, FUTEX_WAKE, 1 <unfinished ...>
[pid  7617] connect(3, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("10.155.2
[pid  7621] <... futex resumed> )       = 0
[pid  7617] <... connect resumed> )     = 0
[pid  7619] <... futex resumed> )       = 1
[pid  7617] epoll_create1(EPOLL_CLOEXEC <unfinished ...>
[pid  7621] futex(0x703e20, FUTEX_WAKE, 1 <unfinished ...>
[pid  7617] <... epoll_create1 resumed> ) = 4
[pid  7621] <... futex resumed> )       = 0
[pid  7617] epoll_ctl(4, EPOLL_CTL_ADD, 3, {EPOLLIN|EPOLLOUT|EPOLLRDHUP|EPOLLET, {u32=13877
[pid  7617] <... epoll_ctl resumed> )   = 0
[pid  7621] clock_gettime(CLOCK_REALTIME,  <unfinished ...>
[pid  7617] getsockname(3,  <unfinished ...>
[pid  7617] <... getsockname resumed> {sa_family=AF_INET, sin_port=htons(59996), sin_addr=i
[pid  7617] getpeername(3,  <unfinished ...>
[pid  7617] <... getpeername resumed> {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet
[pid  7621] socket(PF_INET, SOCK_DGRAM|SOCK_CLOEXEC|SOCK_NONBLOCK, IPPROTO_IP <unfinished .
[pid  7621] <... socket resumed> )      = 5
[pid  7621] setsockopt(5, SOL_SOCKET, SO_BROADCAST, [1], 4 <unfinished ...>
[pid  7621] <... setsockopt resumed> )  = 0
[pid  7621] connect(5, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("10.155.2
[pid  7621] <... connect resumed> )     = 0
[pid  7617] futex(0x703e38, FUTEX_WAKE, 1 <unfinished ...>
[pid  7621] epoll_ctl(4, EPOLL_CTL_ADD, 5, {EPOLLIN|EPOLLOUT|EPOLLRDHUP|EPOLLET, {u32=13877
[pid  7617] <... futex resumed> )       = 0
[pid  7621] <... epoll_ctl resumed> )   = 0
[pid  7621] getsockname(5,  <unfinished ...>
[pid  7621] <... getsockname resumed> {sa_family=AF_INET, sin_port=htons(43899), sin_addr=i
[pid  7617] write(3, "f\273\1\0\0\1\0\0\0\0\0\0\nregistry-1\6docker\2i"..., 38 <unfinished
[pid  7621] getpeername(5,  <unfinished ...>
[pid  7619] futex(0x703e38, FUTEX_WAIT, 0, {4, 999648197} <unfinished ...>
[pid  7617] <... write resumed> )       = 38
[pid  7621] <... getpeername resumed> {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet
[pid  7619] <... futex resumed> )       = -1 EAGAIN (Resource temporarily unavailable)
[pid  7617] read(3,  <unfinished ...>
[pid  7617] <... read resumed> 0xc8200e8000, 512) = -1 EAGAIN (Resource temporarily unavail
[pid  7617] epoll_wait(4,  <unfinished ...>
[pid  7617] <... epoll_wait resumed> {{EPOLLOUT, {u32=1387773736, u64=139927127310120}}, {E
[pid  7617] epoll_wait(4,  <unfinished ...>
[pid  7621] sched_yield( <unfinished ...>
[pid  7621] <... sched_yield resumed> ) = 0
[pid  7621] futex(0x703e20, FUTEX_WAKE, 1 <unfinished ...>
[pid  7619] futex(0x703e38, FUTEX_WAIT, 0, {4, 998688799} <unfinished ...>
[pid  7621] <... futex resumed> )       = 0
[pid  7618] <... select resumed> )      = 0 (Timeout)
[pid  7621] write(5, "4\274\1\0\0\1\0\0\0\0\0\0\nregistry-1\6docker\2i"..., 38 <unfinished
[pid  7621] <... write resumed> )       = 38
[pid  7617] <... epoll_wait resumed> {{EPOLLOUT, {u32=1387773544, u64=139927127309928}}}, 1
[pid  7621] read(5,  <unfinished ...>
[pid  7617] epoll_wait(4,  <unfinished ...>
[pid  7618] select(0, NULL, NULL, NULL, {0, 20} <unfinished ...>
[pid  7617] <... epoll_wait resumed> {{EPOLLIN|EPOLLOUT, {u32=1387773736, u64=1399271273101
[pid  7621] <... read resumed> 0xc8200ee000, 512) = -1 EAGAIN (Resource temporarily unavail
[pid  7621] futex(0xc82005a110, FUTEX_WAIT, 0, NULL <unfinished ...>
[pid  7617] read(3, "f\273\201\0\0\1\0\0\0\0\0\0\nregistry-1\6docker\2i"..., 512) = 38
[pid  7617] epoll_ctl(4, EPOLL_CTL_DEL, 3, {0, {u32=0, u64=0}} <unfinished ...>
[pid  7618] <... select resumed> )      = 0 (Timeout)
[pid  7617] <... epoll_ctl resumed> )   = 0
[pid  7617] close(3 <unfinished ...>
[pid  7617] <... close resumed> )       = 0
[pid  7617] futex(0xc82005a110, FUTEX_WAKE, 1 <unfinished ...>
[pid  7617] <... futex resumed> )       = 1
[pid  7621] <... futex resumed> )       = 0
[pid  7617] epoll_wait(4,  <unfinished ...>
[pid  7621] epoll_wait(4,  <unfinished ...>
```

```
[pid  7617] <... epoll_wait resumed> {{EPOLLIN|EPOLLOUT, {u32=1387773544, u64=1399271273099
[pid  7621] <... epoll_wait resumed> {}, 128, 0) = 0
[pid  7617] read(5, "4\274\201\0\0\1\0\0\0\0\0\0\nregistry-1\6docker\2i"..., 512) = 38
[pid  7621] epoll_wait(4,  <unfinished ...>
[pid  7618] select(0, NULL, NULL, NULL, {0, 20} <unfinished ...>
[pid  7617] epoll_ctl(4, EPOLL_CTL_DEL, 5, {0, {u32=0, u64=0}}) = 0
[pid  7617] close(5)                    = 0
[pid  7617] futex(0xc82002ed10, FUTEX_WAKE, 1) = 1
[pid  7620] <... futex resumed> )       = 0
[pid  7618] <... select resumed> )      = 0 (Timeout)
[pid  7617] write(2, "Get http://registry-1.docker.io/"..., 108 <unfinished ...>
Get http://registry-1.docker.io/v2: dial tcp: lookup registry-1.docker.io on 10.155.248.190
```

mdempsky commented on Apr 26, 2016 — Member

@fraenkel Thanks, that looks more like what I would expect.

I suspect the two connects are because we're issuing both A and AAAA queries, but the strace output is truncating the DNS packets so I can't see the QTYPE field.

Anyway, I just uploaded https://go-review.googlesource.com/#/c/22428/, which I believe should fix the issue by replicating libresolv's behavior. Do you mind testing if it fixes your problem?

fraenkel commented on Apr 26, 2016 — Contributor

It works. Thanks for the quick turnaround.

gopherbot commented on Apr 26, 2016

CL https://golang.org/cl/22428 mentions this issue.

gopherbot closed this in 98b99d5 on Apr 27, 2016

mdempsky referenced this issue on Jun 30, 2016

net: LookupHost only using the first configure DNS server in /etc/resolv.conf on Linux #16215 — Closed

mikioh changed the title from **Go's DNS resolver fails with no such host** to **net: Go's DNS resolver fails with no such host** on Oct 19, 2016

mikioh added this to the **Go1.7** milestone on Oct 19, 2016

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Comment

© 2017 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub API Training Shop Blog About