

This repository

Search

Pull requests

Issues

Marketplace

Gist

golang / go

Watch2,262

Star29,207

Fork3,925

<> Code

🔔 Issues2,825

🔗 Pull requests0

📖 Wiki

🔍 Insights

net: revisit unconditional use of cgo lookups for darwin

#16345

New issue

Open danp opened this issue on Jul 14, 2016 · 20 comments



danp commented on Jul 14, 2016 • edited Contributor + 🗨️

<https://golang.org/cl/8945> changed to using the native Go stub resolver for most systems. Darwin was excluded, [here](#), due to firewall warnings.

Is this still an issue?

On my 10.11 system with the firewall enabled this test program doesn't produce any warnings, even with cgo disabled:

```
package main

import (
    "fmt"
    "net"
)

func main() {
    ns, err := net.LookupHost("www.google.com")
    if err != nil {
        panic(err)
    }
    fmt.Println(ns)
}
```

% CGO_ENABLED=0 GODEBUG=netdns=2 go run main.go
go package net: built with netgo build tag; using Go's DNS resolver
go package net: hostLookupOrder(www.google.com) = files,dns
[216.58.192.164 2607:f8b0:4009:80e::2004]

(nothing pops up when running)

There are probably other reasons to conditionally exclude Darwin for now, such as if /etc/resolver config is used ([#12524](#)), but perhaps removing this blanket condition could be a start.

cc @mdempsky and @bradfitz since you authored that change.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

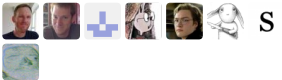
Unplanned

Notifications

🔔 Subscribe

You're not receiving notifications from this thread.

8 participants



🔖 ianlancetaylor added this to the Go1.8 milestone on Jul 14, 2016



bradfitz commented on Jul 17, 2016 Owner + 🗨️

The Darwin exclusion predates <https://golang.org/cl/8945> I thought. Can you do some digging and investigate its history?

What do we gain by using Go's resolver by default on Mac? It seems like we'd need to do [#12524](#) first as you mentioned otherwise we risk doing more harm than good.



danp commented on Jul 20, 2016 • edited Contributor + 🗨️

I did some digging!

Before <https://golang.org/cl/8945> the only way to use the Go's resolver (for most platforms, not just darwin) was to build with netgo (and/or without cgo?). This caused the [stubs to return false for](#)

completed and triggered use of the Go resolver.

I discovered along the way that android is also currently summarily excluded from using Go's resolver due to #10714, though it's done per request instead of per-conf-setup like darwin. In the CL for that, @minux called for the same to be done for iOS and then subsequently found the exclusion for darwin.

All unix platforms except for darwin and android now use Go's resolver unless weirdness is detected along here or here. I think we could gain more consistency across platforms if darwin were to join in with necessary safety checks.

Re #12524, that could be something we detect and fall back to cgo for initially. Once there is support for it in the Go resolver the check could be removed and Go's resolver could be used in that case.

At the very least if we decide not to pursue this it would be good to update the comment around the darwin exclusion to explain it's due to more than possible firewall warnings.



minux commented on Jul 20, 2016

Member +

Have you tested the pure Go resolver on older OS X systems?



danp commented on Jul 20, 2016

Contributor +

I only have immediate access to 10.11 and 10.10 systems, neither of which pop up anything when trying the test program in the description (which I've updated to show exactly how I ran it). Trying to see if any community members have access to older systems for testing. Worth noting that 10.10 would have been current at the time of <https://golang.org/cl/8945>.

I'm also unable to find anything when searching for this issue happening generally. @bradfitz, did you experience this firsthand? Is it possible it was this standard firewall warning that pops up when a process listens for outside connections, and maybe caused by something else?



groob commented on Jul 21, 2016

Contributor +

@dpiddy @bradfitz I don't think this a bug. I tested on 10.11 and the application firewall is not triggered either.

Having a "server" triggers the firewall on OS X, since that would offer an incoming connection. But DNS resolution is outgoing, so the firewall would ignore it.



danp commented on Aug 16, 2016

Contributor +

Any further thoughts on this?

I can try and put together ways to detect when we should not use Go's resolver if we want to move forward with removing the blanket condition.



bradfitz commented on Sep 9, 2016

Owner +

@dpiddy, I can test on OS X 10.8, now that we have VM-based builders and ancient versions available.



danp commented on Sep 9, 2016

Contributor +

Great! Let me know what I can do to help.



bradfitz commented on Sep 9, 2016

Owner +

I tried on OS X 10.8 with the firewall on in its most restrictive mode (block all incoming connections), and I saw no pop-up dialog using Go's DNS resolver. I set `GODEBUG=netdns=go+2` and saw it use Go's DNS resolver and get the right answers.

bradfitz commented on Sep 9, 2016



I can try and put together ways to detect when we should not use Go's resolver if we want to move forward with removing the blanket condition.

Owner



Propose away.

1



bradfitz referenced this issue on Sep 21, 2016

net: dial tcp: lookup "domain": no such host, on Darwin #17172

Closed



danp commented on Sep 21, 2016

Contributor



Hope to spend some good time on this soon. So far existence of `/etc/resolvers` is the first obvious cgo fallback condition.

What I'm more unsure of is this bit in the [darwin resolver\(5\) man page](#):

However, client configurations are not limited to file storage. The implementation of the DNS multi-client search strategy may also locate client configuratins in other data sources, such as the System Configuration Database. Users of the DNS system should make no assumptions about the source of the configuration data.

I'm not familiar with what those other data sources might be or if we can detect their use.

Issues like [docker/for-mac#19](#) suggest `scutil` can be used to discover DNS config in the System Configuration Database. Would that give enough hints on when cgo should be preferred? Would it work for all Go-supported OS X versions?

There might also be special names that should prefer cgo.

Any available insight on these things would be greatly appreciated!

And if it would help get things started to open a CL removing the blanket darwin exclusion but falling back to cgo if `/etc/resolvers` exists I can certainly do that.



quentinmit commented on Oct 8, 2016

Contributor



`scutil` is a wrapper around the SystemConfiguration framework. `scutil --dns` prints the current resolver configuration. The default configuration is DNS from DHCP/manual, followed by mdns for a number of specific domains ("local" and the PTR domains).

It looks like Chromium uses the unexported symbol `dns_configuration_copy` to get the configuration: https://chromium.googlesource.com/chromium/src/+/b4aadc32a7fd6d42ac3cc9adbb20a8ee4a267572/net/dns/dns_config_watcher_mac.cc

quentinmit added the **NeedsFix** label on Oct 8, 2016



quentinmit commented on Oct 8, 2016

Contributor



Also, another link. Here is the implementation of `dns_configuration_copy` (though I think we should use it from `libSystem` and/or `SystemConfiguration.framework`, not copy it into Go):

http://publicsource.apple.com/source/configd/configd-802.40.13/dnsinfo/dnsinfo_copy.c



danp commented on Oct 8, 2016

Contributor



Thanks for the tips!

though I think we should use it from `libSystem` and/or `SystemConfiguration.framework`, not copy it into Go

Any pointers on what that might look like?

Couple other notes from digging:

- [resolver\(5\)](#) describes support for custom ports both in `nameserver` lines with `1.2.3.4.55` for port 55 and via the `port` directive. Use of `port` would trigger [setting unknownOpt](#) on the config but the

nameserver form will need consideration.

- [like OpenBSD](#), I don't think OS X has any notion of nsswitch.conf so the condition there should probably be expanded or reworked



quentinmit commented on Oct 8, 2016

Contributor + 🗨️

OS X's version of nsswitch.conf is the data in the SystemConfiguration framework.

Look at the Chromium source code; also look at our Keychain code in crypto/x509. I imagine we'd do something similar where we weakly link the symbols and call them from cgo.

Though an open question is whether it's actually worth the complexity to use the native resolvers for some but not all queries, I think.



danp commented on Oct 8, 2016

Contributor + 🗨️

The crypto/x509 tip helped me get an idea for what would be involved, thanks.

Though an open question is whether it's actually worth the complexity to use the native resolvers for some but not all queries, I think.

If the result of this is deciding it's too complex to use Go's resolver when the native resolver (via cgo) is available, that's fine with me. At least we know and have info to consider for the future.

There might still be room for improving the experience when the native resolver is unavailable, such as with support for /etc/resolver.

🏷️ **rsc** added **NeedsDecision** and removed **NeedsFix** labels on Oct 21, 2016



rsc commented on Oct 27, 2016

Contributor + 🗨️

On my OS X 10.11.6 system, if I go to Security & Privacy and then Firewall and Turn Firewall On and then Firewall Options... and then check "Block all incoming connections", then `GODEBUG=netdns=go+2 go run lookup.go` hangs, but `GODEBUG=netdns=cgo+2 go run lookup.go` keeps working.

It is true that without "Block all incoming connections", the cgo mode does not get popup dialogs like it used to long ago. But I think we probably still can't turn on Go resolution by default.

For the record, the original CL adding cgo support for resolving host names, specifically for OS X, was [golang.org/cl/4437053](https://github.com/golang/go/pull/4437053) aka [c9164a5](#).

I agree it would be nice if we could do what we do on Linux etc where we look at resolv.conf and nsswitch.conf and decide if it's OK to use the pure Go resolver. The reason we look at nsswitch.conf is to see if there are any non-DNS lookup methods configured, and if so we delegate to the C library. On the Mac there is no nsswitch.conf but as I understand it there's effectively always a non-DNS lookup method configured (Bonjour). So if there were an accurate nsswitch.conf we'd never use the Go resolver by default.

To summarize:

1. The most restrictive OS X firewall setting makes the Go resolver hang.
2. OS X has no nsswitch.conf but if it had an accurate one we'd see non-DNS resolution methods listed and would choose to use the cgo resolver.

For both these reasons, I think we should leave the default on OS X where it is, namely using the cgo resolver.

Note that people who want to use the pure Go resolver need not recompile their programs, as in [@danp](#)'s example (the `CGO_ENABLED=0` is causing package net to be rebuilt entirely). It suffices to set `GODEBUG=netdns=go`, as mentioned in the net package doc.



bradfitz commented on Nov 2, 2016


Owner + 🗨️

On the Mac there is no nsswitch.conf but as I understand it there's effectively always a non-DNS lookup method configured (Bonjour).

That's only for *.local names, or things without a dot. Even on Linux when Avahi/mDNS stuff is listed in nslookup.conf, we only do cgo if we see *.local or no dot (if search domains are listed), irc. We could probably do the same on Darwin.

I'm going to kick this to Unplanned for now. If somebody wants to own this and do the pure Go thing when it's really safe on macOS and won't be annoying for the user, feel free to research and post your plan. It could go in Go 1.9 if there's a plan that addresses @rsc's concerns.

 **bradfitz** modified the milestone: **Unplanned, Go1.8** on Nov 2, 2016

 **bradfitz** removed the **NeedsDecision** label on Nov 2, 2016

 **bassam** referenced this issue in **rook/rook** on Nov 17, 2016

build: using netgo for macOS might cause hangs #205

Closed



danp commented on Feb 22

Contributor + 

Just tried the test on my 10.12.3 system, with "Block all incoming connections" enabled, and it worked both with `GODEBUG=netdns=go+2` and `GODEBUG=netdns=cgo+2`. Can others confirm?

S

shawnp commented on Feb 25

Member + 

@danp same result for me



Write Preview

AA ▾ B i “ < > ↺ ⋮ ≡ ≡ ↶ @ ★

Leave a comment

Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

 Styling with Markdown is supported

Comment

