OX PowerDNS Cloud Control

Reference

Dec 15, 2021

Release 2.0.0-BETA1

Contents

1 Overview

Configuration of a deployment can be achieved by overriding specific values that are exposed by the Helm chart. This reference guide lists all the values that can be set/configured. Since there are many different configuration options available, these have been separated into the following sections inside the helm values:

- **global**: Overrides to pull images from a private registry
- **registrySecrets**: Configuration of credentials for the image registry
- **ipFamily**: Configuration of the cluster networking
- **dnsdist**: Default configuration for all dnsdist instances
- **dnsdists**: Collection of dnsdist instances to deploy
- **rulesets**: Sets of rules to deploy, for dnsdist to use
- **recursor**: Default configuration for all recursor instances
- **recursors**: Collection of recursor instances to deploy
- **resolver**: Default configuration for all resolver instances
- **resolvers**: Collection of resolver instances to deploy
- **prometheus**: Configuration of Prometheus scrape settings

Each of these sections is explained in detail in the chapters below.

For a basic example of how to use the values, please refer to the *Getting Started* chapter in the *Overview* guide.

2 Registry configuration

All images referenced are stored on a registry at registry.open-xchange.com. To ensure the images can be pulled you need to provide the corresponding registry's credentials. This can be done in the following block in the helm values (replace the username & password accordingly):

```
registrySecrets:
  registry: registry.open-xchange.com
  username: REGISTRY_USERNAME_HERE
  password: REGISTRY_PASSWORD_HERE
  email: user@some.domain.com
```

If you intend to run the monitoring stack on a kubernetes cluster which makes use of a local/private registry, you can use one or more of the following settings in your helm values to configure that registry:

```
# Monitoring - global overrides
global:
  # Override image-related settings for this chart and all subcharts
  image:
    # Override registry for all images
    registry: "myregistry.local:8085"

    # Override repository for all images
    repository: "myrepository"

    # Override pullPolicy for all images
    pullPolicy: "IfNotPresent"
```

Each setting explained:

global.image.registry

All images will be attempted to be pulled from this registry (format: host:port)

global.image.repository

All images will be attempted to be pulled from this repository, on above configured registry

global.image.pullPolicy

This pull policy will be specified for each image

If you have a need to override the above settings for specific images, you can find the corresponding 'image:' configuration blocks in the helm values file.

3 Cluster networking

To be able to support Kubernetes clusters with IPv4, IPv6 or dual stack (IPv4 & IPv6) configurations, it is required to ensure the 'ipFamily' configuration in the helm values matches your cluster. The 'ipFamily' section contains the following parameters:

- **ipFamily.ipv4**: Whether or not your cluster has IPv4 enabled (Default: true)
- **ipFamily.ipv6**: Whether or not your cluster has IPv6 enabled (Default: false)
- **ipFamily.families**: Preference of IP families on your cluster, if it is a dualstack cluster

To ensure your deployment is correctly configured, you need to provide one of the 4 possible variations:

3.1 IPv4 only (default)

```
# Networking configuration
ipFamily:
  ipv4: true
  ipv6: false
  families:
    - "IPv4"
    - "IPv6"
```

Note: 'families' is ignored in this configuration. It is only used in a dualstack setup.

3.2 IPv6 only

```
# Networking configuration
ipFamily:
  ipv4: false
  ipv6: true
  families:
    - "IPv4"
    - "IPv6"
```

Note: 'families' is ignored in this configuration. It is only used in a dualstack setup.

3.3 Dualstack - IPv4 primary

If you are running a dualstack cluster, you can check any Pod to see if your cluster has a preference for IPv4 or IPv6. Your pods will have a 'podIP' and 2 values for 'podIPs'. If the 'podIP' is an IPv4 address as shown in the example below, then you are running a cluster with IPv4 as primary:

```
podIP: 172.17.183.4 # IPv4
podIPs:
  - ip: 172.17.183.4 # IPv4
  - ip: fd43:128b:8658:b73b:3eb7:2e30:8815:3f6 # IPv6
```

Configuration for dualstack with IPv4 primary:

```
# Networking configuration
ipFamily:
  ipv4: true
  ipv6: true
  families:
    - "IPv4" # IPv4 is primary
    - "IPv6"
```

3.4 Dualstack - IPv6 primary

If you are running a dualstack cluster, you can check any Pod to see if your cluster has a preference for IPv4 or IPv6. Your pods will have a 'podIP' and 2 values for 'podIPs'. If the 'podIP' is an IPv6 address as shown in the example below, then you are running a cluster with IPv6 as primary:

```
podIP: fd43:128b:8658:b73b:3eb7:2e30:8815:3f6 # IPv6
podIPs:
  - ip: fd43:128b:8658:b73b:3eb7:2e30:8815:3f6 # IPv6
  - ip: 172.17.183.4 # IPv4
```

Configuration for dualstack with IPv6 primary:

```
# Networking configuration
ipFamily:
  ipv4: true
  ipv6: true
  families:
    - "IPv6" # IPv6 is primary
    - "IPv4"
```

4 Exposing dnsdist

By default a set of dnsdist instances will not have a Service object created. If you wish to expose dnsdist you can do so by configuring the Service object.

For these examples we will assume the following dnsdist instance is to be exposed:

```
dnsdists:
  mydnsdist:
    replicas: 2
    pools:
      default:
        serverGroups:
          - myrecursor
        packetcache:
          maxEntries: 200000
```

If the above is deployed, 2 replicas of dnsdist will be created, but no Service object will exist.

4.1 Exposing via ClusterIP

To expose dnsdist via a ClusterIP, you can specify the appropriate type and choose the ports to be included. In this example, both UDP & TCP will be exposed via a ClusterIP on port 53:

```
dnsdists:
  mydnsdist:
    replicas: 2
    pools:
      default:
        serverGroups:
          - myrecursor
        packetcache:
          maxEntries: 200000
    service:
      type: ClusterIP
      ports:
        udp:
          port: 53
          protocol: UDP
        tcp:
          port: 53
          protocol: TCP
```

4.2 Exposing via NodePort

To expose dnsdist via a NodePort, you can specify the appropriate type and choose the ports to be included. In this example, both UDP & TCP will be exposed via a NodePort, where the NodePort will be randomly selected from the node-port range:

```
dnsdists:
  mydnsdist:
    replicas: 2
    pools:
      default:
        serverGroups:
          - myrecursor
        packetcache:
          maxEntries: 200000
    service:
      type: NodePort
      ports:
        udp:
          port: 53
          protocol: UDP
        tcp:
          port: 53
          protocol: TCP
```

You can also specify the NodePort which you want to use, for example:

```
dnsdists:
  mydnsdist:
    replicas: 2
    pools:
      default:
        serverGroups:
          - myrecursor
        packetcache:
          maxEntries: 200000
    service:
      type: NodePort
      ports:
        udp:
          port: 53
          protocol: UDP
          nodePort: 30053
        tcp:
          port: 53
          protocol: TCP
          nodePort: 30053
```

4.3 Exposing via LoadBalancer

To expose dnsdist via a LoadBalancer, you can specify the appropriate type and choose the ports to be included. When 'LoadBalancer' is requested, it will create 1 ClusterIP Service + 1 LoadBalancer Service for each specified port. This is done to satisfy a Kubernetes constraint regarding single LoadBalancer objects when multiple protocols are requested.

In this example, both UDP & TCP will be exposed via a LoadBalancer Service (MetalLB annotation procided as example, substitute with annotations required by your LoadBalancer provider) on port 53:

```
dnsdists:
  mydnsdist:
    replicas: 2
    pools:
      default:
        serverGroups:
          - myrecursor
        packetcache:
          maxEntries: 200000
    service:
      type: LoadBalancer
      annotations:
        metallb.universe.tf/address-pool: name_of_pool
      ports:
        udp:
          port: 53
          protocol: UDP
        tcp:
          port: 53
          protocol: TCP
```

If necessary, you can also request a specific LoadBalancer IP, as shown in the following example:

```
dnsdists:
  mydnsdist:
    replicas: 2
    pools:
      default:
        serverGroups:
          - myrecursor
        packetcache:
          maxEntries: 200000
    service:
      type: LoadBalancer
      annotations:
        metallb.universe.tf/address-pool: name_of_pool
      loadBalancerIP: 12.34.56.78
      ports:
        udp:
          port: 53
          protocol: UDP
        tcp:
          port: 53
          protocol: TCP
```

5 dnsdist defaults

Default settings for all dnsdist instances, configurable under 'dnsdist' in helm values.

- **aclAdd**: Netmasks allowed to access dnsdist, in addition to the loopback, RFC1918 and other local addresses. Only applicable if aclAllowAll is *false* (default: *[]*)
- **aclAllowAll**: Allow all inbound traffic to dnsdist, regardless of source IP. (default: *true*)
- **affinity**: Kubernetes pod affinity (default: *{}*)
- **consistentHashingBalancingFactor**: Set the maximum imbalance between the number of outstanding queries intended for a given server, based on its weight, and the actual number, when using the chashed consistent hashing load-balancing policy. Default is 0, which disables the bounded-load algorithm. (default: *0*)
- **do53Locals**: Default amount of Do53 listen sockets per dnsdist pod (default: *1*)
- **do53TcpFastOpenQueueSize**: Default size of the TCP Fast Open queue on Do53 listen sockets (Dnsdist default)
- **do53TcpListenQueueSize**: Default size of the listen queue on Do53 listen sockets (Dnsdist default)
- **nodeSelector**: Kubernetes pod nodeSelector (default: *{}*)
- **port**: Port on which dnsdist will listen for Do53 traffic (default: *5353*)
- **readinessDo53ProbeInterval**: How often (in seconds) the agent will judge the health state of dnsdist via tests against the Do53 listeners. (default: *2*)
- **readinessDo53QDomain**: Domain used in the query to judge whether the Do53 listeners of a dnsdist container are healthy and ready for traffic. (default: *a.root-servers.net*)
- **readinessDo53QTimeout**: Number of seconds after which the query used to judge whether the Do53 listeners of a dnsdist container are healthy and ready for traffic time out. (default: *1*)
- **readinessDo53QType**: Type of query used to judge whether the Do53 listeners of a dnsdist container are healthy and ready for traffic. (default: *A*)
- **readinessFailureThreshold**: When a probe fails, Kubernetes will try this many times before marking the container as Unready. Updates deployment, resulting in respawn of dnsdist pods (default: *2*)
- **readinessInitialDelaySeconds**: Number of seconds after the containers have started before readiness probes are initiated. Updates deployment, resulting in respawn of dnsdist pods. (default: *5*)
- **readinessPeriodSeconds**: How often (in seconds) to perform the readiness probes. Updates deployment, resulting in respawn of dnsdist pods (default: *2*)

- **readinessStateProbeInterval**: How often (in seconds) the agent will judge the health state of the dnsdist agent. (default: *2*)
- **readinessSuccessThreshold**: Minimum consecutive succesfull probes before a container is marked Ready. Updates deployment, resulting in respawn of dnsdist pods (default: *2*)
- **readinessTimeoutSeconds**: Number of seconds after which the readiness probes time out. Updates deployment, resulting in respawn of dnsdist pods (default: *1*)
- **replicas**: Default number of replicas in a dnsdist deployment (default: *2*)
- **roundRobinFailOnNoServer**: By default the roundrobin load-balancing policy will still try to select a backend even if all backends are currently down. Setting this to true will make the policy fail and return that no server is available instead. (default: *false*)
- **servFailWhenNoServer**: If true, return a ServFail when no servers are available, instead of the default behaviour of dropping the query. (default: *false*)
- **serviceAccount**: Name of the service account used by dnsdist (default: *dnsdist-serviceaccount*)
- **webserverACL**: Netmasks to allow webserver traffic from (default: *127.0.0.1/32, 192.168.0.0/16, 10.0.0.0/8, 172.16.0.0/12, ::1, fc00::/7*)
- **webserverPort**: Port on which the dnsdist webserver will listen (default: *8083*)
- **weightedBalancingFactor**: Set the maximum imbalance between the number of outstanding queries intended for a given server, based on its weight, and the actual number, when using the whashed or wrandom load-balancing policy. Default is 0, which disables the bounded-load algorithm. (default: *0*)
- **ecs**: (dnsdist ECS functions)
  - **setECSOverride**: Whether to override an existing EDNS Client Subnet option present in the query (only effective when the backend server has annotation useClientSubnet = true). (Dnsdist default)
  - **setECSSourcePrefixV4**: Truncate the requestors IPv4 address to this length, in bits (only effective when the backend server has annotation useClientSubnet = true). (Dnsdist default)
  - **setECSSourcePrefixV6**: Truncate the requestors IPv6 address to this length, in bits (only effective when the backend server has annotation useClientSubnet = true). (Dnsdist default)
- **hpa**: (Horizontal Pod Autoscaler defaults)
  - **enabled**: Enable or disable the HPA (default: *false*)
  - **minReplicas**: Minimum # of replicas (default: *2*)
  - **maxReplicas**: Maximum # of replicas (default: *4*)
  - **metrics**: Metric(s) upon which to make decisions regarding scaling. Must be an array of MetricSpec (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#metricspec-v2beta2-autoscaling). Examples are in the helm values.

  - **behavior**: Behavior of the HPA, must be instance of HorizontalPodAutoscalerBehavior (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#horizontalpodautoscalerspec-v2beta2-autoscaling).

- **ringBuffers**: (Ringbuffer configuration)
  - **retries**: Number of shards to attempt to lock without blocking before giving up and simply blocking while waiting for the next shard to be available. Defaults to 5 if there is more than 1 shard, else it defaults to 0. (Dnsdist default)
  - **shards**: Number of shards to use to limit lock contention, defaults to 1. (Dnsdist default)
  - **size**: Maximum amount of queries to keep in the ringbuffer, defaults to 10000. (Dnsdist default)
- **securityPolling**: (Security polling configuration)
  - **securityPollInterval**: Interval between security pollings, in seconds. Defaults to 3600. (Dnsdist default)
  - **securityPollSuffix**: Domain name from which to query security update notifications. Setting this to an empty string disables secpoll. (Dnsdist default)
- **tuning**: (dnsdist tuning functions)
  - **setCacheCleaningDelay**: Interval in seconds between two runs of the cache cleaning algorithm, removing expired entries. (Dnsdist default)
  - **setCacheCleaningPercentage**: Percentage of the cache that the cache cleaning algorithm will try to free by removing expired entries. By default (100), all expired entries are removed. (Dnsdist default)
  - **setMaxTCPClientThreads**: Maximum amount of TCP client threads, handling TCP connections. By default this value is 10, unless more than 10 TCP listen sockets have been defined. (Dnsdist default)
  - **setMaxTCPConnectionDuration**: Maximum duration of an incoming TCP connection, in seconds. 0 (the default) means unlimited. (Dnsdist default)
  - **setMaxTCPConnectionsPerClient**: Maximum number of TCP connections per client. 0 (the default) means unlimited. (Dnsdist default)
  - **setMaxTCPQueriesPerConnection**: Maximum number of queries in an incoming TCP connection. 0 (the default) means unlimited. (Dnsdist default)
  - **setMaxTCPQueuedConnections**: Maximum number of TCP connections queued (waiting to be picked up by a client thread), defaults to 1000. 0 means unlimited. (Dnsdist default)
  - **setMaxUDPOutstanding**: Maximum number of outstanding UDP queries to a given backend server, defaults to 65535. (Dnsdist default)
  - **setStaleCacheEntriesTTL**: TTL for expired cache entries to be eligible as answer when no backends are available for a query, in seconds. (Dnsdist default)
  - **setTCPRecvTimeout**: Read timeout on TCP connections from the client, in seconds. (Dnsdist default)
  - **setTCPSendTimeout**: Write timeout on TCP connections from the client, in seconds. (Dnsdist default)

  - **setUDPTimeout**: Maximum time dnsdist will wait for a response from a backend over UDP, in seconds. Defaults to 2. (Dnsdist default)
- **agentResources**: (dnsdist agent resource allocation defaults)
  - **limits**: (Limit amounts)
    - **cpu**: Limit amount of CPU (Kubernetes default)
    - **memory**: Limit amount of memory (Kubernetes default)
  - **requests**: (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)
- **rpcServerResources**: (dnsdist RPC server resource allocation defaults)
  - **limits**: (Limit amounts)
    - **cpu**: Limit amount of CPU (Kubernetes default)
    - **memory**: Limit amount of memory (Kubernetes default)
  - **requests**: (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)
- **stateResources**: (dnsdist state resource allocation defaults)
  - **limits**: (Limit amounts)
    - **cpu**: Limit amount of CPU (Kubernetes default)
    - **memory**: Limit amount of memory (Kubernetes default)
  - **requests**: (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)
- **resources**: (dnsdist resource allocation defaults)
  - **limits**: (Limit amounts)
    - **cpu**: Limit amount of CPU (Kubernetes default)
    - **memory**: Limit amount of memory (Kubernetes default)
  - **requests**: (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)

6 Recursor defaults

Default settings for all recursor instances, configurable under 'recursor' in helm values.

- **affinity**: Kubernetes pod affinity (default: *{}*)
- **containerPort**: Port on which recursor will listen for traffic (default: *5353*)
- **nodeSelector**: Kubernetes pod nodeSelector (default: *{}*)
- **replicas**: Default number of replicas in a recursor deployment (default: *2*)
- **serviceAccount**: Name of the service account used by recursor (default: *recursor-serviceaccount*)
- **webserverACL**: Netmasks to allow webserver traffic from (default: *127.0.0.1/32, 192.168.0.0/16, 10.0.0.0/8, 172.16.0.0/12, ::1, fc00::/7*)
- **webserverPort**: Port on which the dnsdist webserver will listen (default: *8083*)
- **hpa**: (Horizontal Pod Autoscaler defaults)
  - **enabled**: Enable or disable the HPA (default: *false*)
  - **minReplicas**: Minimum # of replicas (default: *2*)
  - **maxReplicas**: Maximum # of replicas (default: *4*)
  - **metrics**: Metric(s) upon which to make decisions regarding scaling. Must be an array of MetricSpec (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#metricspec-v2beta2-autoscaling). Examples are in the helm values.
  - **behavior**: Behavior of the HPA, must be instance of HorizontalPodAutoscalerBehavior (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#horizontalpodautoscalerspec-v2beta2-autoscaling).
- **resources**: (recursor resource allocation defaults)
  - **limits**: (Limit amounts)
    - **cpu**: Limit amount of CPU (Kubernetes default)
    - **memory**: Limit amount of memory (Kubernetes default)
  - **requests**: (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)

7 Resolver defaults

Default settings for resolver instances, configurable under ‘resolver’ in helm values.

- **backendDefaultPort**: Port on which the endpoints belonging to a resolver partition will receive traffic (default: *53*)
- **servicePort**: Port on which a resolver service will listen (unused in normal operation, as the service is only used to discover the corresponding endpoints). (default: *53*)

8 Instances

8.1 dnsdists

Configuration of dnsdist instances

Key: name (Name of the dnsdist instance)

8.1.1 Parameters

- **aclAdd**: Netmasks allowed to access dnsdist, in addition to the loopback, RFC1918 and other local addresses. Overrules a AllowAll setting (if configured) on the global dnsdist defaults.
- **aclAllowAll**: Allow all inbound traffic to dnsdist, regardless of source IP.
- **affinity**: Kubernetes pod affinity
- **do53Locals**: Default amount of Do53 listen sockets per dnsdist pod (default: *1*)
- **do53TcpFastOpenQueueSize**: Default size of the TCP Fast Open queue on Do53 listen sockets (Dnsdist default)
- **do53TcpListenQueueSize**: Default size of the listen queue on Do53 listen sockets (Dnsdist default)
- **hostNetwork**: Use host networking for dnsdist pods
- **nodeSelector**: Kubernetes pod nodeSelector
- **replicas**: Default number of replicas in a dnsdist deployment (default: *2*)
- **revisionHistoryLimit**: Default 'revisionHistoryLimit' for dnsdist deployments (default: *0*)
- **verbose**: Be verbose (default: *false*)

8.1.2 Parameter Sets

- **config** (Dnsdist Configuration)
  - **addEDNSToSelfGeneratedResponses**: Whether to add EDNS to self-generated responses, provided that the initial query had EDNS. (Dnsdist default)
  - **allowEmptyResponse**: Set to true (defaults to false) to allow empty responses (qdcount=0) with a NoError or NXDomain rcode (default) from backends. dnsdist drops these responses by default because it can't match them against the initial query since

they don't contain the qname, qtype and qclass, and therefore the risk of collision is much higher than with regular responses. (Dnsdist default)

  - **consistentHashingBalancingFactor**: Maximum imbalance between the number of outstanding queries intended for a given server, based on its weight, and the actual number, when using the chashed consistent hashing load-balancing policy. Default is 0, which disables the bounded-load algorithm. (Dnsdist default)
  - **payloadSizeOnSelfGeneratedAnswers**: Set the UDP payload size advertised via EDNS on self-generated responses. In accordance with RFC 6891, values lower than 512 will be treated as equal to 512. (Dnsdist default)
  - **roundRobinFailOnNoServer**: By default the roundrobin load-balancing policy will still try to select a backend even if all backends are currently down. Setting this to true will make the policy fail and return that no server is available instead. (Dnsdist default)
  - **servFailWhenNoServer**: If set, return a ServFail when no servers are available, instead of the default behaviour of dropping the query. (Dnsdist default)
  - **verboseHealthChecks**: Set whether health check errors should be logged. This is turned off by default. (Dnsdist default)
  - **weightedBalancingFactor**: Maximum imbalance between the number of outstanding queries intended for a given server, based on its weight, and the actual number, when using the whashed or wrandom load-balancing policy. Default is 0, which disables the bounded-load algorithm. (Dnsdist default)

- **ecs** (dnsdist ECS functions)
  - **setECSOverride**: Whether to override an existing EDNS Client Subnet option present in the query (only effective when the backend server has annotation useClientSubnet = true). (Dnsdist default)
  - **setECSSourcePrefixV4**: Truncate the requestors IPv4 address to this length, in bits (only effective when the backend server has annotation useClientSubnet = true). (Dnsdist default)
  - **setECSSourcePrefixV6**: Truncate the requestors IPv6 address to this length, in bits (only effective when the backend server has annotation useClientSubnet = true). (Dnsdist default)
- **hpa** (Horizontal Pod Autoscaler - see helm values 'dnsdist.hpa' for examples)
  - **enabled**: Enable or disable the HPA (default: *false*)
  - **minReplicas**: Minimum # of replicas (default: *2*)
  - **maxReplicas**: Maximum # of replicas (default: *4*)
  - **metrics**: Metric(s) upon which to make decisions regarding scaling. Must be an array of MetricSpec (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#metricspec-v2beta2-autoscaling). Examples are in the helm values.
  - **behavior**: Behavior of the HPA, must be instance of HorizontalPodAutoscalerBehavior (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#horizontalpodautoscalerspec-v2beta2-autoscaling).
- **readiness** (Readiness probe Configuration)

  - **do53ProbeInterval**: How often (in seconds) the agent will judge the health state of dnsdist via tests against the Do53 listeners. (default: *2*)
  - **do53QDomain**: Domain used in the query to judge whether the Do53 listeners of a dnsdist container are healthy and ready for traffic. (default: *a.root-servers.net*)
  - **do53QTimeout**: Number of seconds after which the query used to judge whether the Do53 listeners of a dnsdist container are healthy and ready for traffic time out. (default: *1*)
  - **do53QType**: Type of query used to judge whether the Do53 listeners of a dnsdist container are healthy and ready for traffic. (default: *A*)
  - **failureThreshold**: When a probe fails, Kubernetes will try this many times before marking the container as Unready. Updates deployment, resulting in respawn of dnsdist pods (default: *2*)
  - **initialDelaySeconds**: Number of seconds after the containers have started before readiness probes are initiated. Updates deployment, resulting in respawn of dnsdist pods. (default: *5*)
  - **periodSeconds**: How often (in seconds) to perform the readiness probes. Updates deployment, resulting in respawn of dnsdist pods (default: *2*)
  - **stateProbeInterval**: How often (in seconds) the agent will judge the health state of the dnsdist agent. (default: *2*)
  - **successThreshold**: Minimum consecutive succesfull probes before a container is marked Ready. Updates deployment, resulting in respawn of dnsdist pods (default: *2*)
  - **timeoutSeconds**: Number of seconds after which the readiness probes time out. Updates deployment, resulting in respawn of dnsdist pods (default: *1*)
- **resources** (dnsdist resource allocation defaults)
  - **limits** (Limit amounts)
    - **cpu**: Limit amount of CPU (Kubernetes default)
    - **memory**: Limit amount of memory (Kubernetes default)
  - **requests** (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)
- **ringBuffers** (Ringbuffer configuration)
  - **retries**: Number of shards to attempt to lock without blocking before giving up and simply blocking while waiting for the next shard to be available. Defaults to 5 if there is more than 1 shard, else it defaults to 0. (Dnsdist default)
  - **shards**: Number of shards to use to limit lock contention, defaults to 1. (Dnsdist default)
  - **size**: Maximum amount of queries to keep in the ringbuffer, defaults to 10000. (Dnsdist default)
- **securityPolling** (Security polling configuration)

  - **securityPollInterval**: Interval between security pollings, in seconds. Defaults to 3600. (Dnsdist default)
  - **securityPollSuffix**: Domain name from which to query security update notifications. Setting this to an empty string disables secpoll. (Dnsdist default)

- **service** (Configuration of the Kubernetes service object **Note**: See the section 'Exposing dnsdist' for more details and examples on how to configure this)
  - **type**: Type of service (One of NodePort, ClusterIP, LoadBalancer). Defaults to ClusterIP
  - **annotations**: List of annotations to add to the Service
  - **loadBalancerIP**: For a Service of type LoadBalancer, a loadbalancer IP can be requested here. If left empty/undefined, this will be assigned by your LoadBalancer provider
  - **ports**: A dictionary of ports you want to have exposed via the Service
- **tuning** (dnsdist tuning functions)
  - **setCacheCleaningDelay**: Interval in seconds between two runs of the cache cleaning algorithm, removing expired entries. (Dnsdist default)
  - **setCacheCleaningPercentage**: Percentage of the cache that the cache cleaning algorithm will try to free by removing expired entries. By default (100), all expired entries are removed. (Dnsdist default)
  - **setMaxTCPClientThreads**: Maximum amount of TCP client threads, handling TCP connections. By default this value is 10, unless more than 10 TCP listen sockets have been defined. (Dnsdist default)
  - **setMaxTCPConnectionDuration**: Maximum duration of an incoming TCP connection, in seconds. 0 (the default) means unlimited. (Dnsdist default)
  - **setMaxTCPConnectionsPerClient**: Maximum number of TCP connections per client. 0 (the default) means unlimited. (Dnsdist default)
  - **setMaxTCPQueriesPerConnection**: Maximum number of queries in an incoming TCP connection. 0 (the default) means unlimited. (Dnsdist default)
  - **setMaxTCPQueuedConnections**: Maximum number of TCP connections queued (waiting to be picked up by a client thread), defaults to 1000. 0 means unlimited. (Dnsdist default)
  - **setMaxUDPOutstanding**: Maximum number of outstanding UDP queries to a given backend server, defaults to 65535. (Dnsdist default)
  - **setStaleCacheEntriesTTL**: TTL for expired cache entries to be eligible as answer when no backends are available for a query, in seconds. (Dnsdist default)
  - **setTCPRecvTimeout**: Read timeout on TCP connections from the client, in seconds. (Dnsdist default)
  - **setTCPSendTimeout**: Write timeout on TCP connections from the client, in seconds. (Dnsdist default)
  - **setUDPTimeout**: Maximum time dnsdist will wait for a response from a backend over UDP, in seconds. Defaults to 2. (Dnsdist default)

8.1.3 Instance Sets

pools (Configuration of pools)

Key: name (Name of the pool)

- **ecs**: Set to true if dnsdist should add EDNS Client Subnet information to the query before looking up into the cache, when all servers from this pool are down. If at least one server is up, the preference of the selected server is used, this parameter is only useful if all the backends in this pool are down and have EDNS Client Subnet enabled, since the queries in the cache will have been inserted with ECS information. Default is false. (Dnsdist default)
- **serverGroups**: List of [Name] of recursor & resolver instances whose endpoints will be member of this pool (Dnsdist default)
- **serverPolicy**: Policy for dnsdist to use to select the server in this pool to send a query to. Supported policies are 'leastOutstanding', 'firstAvailable', 'wrandom', 'whashed', 'chashed' & 'roundrobin' (Dnsdist default)
- **packetcache** (Configurable cache that holds responses from prior requests served by the pool)
  - **cookieHashing**: Whether EDNS Cookie values will be hashed, resulting in separate entries for different cookies in the packet cache. This is required if the backend is sending answers with EDNS Cookies, otherwise a client might receive an answer with the wrong cookie. (Dnsdist default)
  - **deferrableInsertLock**: Whether the cache should give up insertion if the lock is held by another thread, or simply wait to get the lock. (Dnsdist default)
  - **keepStaleData**: Whether to suspend the removal of expired entries from the cache when there is no backend available in at least one of the pools using this cache. (Dnsdist default)
  - **maxEntries**: Max (Dnsdist default)
  - **maxNegativeTTL**: Cache a NXDomain or NoData answer from the backend for at most this amount of seconds, even if the TTL of the SOA record is higher. (Dnsdist default)
  - **maxTTL**: Cap the TTL for records to his number. (Dnsdist default)
  - **minTTL**: Do not cache entries with a TTL lower than this. (Dnsdist default)
  - **numberOfShards**: Number of shards to divide the cache into, to reduce lock contention. (Dnsdist default)
  - **parseECS**: Whether any EDNS Client Subnet option present in the query should be extracted and stored to be able to detect hash collisions involving queries with the same qname, qtype and qclass but a different incoming ECS value. (Dnsdist default)
  - **staleTTL**: When the backend servers are not reachable, and global configuration setStaleCacheEntriesTTL is set appropriately, TTL that will be used when a stale cache entry is returned. (Dnsdist default)
  - **temporaryFailureTTL**: On a SERVFAIL or REFUSED from the backend, cache for this amount of seconds. (Dnsdist default)

8.2 recursors

Configuration of recursor instances

Key: name (Name of the recursor instance)

8.2.1 Parameters

- **affinity**: Kubernetes pod affinity
- **hostNetwork**: Use host networking for recursor pods
- **nodeSelector**: Kubernetes pod nodeSelector
- **replicas**: Default number of replicas in a recursor deployment (default: *2*)
- **revisionHistoryLimit**: Default 'revisionHistoryLimit' for recursor deployments (default: *0*)

8.2.2 Parameter Sets

- **annotations** (Settings to be applied to each recursor instance when added to dnsdist as a server)
  - **addXPF**: Add the client's IP address and port to the query, along with the original destination address and port. Default is disabled (0)
  - **checkClass**: Number to use as QCLASS in the health-check query, default is DNSClass.IN
  - **checkInterval**: The time in seconds between health checks
  - **checkName**: String to use as QNAME in the health-check query, default is "a.root-servers.net."
  - **checkTimeout**: The timeout (in milliseconds) of a health-check query, default to 1000 (1s)
  - **checkType**: String to use as QTYPE in the health-check query, default is "A"
  - **disableZeroScope**: Disable the EDNS Client Subnet 'zero scope' feature, which does a cache lookup for an answer valid for all subnets (ECS scope of 0) before adding ECS information to the query and doing the regular lookup. This requires the *parseECS* option of the corresponding cache to be set to true
  - **maxCheckFailures**: Allow this amount of check failures before declaring the back-end down, default is 1
  - **mustResolve**: Set to true when the health check MUST return a RCODE different from NXDomain, ServFail and Refused. Default is false, meaning that every RCODE except ServFail is considered valid
  - **order**: The order of servers in this set, used by the *leastOutstanding* and *firstAvailable* policies
  - **qps**: Limit the number of queries per second to this amount, when using the *firstAvailable* policy

  - **reconnectOnUp**: Close and reopen the sockets when a server transits from Down to Up. This helps when an interface is missing when dnsdist is started. Default is false
  - **retries**: The number of TCP connection attempts to servers in this set, for a given query
  - **rise**: Require NUM consecutive successful checks before declaring the backend up, default is 1
  - **setCD**: Set the CD (Checking Disabled) flag in the health-check query, default is false
  - **sockets**: Number of sockets (and thus source ports) used toward the backend server, defaults to 1
  - **tcpConnectTimeout**: The timeout (in seconds) of a TCP connection attempt
  - **tcpFastOpen**: Whether to enable TCP Fast Open
  - **tcpRecvTimeout**: The timeout (in seconds) of a TCP read attempt
  - **tcpSendTimeout**: The timeout (in seconds) of a TCP write attempt
  - **useClientSubnet**: Add the client's IP address in the EDNS Client Subnet option when forwarding the query to this backend
  - **useProxyProtocol**: Add a proxy protocol header to the query, passing along the client's IP address and port along with the original destination address and port. Default is false
  - **weight**: The weight of servers in this set, used by the *wrandom*, *whashed* and *chashed* policies, default is 1

- **config** (Recursor configuration, any configuration item listed in the recursor documentation (https://doc.powerdns.com/recursor/settings.html) can be referenced here. Settings which are explicitly ignored due to conflicts with Cloud Control are filtered by 'webserver*', 'local*', 'disable-syslog', 'daemon', 'cpu-map', 'chroot', 'socket*' and 'config*')
- **hpa** (Horizontal Pod Autoscaler - see helm values 'recursor.hpa' for examples)
  - **enabled**: Enable or disable the HPA (default: *false*)
  - **minReplicas**: Minimum # of replicas (default: *2*)
  - **maxReplicas**: Maximum # of replicas (default: *4*)
  - **metrics**: Metric(s) upon which to make decisions regarding scaling. Must be an array of MetricSpec (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#metricspec-v2beta2-autoscaling). Examples are in the helm values.
  - **behavior**: Behavior of the HPA, must be instance of HorizontalPodAutoscalerBehavior (https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#horizontalpodautoscalerspec-v2beta2-autoscaling).
- **resources** (recursor resource allocation defaults)
  - **limits** (Limit amounts)
    * **cpu**: Limit amount of CPU (Kubernetes default)
    * **memory**: Limit amount of memory (Kubernetes default)

  - **requests** (Request amounts)
    - **cpu**: Request amount of CPU (Kubernetes default)
    - **memory**: Request amount of memory (Kubernetes default)

8.3 resolvers

Configuration of external resolvers

Key: name (Name of the resolver instance)

8.3.1 Parameters

- **ips**: List of IP addresses of resolver endpoints
- **port**: Port to send traffic to for resolver endpoints

8.3.2 Parameter Sets

- **annotations** (Settings to be applied to each resolver instance when added to dnsdist as a server)
  - **addXPF**: Add the client's IP address and port to the query, along with the original destination address and port. Default is disabled (0)
  - **checkClass**: Number to use as QCLASS in the health-check query, default is DNSClass.IN
  - **checkInterval**: The time in seconds between health checks
  - **checkName**: String to use as QNAME in the health-check query, default is "a.root-servers.net."
  - **checkTimeout**: The timeout (in milliseconds) of a health-check query, default to 1000 (1s)
  - **checkType**: String to use as QTYPE in the health-check query, default is "A"
  - **disableZeroScope**: Disable the EDNS Client Subnet 'zero scope' feature, which does a cache lookup for an answer valid for all subnets (ECS scope of 0) before adding ECS information to the query and doing the regular lookup. This requires the *parseECS* option of the corresponding cache to be set to true
  - **maxCheckFailures**: Allow this amount of check failures before declaring the back-end down, default is 1
  - **mustResolve**: Set to true when the health check MUST return a RCODE different from NXDomain, ServFail and Refused. Default is false, meaning that every RCODE except ServFail is considered valid
  - **order**: The order of servers in this set, used by the *leastOutstanding* and *firstAvailable* policies
  - **qps**: Limit the number of queries per second to this amount, when using the *firstAvailable* policy

- **reconnectOnUp**: Close and reopen the sockets when a server transits from Down to Up. This helps when an interface is missing when dnsdist is started. Default is false
- **retries**: The number of TCP connection attempts to servers in this set, for a given query
- **rise**: Require NUM consecutive successful checks before declaring the backend up, default is 1
- **setCD**: Set the CD (Checking Disabled) flag in the health-check query, default is false
- **sockets**: Number of sockets (and thus source ports) used toward the backend server, defaults to 1
- **tcpConnectTimeout**: The timeout (in seconds) of a TCP connection attempt
- **tcpFastOpen**: Whether to enable TCP Fast Open
- **tcpRecvTimeout**: The timeout (in seconds) of a TCP read attempt
- **tcpSendTimeout**: The timeout (in seconds) of a TCP write attempt
- **useClientSubnet**: Add the client's IP address in the EDNS Client Subnet option when forwarding the query to this backend
- **useProxyProtocol**: Add a proxy protocol header to the query, passing along the client's IP address and port along with the original destination address and port. Default is false
- **weight**: The weight of servers in this set, used by the *wrandom*, *whashed* and *chashed* policies, default is 1

8.4 rulesets

Rulesets allow for the configuration of dnsdist Packet Policies. For a detailed example of how to use these rulesets, please refer to the *Getting Started* chapter in the *Overview* guide.

A basic ruleset looks as follows:

```
rulesets:
  tcp-refusal-ruleset:
    group: block-traffic
    type: DNSDistRule
    priority: 100
    rules:
    - name: Refuse TCP
      combinator: AND
      selectors:
        - TCP: true
        - QName: "tcptrue.example.com"
      action:
        RCode:
          rcode: "REFUSED"
```

Where 'tcp-refusal-ruleset' is the name of the ruleset, which will be used to create a uniquely named object. Under this name, the ruleset is defined, using the following values:

- **group**: The value of this parameter can be referenced in a dnsdist configuration to apply the rules from this ruleset to that instance
- **type**: Type of ruleset, currently limited to 'DNSDistRule'
- **priority**: Priority of this ruleset. If multiple are assigned to a dnsdist instance, it will process the rule with the lowest 'priority' value first.
- **rules**: An array of rules to be applied to this ruleset (see below for more details on rules)

8.4.1 Rules

Rules are configurable via 4 different parameters:

- **name**: Name of the rule
- **combinator**: One of 'AND', 'OR' or 'NOT'
- **selectors**: List of filters on which to apply the logic of the combinator
- **action**: Action to apply to the traffic selected by the above selectors

As a result, a single rule will look as follows:

```
name: Refuse TCP
combinator: AND
selectors:
  - TCP: true
  - QName: "tcptrue.example.com"
action:
  RCode:
    rcode: "REFUSED"
```

8.4.2 Combinators

The available combinators function as follows:

- **AND**: If all selectors match, the action will be applied
- **OR**: If any of the selectors match, the action will be applied
- **NOT**: If the selector does not match, the action will be applied (Only 1 selector is allowed when using a NOT combinator)

8.4.3 Selectors

The following selectors are available:

TCP

Format:

```
TCP: true
```

If 'true', this will select queries received over TCP. If 'false' it will select non-TCP traffic (ie: UDP)

MaxQPS

Format:

```
MaxQPS:
  qps: 50
```

Matches traffic not exceeding this qps limit. If e.g. this is set to 50, starting at the 51st query of the current second traffic stops being matched. This can be used to enforce a global QPS limit.

MaxQPSIP

Format:

```
MaxQPSIP:
  qps: 20
  v4Mask: 32
  v6Mask: 64
  burst: 20
  expiration: 300
  cleanupDelay: 60
  scanFraction: 10
```

Explanation of each parameter:

- qps (int) – The number of queries per second allowed, above this number traffic is matched
- v4Mask (int) – The IPv4 netmask to match on. Default is 32 (the whole address)
- v6Mask (int) – The IPv6 netmask to match on. Default is 64
- burst (int) – The number of burstable queries per second allowed. Default is same as qps
- expiration (int) – How long to keep netmask or IP addresses after they have last been seen, in seconds. Default is 300
- cleanupDelay (int) – The number of seconds between two cleanups. Default is 60
- scanFraction (int) – The maximum fraction of the store to scan for expired entries, for example 5 would scan at most 20% of it. Default is 10 so 10%

Since most of the parameters have defaults, you can define a basic MaxQPSIP selector as follows:

```
MaxQPSIP:
  qps: 20
```

NetmaskGroup

Format:

```
NetmaskGroup:
  nmg:
  - "192.0.2.0/28"
  - "2001:db8:1234::/64"
  src: true
  quiet: false
```

Explanation of each parameter:

- nmg (NetMaskGroup) – The NetMaskGroups to match on
- src (bool) – Whether to match source or destination address of the packet. Defaults to true (matches source)
- quiet (bool) – Do not display the list of matched netmasks in Rules. Default is false.

Since most of the parameters have defaults, you can define a basic NetmaskGroup selector as follows:

```
NetmaskGroup:
  nmg:
  - "192.0.2.0/28"
  - "2001:db8:1234::/64"
```

QName

Format:

```
QName: "host.example.com"
```

This selector matches queries with the specified qname exactly.

8.4.4 Actions

The following actions are available:

Pool

Format:

```
Pool:
  poolname: "name-of-a-pool"
```

The Pool action will send the packet into the specified pool

QPS

Format:

```
QPS:
  maxqps: 10
```

Drop a packet if it does exceed the maxqps queries per second limits.

RCode

Format:

```
RCode:
  rcode: "REFUSED"
```

The RCode action will answer any selector queries with the rcode specified.

9 Prometheus

This section allows you to configure 2 forms of Prometheus scraping configurations:

- Via PodMonitor objects (if a Prometheus Operator) is available
- Via Prometheus scraping annotations on Pods

By default, all Pods will have the following annotations present for Prometheus scraping:

- prometheus.io/scrape = whether or not to scrape this endpoint (default: *true*)
- prometheus.io/port = the port to which the metrics endpoint listens (default: *8083*)
- prometheus.io/path = the path where the metrics endpoint serves metrics (default: */metrics*)

To disable this default behavior, you can set the value *prometheus.annotations* to *false* (default: *true*)

If you have a Prometheus Operator available or if you deployed the stack including the *Monitoring Operators* chart, you can set the value *prometheus.operator.available* to *true* (default: *false*). As a result PodMonitor objects will be created for each dnsdist & recursor instance, with all the necessary scraping details automatically included. These pods should then all be discovered automatically by the Prometheus Operator.