
KubeCon
CloudNativeCon
North America 2024

CoreDNS Plugins: A Deep Dive

John Belamaric, Google & Yong Tang, Ivanti

CoreDNS

- Flexible DNS server written in Go
- Focus on service discovery
- Plugin based architecture, easily extended
- Default DNS server in Kubernetes
- Supports DNS, DNS over TLS, DNS over gRPC
- AWS Route53, Azure DNS, Google Cloud DNS

CoreDNS

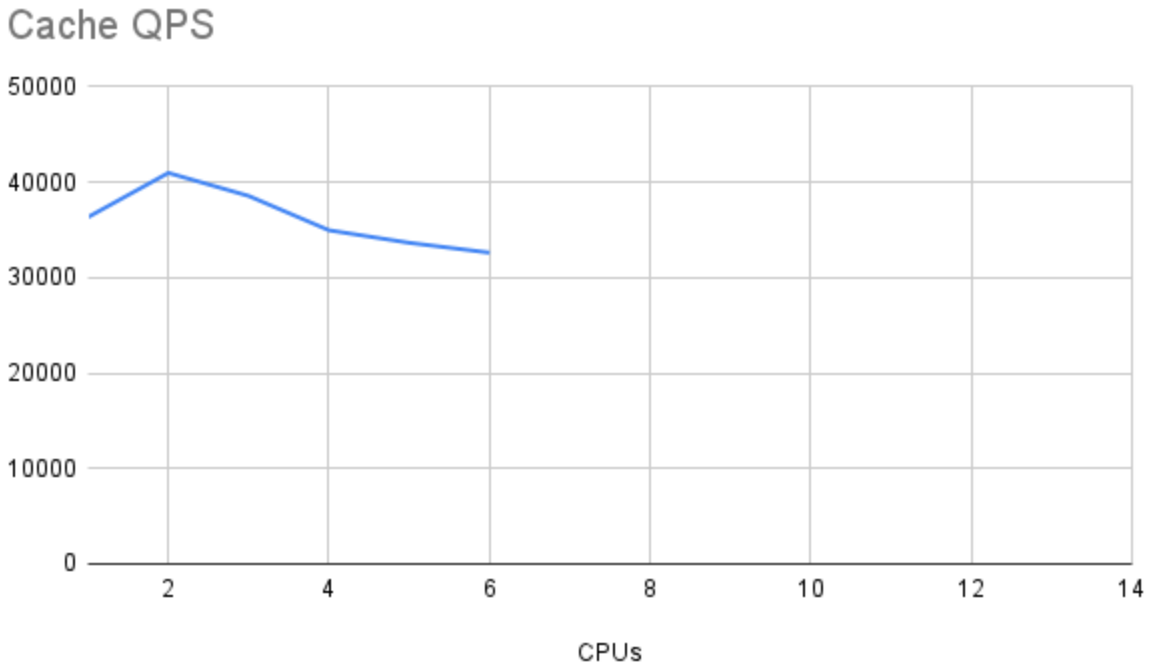
- 387 Contributors (Big Thanks!)
- 12,400+ Stars
- 36 Public Adopters
- 28 Maintainers
- Steering Committee

Recent Releases

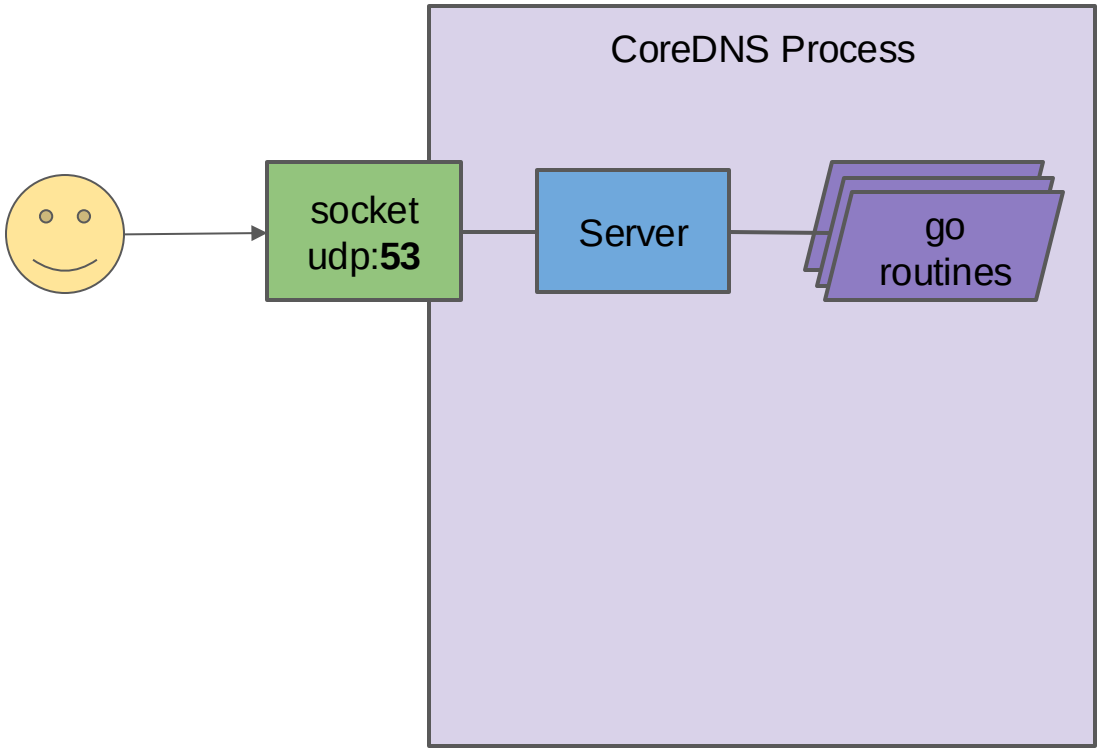
- 1.10.1- 1.11.4
  - **1.11.4** Released November 12, 2024
- New plugins:
  - **timeouts** - allows configuration of server listener timeout
- New features:
  - **acl** - drop queries as an action
  - **dnssec** - new option to load keys from AWS Secrets Manager
  - **template** - create responses with extended DNS errors
  - **cache** - option to serve original record TTLs from cache
  - **forward** - new option next, to try alternate upstreams when receiving specified response codes upstreams on (functions like the external plugin alternate)
  - **loadbalance** - add CNAME target rewrites to plugin
  - **dnstap** - add support for "extra" field in payload
  - **rewrite** - add support response code rewrite, and a new option to revert EDNS0 option rewrites in responses

More cores != more QPS

- Long standing issue
- Test with cache (no external calls)
- Peaks at ~40k qps with 2 CPUs with this machine
- **What's the bottleneck?**

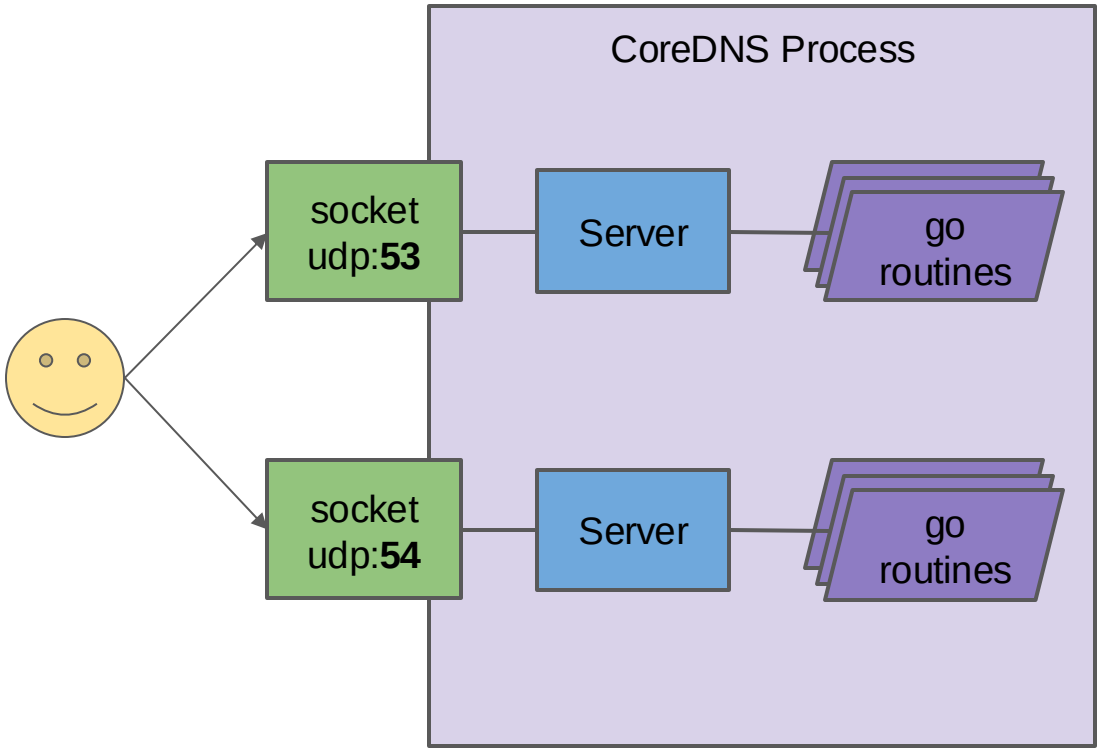


CoreDNS Internal Request Handling



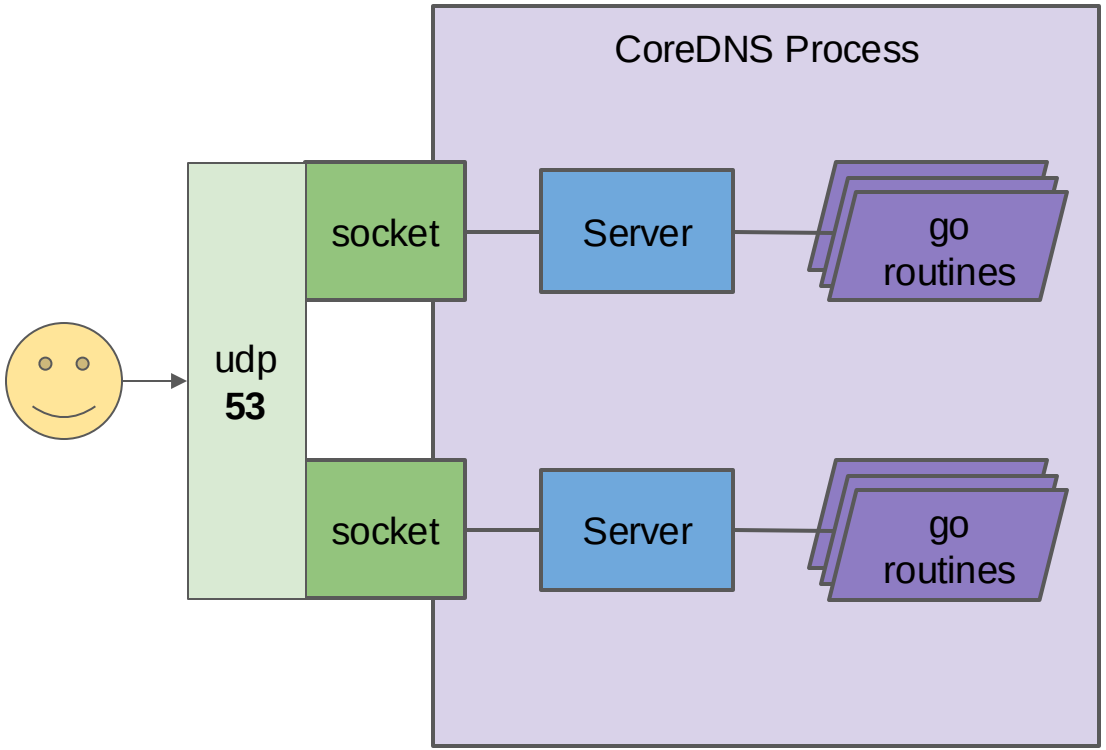
- Internal structure “Server” handles a given port
- Dispatches requests to go routines

CoreDNS Internal Request Handling



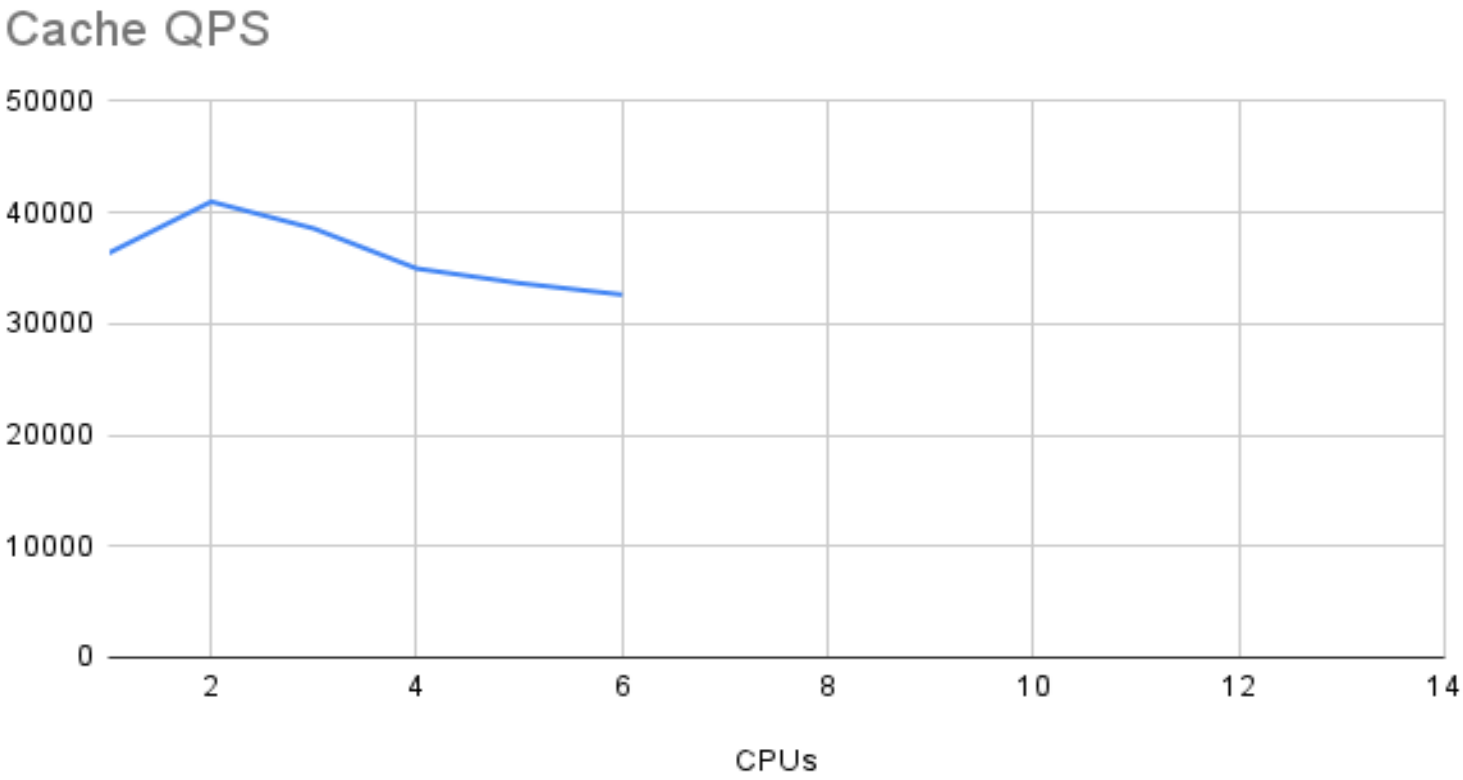
- What happens if we split traffic across ports?
- Aggregate QPS goes up!
- But we can't really force users to use two different ports

Introducing the multisocket Plugin

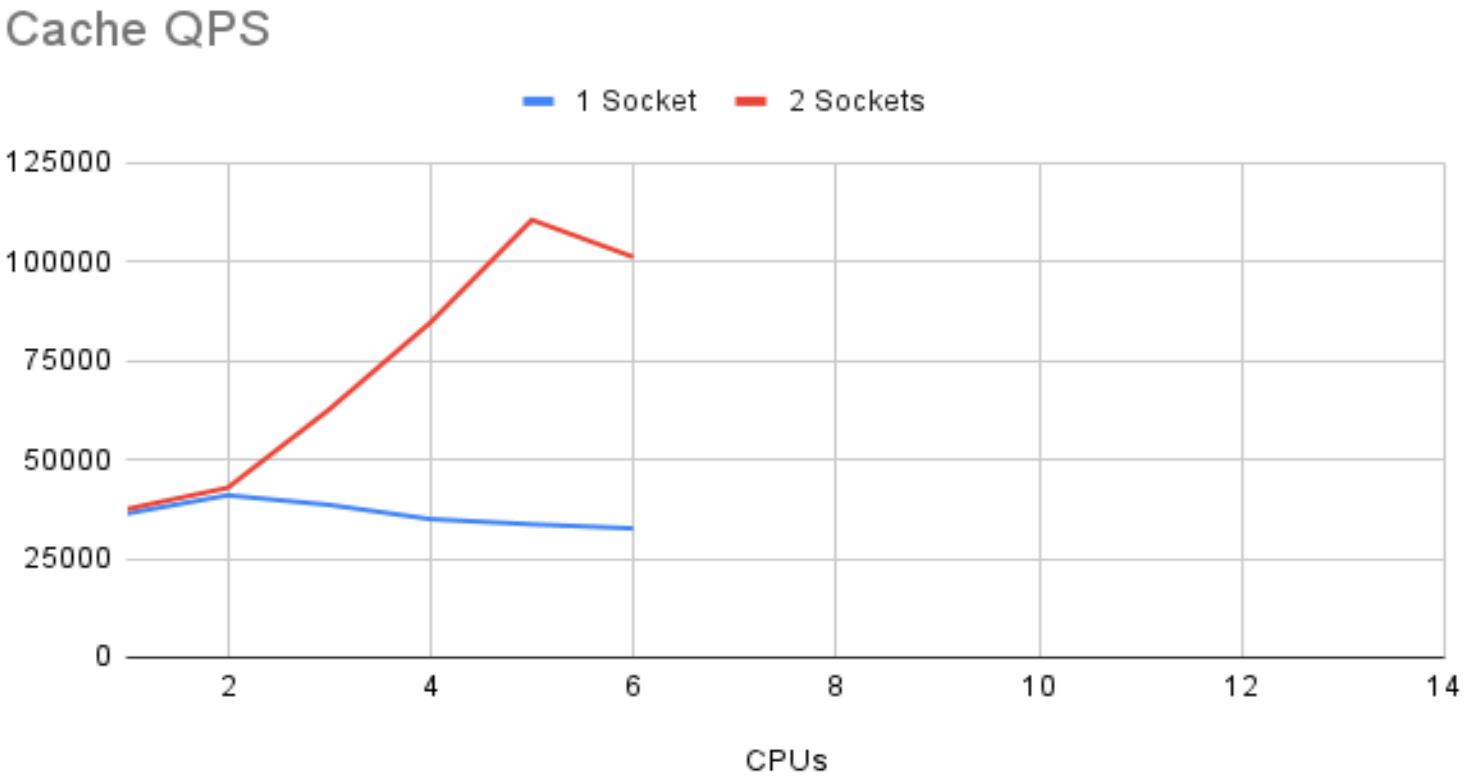


- SO_REUSEPORT
- Multiple sockets on the same port
- kernel distributes packets across the sockets
- Internal changes to create multiple servers for a port (one per socket)
- uber-go/automaxprocs

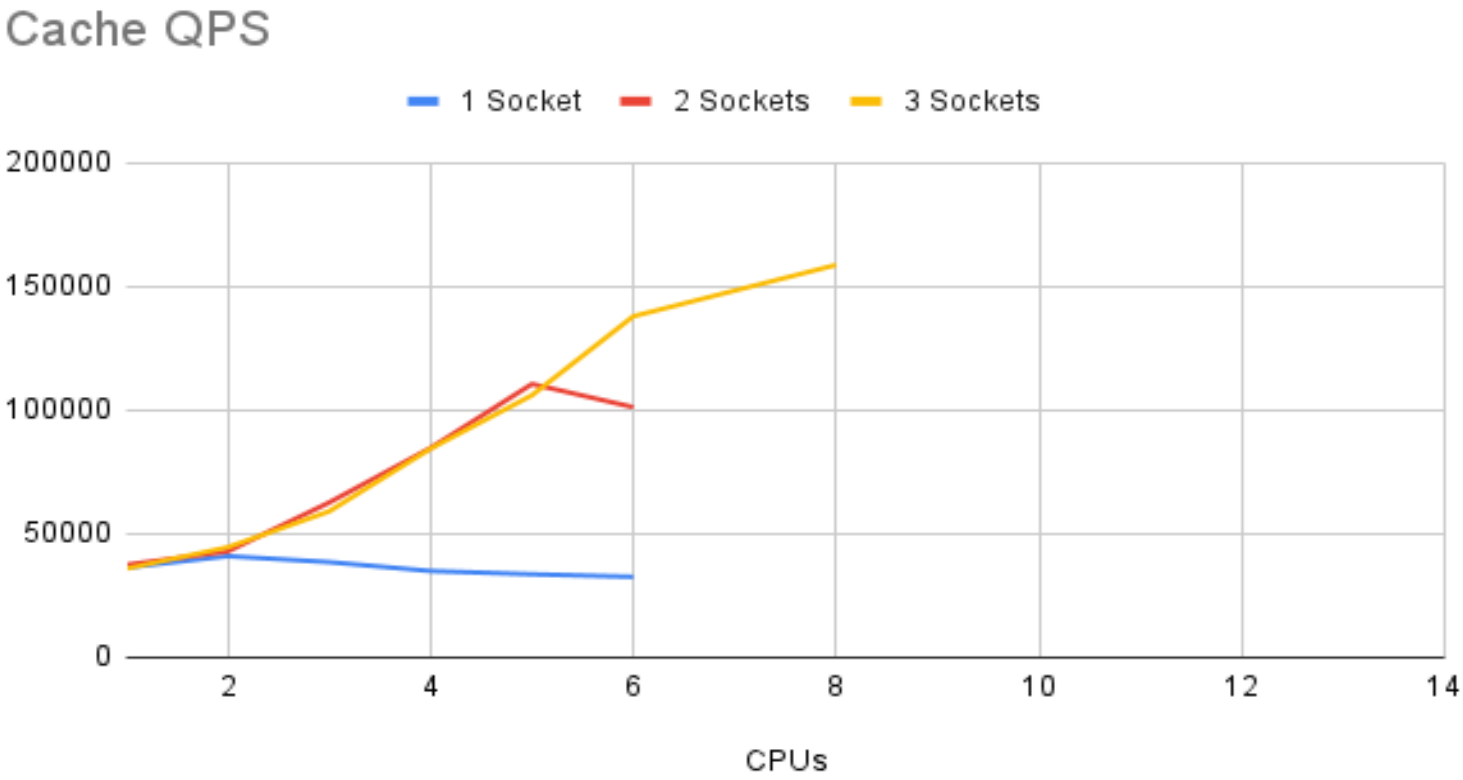
Multisocket Plugin



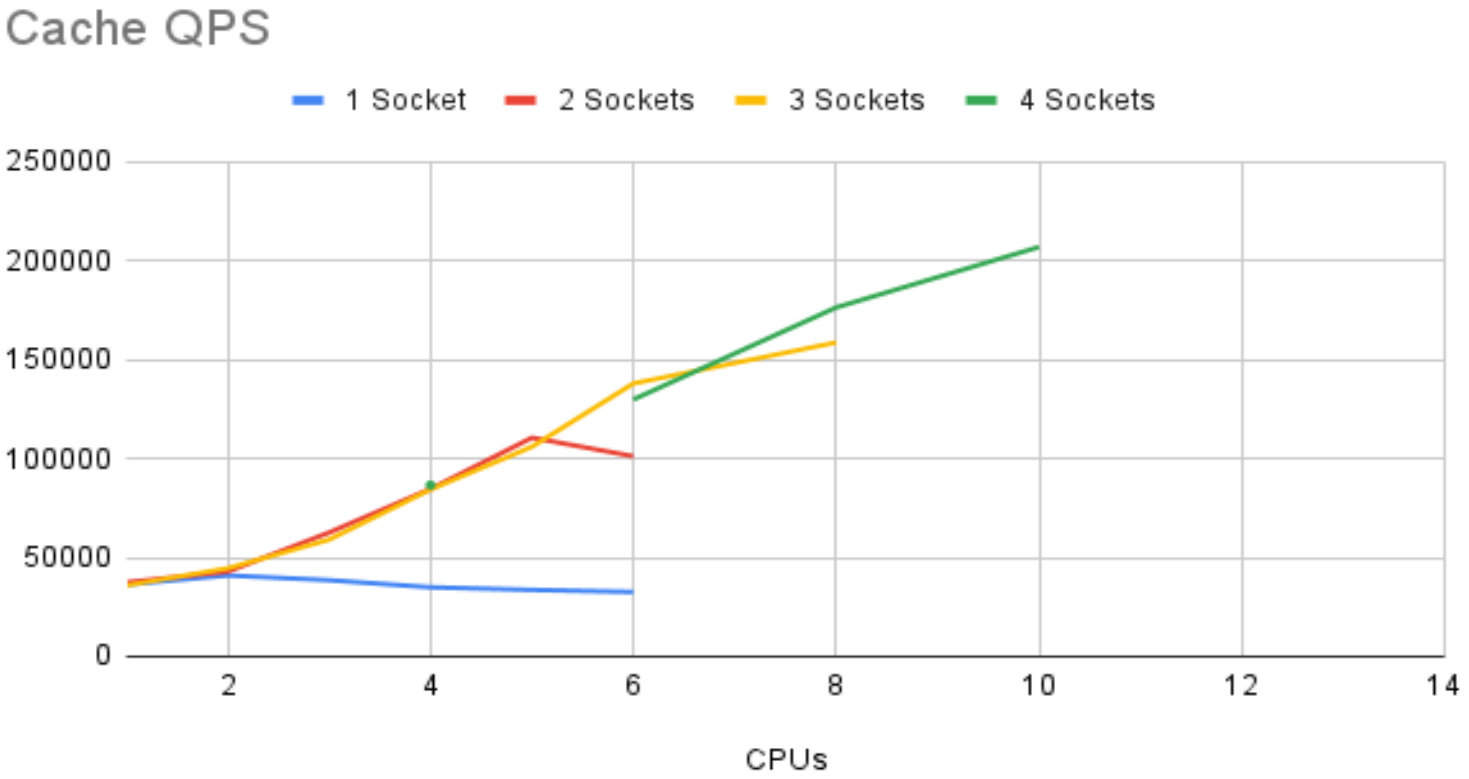
Multisocket Plugin



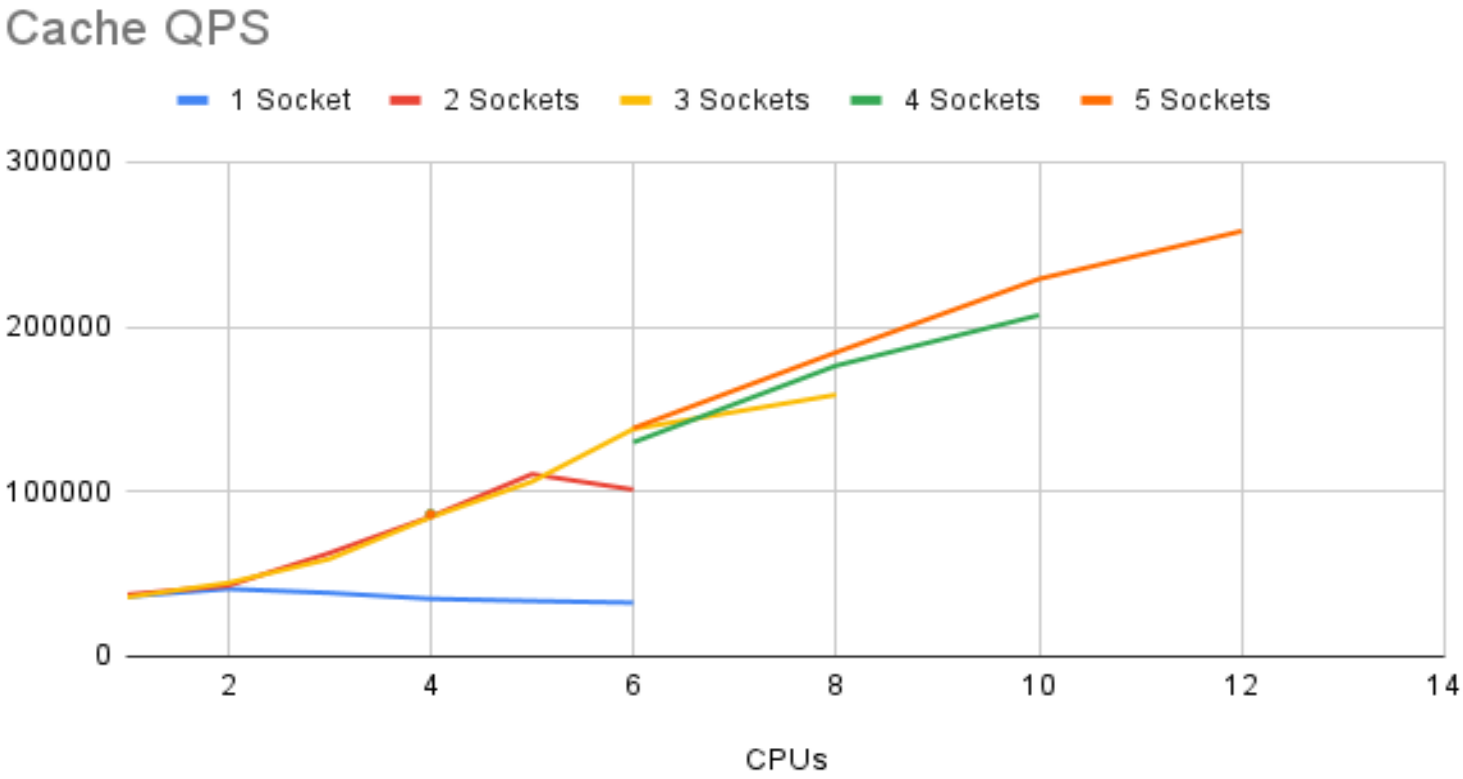
Multisocket Plugin



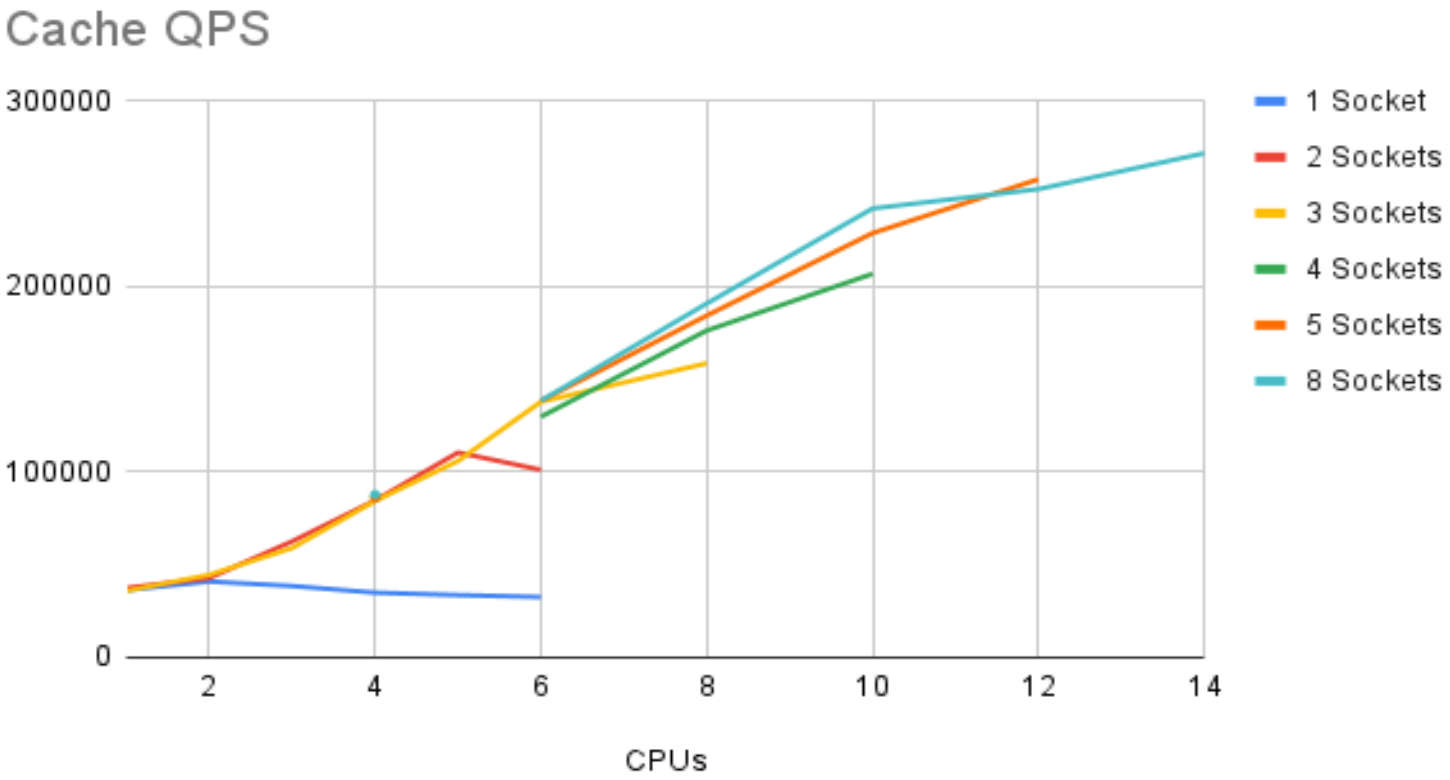
Multisocket Plugin



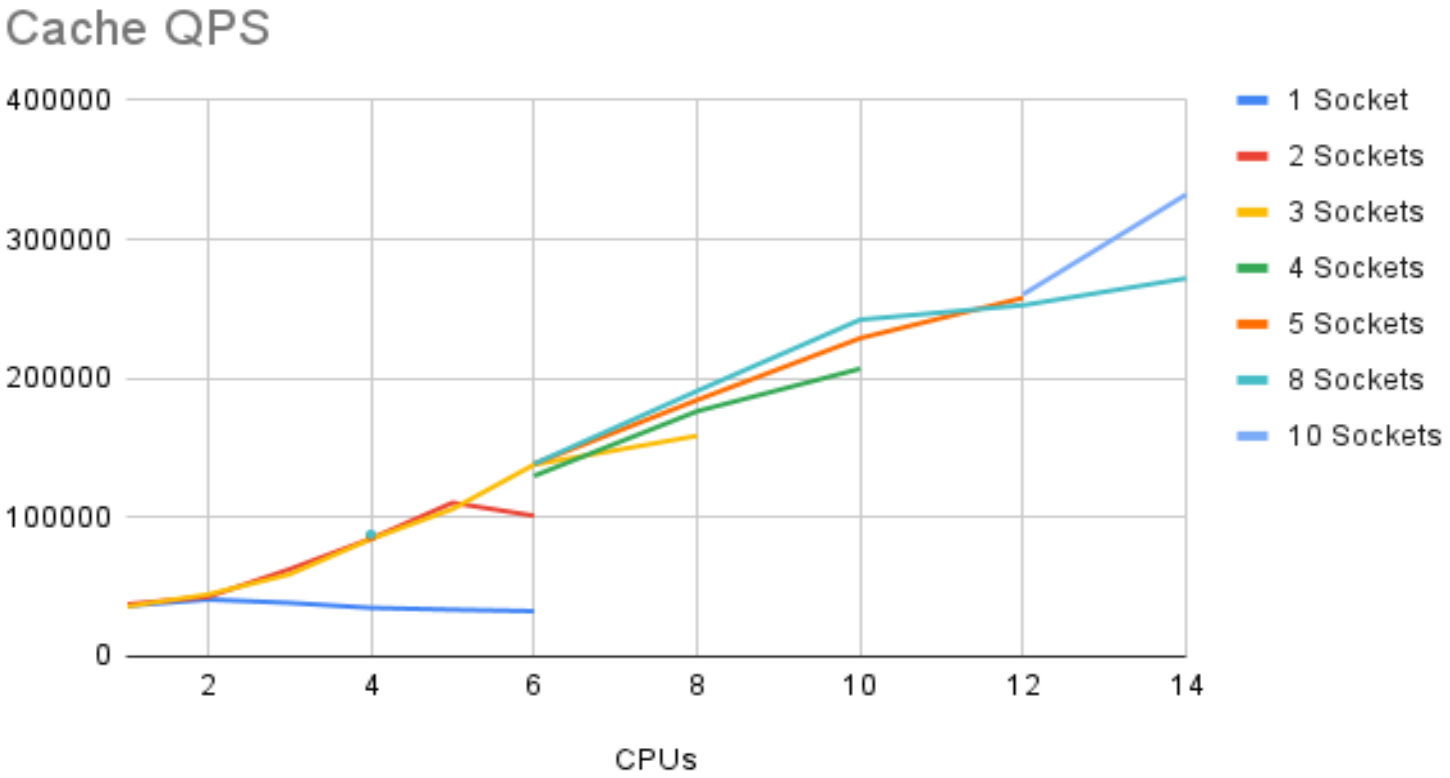
Multisocket Plugin



Multisocket Plugin



Multisocket Plugin

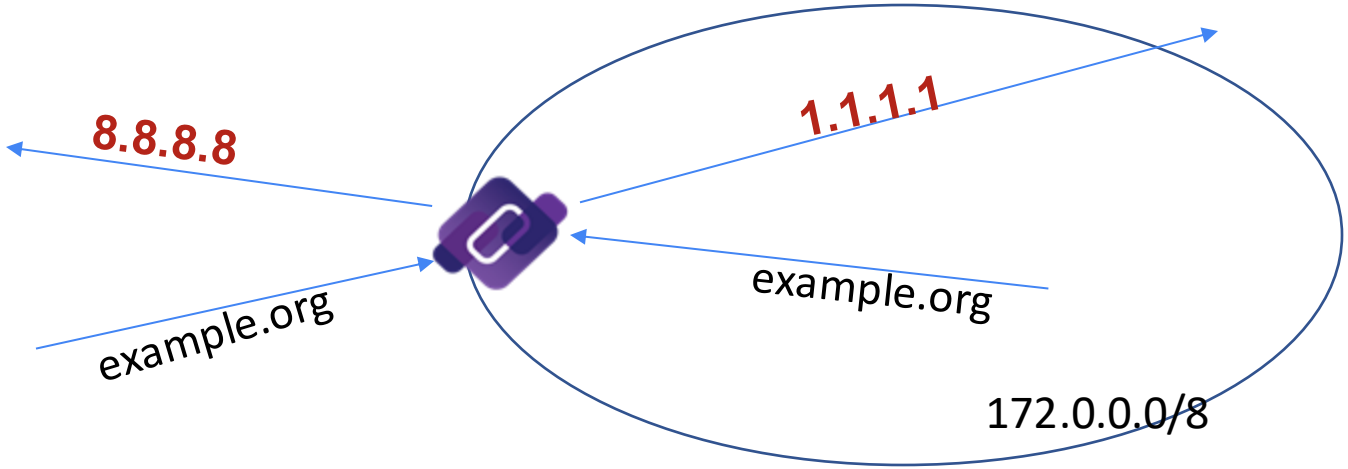


Multisocket Plugin

- Merged in master
- Will be in 1.12
- Targeting release O(weeks)
- Requires SO_REUSEPORT
- Config
  - multisocket [NUM]
- Default NUM == GOMAXPROCS
- May see better performance in specific scenarios with different NUM
  - But GOMAXPROCS is a good default

Demo Plugin

- Source IP based service discovery
  - Returns " 1.1.1.1" for 172.0.0.0/8 or 127.0.0.0/8
  - Returns " 8.8.8.8" otherwise



Demo Plugin – functions

[setup.go]:

init()
 - Performs one-time initializations, register the setup function with Caddy

setup(c *caddy.Controller) error
 - Parses the configuration from the file and captures it in a struct
 - Adds the handler to the Config object (represents a stanza)
 - Called once for each use of the plugin in the Corefile

[demo.go]:

ServeDNS(context.Context, dns.ResponseWriter,*dns.Msg) (int, error)
 - Processes the DNS request and returns a response, or,
 - Passes it down the chain

Demo Plugin – init()/setup()


```
func init() {
        caddy.RegisterPlugin("demo", caddy.Plugin{
                ServerType: "dns",
                Action:     setup,
        })
}
func setup(c *caddy.Controller) error {
        c.Next() // 'demo'
        if c.NextArg() {
                return plugin.Error("demo", c.ArgErr())
        }
        dnsserver.GetConfig(c).AddPlugin(func(next plugin.Handler) plugin.Handler {
                return Demo{}
        })
        return nil
}
```

Demo Plugin – ServeDNS()


```
// ServeDNS implements the plugin.Handler.ServeDNS.
func (p Demo) ServeDNS(ctx context.Context, w dns.ResponseWriter, r *dns.Msg) (int, error) {
        state := request.Request{W: w, Req: r}
        qname := state.Name()
        reply := ”8.8.8.8"
        if strings.HasPrefix(state.IP(), "172.") || strings.HasPrefix(state.IP(), "127.") {
                reply = ”1.1.1.1"
        }
        fmt.Printf("Received query %s from %s, expected to reply %s\n", qname, state.IP(),
reply)
        answers := []dns.RR{}
        …
}
```

Demo Plugin – Corefile


```
.:1053 {
    #
    # By default all plugins
    # are disabled initially,
    # unless enabled explicitly
    #
    demo
}
```

Demo Plugin – Build


```
$
$ # add demo:demo to plugin.cfg
$
$ # build with docker (golang:1.22)
$ docker run --rm -i -t \
$   -v $PWD:/go/src/github.com/coredns/coredns \
$     -w /go/src/github.com/coredns/coredns \
$       golang:1.22 sh -c \
$           'GOFLAGS="-buildvcs=false" make gen && GOFLAGS="-buildvcs=false" make'
$
$ # configure Corefile and run coredns
$
$ ./coredns
$
```

Demo Plugin – Source Code

https://github.com/coredns/demo

Contribute to CoreDNS

- Star CoreDNS in GitHub:
  - https://github.com/coredns/coredns
- Add name to ADOPTERS.MD
- Becomes a contributor:
  - Create a pull request
- Becomes a maintainer:
  - One significant pull request
  - Sponsored by one current maintainer


KubeCon
CloudNativeCon
North America 2024
THANK YOU