New issue Jump to bottom

net: reject leading zeros in IP address parsers [freeze exception] #30999

Closed **opennota** opened this issue on Mar 22, 2019 · 33 comments

Labels	NeedsFix Proposal Proposal-Accepted release-blocker Security
Projects	Proposals (old)
Milestone	Go1.17

opennota commented on Mar 22, 2019

What version of Go are you using (`go version`)?

```
$ go version
go version go1.12.1 linux/amd64
```

Does this issue reproduce with the latest release?

Yes.

What operating system and processor architecture are you using (`go env`)?

▶ `go env` Output

What did you do?

```
package main

import "net/http"

func main() {
        http.Get("http://7.7.7.017")
}
```

What did you expect to see?

`7.7.7.017` is interpreted as `7.7.7.15`.

```
$ ping 7.7.7.017
PING 7.7.7.017 (7.7.7.15) 56(84) bytes of data.
```

What did you see instead?

The program tries to connect to `7.7.7.17`.

👍 3

agnivade changed the title ~~net/http: octal literals in IP addresses are interpreted as decimal ones~~ net/url: Parse interprets octal literals in IP addresses as decimal ones on Mar 22, 2019

agnivade commented on Mar 22, 2019 Contributor

This is more of a net/url issue rather than net/http.

What does the RFC say regarding this ?

opennota commented on Mar 22, 2019 Author

I believe that `Parse` doesn't try to interpret the hostname at all.

julieqiu added the NeedsInvestigation label on Apr 22, 2019

julieqiu added this to the **Go1.13** milestone on Apr 22, 2019

andybons modified the milestones: **Go1.13**, **Go1.14** on Jul 8, 2019

rsc modified the milestones: **Go1.14**, **Backlog** on Oct 9, 2019

secenv commented on Mar 30, 2021

As shown in this recent article, this behavior could be used in server-side request forgery, local file inclusion and remote file inclusion vulnerabilities.

tv42 commented on Mar 30, 2021

There is no real RFC on textual IP address representation. The best we have is https://tools.ietf.org/html/draft-main-ipaddr-text-rep-02 which says

> All the forms except for decimal octets are seen as non-standard (despite being quite widely interoperable) and undesirable.

I'd argue Go net.ParseIP/ParseCIDR etc should return an error on zero-prefixed input. It avoids ambiguity and since Go has historically parsed them differently than BSD, an error is a safer change in behavior than silently giving different results.

See also https://man7.org/linux/man-pages/man3/inet_pton.3.html which does not accept zero-prefixed IPs.

👍 5

tv42 commented on Mar 30, 2021

And as I discussed with **@secenv** on IRC, that article is naive. Typical "attacks" that 0127.0.0.1 enables are enabled also by `evil.example.com A 127.0.0.1` in DNS, and the fix for both is to check the target IP *after resolving*, basically `&http.Client{Transport: &http.Transport{DialContext: dialOnlySafeIPs}}`

👍 3

secenv commented on Mar 30, 2021

I forgot to add that it is indeed an issue that affects net.ParseCIDR https://play.golang.org/p/HpWqhr9tZ53 . I agree with **@tv42**, those functions should return errors. The documentation should at least warn the developer.

Guess I should mention **@FiloSottile** for further discussion on the security impact of this issue.

tv42 commented on Mar 30, 2021

I found a more authoritative RFC on IP address textual representation -- although it's only Informational not Standards Track: https://tools.ietf.org/html/rfc6943#section-3.1.1

Since Go doesn't use the "loose" syntax of RFC6943, it's non-conforming already. Rejecting non-dotted-decimal inputs would make Go use the "strict" syntax.

👍 1

rsc commented on Apr 5, 2021 Contributor

I agree about changing Go's IP address parsers
(ParseIP, ParseCIDR, any others) to reject leading zeros (except "0"),
because:

(1) the RFCs are mostly quiet but in a few places hint that decimal is the right interpretation,
(2) Go interprets leading zeros as decimal, and
(3) BSD stacks nonetheless interpret leading zeros as octal.
(4) The fact that basically no one has noticed this divergence implies
that essentially no one uses leading zeros in IP addresses.

It seems like an open question whether this should be done
in a point release or saved for the next major release (Go 1.17).
But to start, we should agree to do it at all.
Adding to the proposal process.

👍 4

rsc changed the title ~~net/url: Parse interprets octal literals in IP addresses as decimal ones~~ **net/url: reject leading zeros in IP address parsers** on Apr 5, 2021

rsc changed the title ~~net/url: reject leading zeros in IP address parsers~~ **proposal: net/url: reject leading zeros in IP address parsers** on Apr 5, 2021

rsc modified the milestones: **Backlog**, **Proposal** on Apr 5, 2021

rsc removed the **NeedsInvestigation** label on Apr 5, 2021

rsc added this to **Incoming** in **Proposals (old)** on Apr 5, 2021

rsc added the **Proposal** label on Apr 5, 2021

FiloSottile commented on Apr 6, 2021 Contributor

> And as I discussed with **@secenv** on IRC, that article is naive. Typical "attacks" that 0127.0.0.1 enables are enabled also by `evil.example.com A 127.0.0.1` in DNS, and the fix for both is to check the target IP *after resolving*, basically `&http.Client{Transport: &http.Transport{DialContext: dialOnlySafeIPs}}`

I find this pretty convincing, especially given that `net.Dial` and `net.Listen` will parse the IPs as decimal.

To end up vulnerable due to this mismatch, an application would have to parse the IP with Go, reject any hostnames, apply security-relevant logic to the return value, and then pass the input (*not* the encoding of the return value) to a different, non-Go application which is happy to parse the IP as octal.

Generally, this is another instance where relying on parser alignment instead of re-encoding outputs is a fragile design.

We are not aware of any application for which this leads to a security issue, if anyone does please let us know at security@golang.org as that would help evaluate whether to backport the fix.

In any case, I definitely agree we should just consider these inputs invalid in Go 1.17.

👍 2

bradfitz commented on Apr 7, 2021 • edited ▾ Contributor

Related: #43389 ("net: limit the size of ParseIP input?")

bradfitz mentioned this issue on Apr 7, 2021

net: limit the size of ParseIP input? #43389

Closed

rsc commented on Apr 7, 2021 Contributor

This proposal has been added to the active column of the proposals project
and will now be reviewed at the weekly proposal review meetings.
— rsc for the proposal review group

👍 1

rsc moved this from **Incoming** to **Active** in **Proposals (old)** on Apr 7, 2021

rsc mentioned this issue on Apr 7, 2021

proposal: review meeting minutes #33502

Open

liggitt mentioned this issue on Apr 7, 2021

[go1.17] Guard against stdlib ParseIP/ParseCIDR changes in API validation kubernetes/kubernetes#100895

Closed

FiloSottile changed the title ~~proposal: net/url: reject leading zeros in IP address parsers~~ proposal: net: reject leading zeros in IP address parsers on Apr 8, 2021

37 hidden items
Load more...

nckturner mentioned this issue on Aug 26, 2021

run hack/update-netparse-cve.sh kubernetes/cloud-provider-aws#261

Merged

ssajal-wr commented on Aug 31, 2021

> backporting this to go1.16 would be a breaking change that would prevent some projects from picking up go 1.16 patch releases

Hi, can you please explain why the fix is not applicable to go 1.16 release? This is for my own understanding and for those using the 1.16 release of go in the yocto community. Thanks in advance!

benjsmi commented on Sep 1, 2021

Hello. Would actually like to request that this be backported to Go v1.15 if at all possible. We'd be eternally grateful. More explanation can be found in the related comment #43389 (comment). In short: we find Go 1.15 to still be supported for amd64 and this issue falls under CVE-2021-29923 along with #43389. We'd be eternally grateful! Thank you!

tv42 commented on Sep 1, 2021

@benjsmi Go1.15 is unsupported, since Go1.16 and Go1.17 have been released. https://golang.org/doc/devel/release#policy

👍 1

rsc commented on Sep 1, 2021 Contributor

We will not be backporting this issue. We are treating the change as a robustness improvement and not a security fix due to its potential for breaking working use cases.

The situation is not nearly so clear cut as the advocates of CVE-2021-29923 would have people believe. They present it as a bug, plain and simple, not to treat leading zeros in IP addresses as indicating octal numbers, but that's not obvious. The BSD TCP/IP stack introduced the octal parsing, perhaps even accidentally, and because BSD is the most commonly used code, that interpretation is also the most common one. But it's not the only interpretation. In fact, the earliest IP RFCs directly contradict the BSD implementation - they are pretty clear that IP addresses with leading zeros are meant to be interpreted as decimal.

Furthermore, the claimed vulnerability is like a TOCTTOU problem where the check and use are handled by different software with differing interpretation of leading zeros. The right fix is, as it always is, to put the check and use together.

Rejecting the leading zeros entirely avoids resolving the radix ambiguity the wrong way, which improves robustness. But it can also break existing code that might be processing config files that contain leading zeros and were happy with the radix-10 interpretation.

Given that

1. the right fix for any security consequence does not involve this change at all (the right fix is to place the check and use in the same program), and that
2. the Go behavior is entirely valid according to some RFCs, and that
3. the change has a very real possibility of breaking existing, valid, working use cases,

we chose to make the change only in a new major version, as a robustness fix, rather than treat it as an urgent security fix that would require backporting. We do not want to push a breaking change that will keep people from being able to pick up critical Go 1.16 patches later.

👍 5

benjsmi commented on Sep 3, 2021

Thanks for the responses everyone. This is an unfortunate situation but I completely understand why you're handling it this way.

halstead pushed a commit to openembedded/openembedded-core that referenced this issue on Sep 7, 2021

`go: Exclude CVE-2021-29923 from report list` … 5bd5faf

kraj pushed a commit to YoeDistro/poky-old that referenced this issue on Sep 7, 2021

`go: Exclude CVE-2021-29923 from report list` … bc7bbf3

kraj pushed a commit to YoeDistro/poky-old that referenced this issue on Sep 7, 2021

`go: Exclude CVE-2021-29923 from report list` … 74a859f

seambot pushed a commit to seamapi/poky that referenced this issue on Sep 7, 2021

`go: Exclude CVE-2021-29923 from report list` … 1ad2ae0

halstead pushed a commit to openembedded/openembedded-core that referenced this issue on Sep 14, 2021

`go: Exclude CVE-2021-29923 from report list` … 9dfc6ab

seambot pushed a commit to seamapi/poky that referenced this issue on Sep 14, 2021

`go: Exclude CVE-2021-29923 from report list` … 7f73831

jpuhlman pushed a commit to MontaVista-OpenSourceTechnology/poky that referenced this issue on Sep 16, 2021

`go: Exclude CVE-2021-29923 from report list` … 39cbffa

halstead pushed a commit to openembedded/openembedded-core that referenced this issue on Sep 17, 2021

`go: Exclude CVE-2021-29923 from report list` … 573337b

seambot pushed a commit to seamapi/poky that referenced this issue on Sep 17, 2021

`go: Exclude CVE-2021-29923 from report list` … 44e80e4

benjsmi commented on Sep 22, 2021

So I'm not seeing this issue specifically mentioned in the Go 1.16 release notes -- is CVE-2021-29923 addressed in Go 1.16.x? And if so, which x?

👀 1

ianlancetaylor commented on Sep 22, 2021 Contributor

The net.ParseIP function rejects IPv4 addresses that contain decimal components with leading zeros in Go 1.17 but not in Go 1.16.

Per #30999 (comment) we do not plan to backport this change to Go 1.16.

zroubalik mentioned this issue on Nov 1, 2021

Found security vulnerabilities in go 1.16 kedacore/keda#2222

Closed

FiloSottile mentioned this issue on Nov 4, 2021

net/netip: IPv4 parser accepts leading zeroes #49365

Closed

gopherbot commented on Nov 5, 2021

Change https://golang.org/cl/361534 mentions this issue: `net/netip: don't accept ParseAddr with leading zeros`

gopherbot pushed a commit that referenced this issue on Nov 5, 2021

`net/netip: don't accept ParseAddr with leading zeros` … 3796df1

fis mentioned this issue on Jan 24

net: should expand IP address 1.1 to 1.0.0.1 #36822

Open

asmacdo mentioned this issue on Apr 7

Update python dependencies operator-framework/operator-sdk#5637

Merged

rsc unassigned **rolandshoemaker** on Jun 23

tatianab mentioned this issue on Sep 13

x/vulndb: potential Go vuln in std: CVE-2021-29923 golang/vulndb#993

Closed

wolffberg added a commit to wolffberg/terraform-provider-dns that referenced this issue on Sep 15

`Remove broken IP address parse test` … `dfaa438`

wolffberg added a commit to danskespil/terraform-provider-dns that referenced this issue on Sep 30

`Remove broken IP address parse test` … `10fdf00`

wolffberg added a commit to danskespil/terraform-provider-dns that referenced this issue on Sep 30

`Remove broken IP address parse test` … `3cdb867`

This was referenced last month

net/url: `unescape()` logic doesn't copy invalid bytes following % as expected by most recent spec #56732

Open

net/url: invalid percent encodings rejected by go1.19 are now accepted #56884

Closed

Assignees

No one assigned

Labels

NeedsFix Proposal Proposal-Accepted release-blocker Security

Projects

No open projects

1 closed project

Milestone

Go1.17

Development

No branches or pull requests

19 participants

and others