How do I add a nameserver to all pods in Google Container Engine [GKE]?

Asked 8 years, 9 months ago Modified 7 years, 8 months ago

Viewed 3k times Part of Google Cloud Collective

9

I am attempting to migrate my on premises cluster to GKE. In order to facilitate this transition I need to be able to resolve the names of legacy services.

Assume that the networking/VPN is a solved problem.

Is there are way to do this with GKE currently?

Effectively I am attempting to add a NS to every /etc/resolv.conf

kubernetes google-kubernetes-engine

Share

Improve this question

Follow

asked Mar 3, 2016 at 21:44

Jake

4,364 ● 2 ● 17 ● 20

3 Answers

Sorted by: Highest score (default)

3

I want to add to what Eric said, and mutate it a bit.

One of the realizations we had during the kubernetes 1.1 "settling period" is that there are not really specs for things like resolv.conf and resolver behavior. Different resolver libraries do different things, and this was causing pain for our users.

Specifically, some common resolvers assume that all `nameserver`s are fungible and would break if you had nameservers that handled different parts of the DNS namespace. We made a decision that for kube 1.2 we will NOT pass multiple `nameserver` lines into containers. Instead, we pass only the kube-dns server, which handles `cluster.local` queries and forwards any other queries to an "upstream" nameserver.

How do we know what "upstream" is? We use the `nameservers` of the node. There is a per-pod dnsPolicy field that governs this choice. The net result is that containers see a single `nameserver` in resolv.conf, which we own, and that nameserver handles the whole DNS namespace.

What this practically means is that there's not a great hook for you to interject your own nameserver. You could change the `--cluster-dns` flag to kubelets to point to your own DNS server, which would then forward to the kube-dns, which would then forward to "upstream". The problem is that GKE doesn't really support changing flags that way. If/when the node is updated, the flag will disappear in favor of the default.

Possible solutions:

- Have kubelets read their flags from an in-cluster config. This is already plan of record, but is not in v1.2
- Have kube-dns take a flag indicating what "upstream" is. Kube-dns is a "cluster addon" and as such isn't really mutable by end users (we will update it with your cluster and lose your changes).
- Have kube-dns read its flags from an in-cluster config, and take a flag indicating what "upstream" is. This is a doable idea, but probably not for v1.2 (too late). It *might* be possible to patch this into a v1.2.x but it's not really a bugfix, it's a feature.
- Get your own DNS server into the resolv.conf on each node so that kube-dns would use you as upstream. I don't think GKE has a way to configure this that won't also get lost on node upgrades. You could write a controller that periodically SSH'ed to VMs and wrote that out, and subsequently checked your kube-dns container for correctness. Blech.

I think the right answer is to use in-cluster configmaps to inform either kubelets or DNS (or both). If you think these might be workable answers (despite the timeframe issues), it would be great if you opened a GitHub issue to discuss. It will get more visibility there.

Share Improve this answer Follow

answered Mar 4, 2016 at 20:11

Tim Hockin
3,663 ● 14 ● 19

1

Effectively No.

If you modify the node's resolv.conf the pods will inherit the changes.

However, glibc prohibits using more than 3 nameservers or more than 6 search records.

GCE VMs use 2 nameservers and 3 searches for accessing node metadata and project networking. And GKE uses 1 nameserver and 3 searches. That leaves you 0 nameservers and 0 searches.

See this issue: https://github.com/kubernetes/kubernetes/issues/9079 and this issue: https://github.com/kubernetes/kubernetes/issues/9132

Share Improve this answer Follow

answered Mar 4, 2016 at 19:08

Eric Tune
8,228 ● 1 ● 18 ● 19

However, maybe you could copy all your legacy service DNS records into Google Cloud DNS. (cloud.google.com/dns/what-is-cloud-dns). That would make them resolvable on all your GCP project instances, and in all GKE containers. – Eric Tune Mar 4, 2016 at 19:12

yes but what if the OP uses an internal dns and not public dns records – nelasx Oct 3, 2016 at 19:56

1

I solved this by setting up a dnsmasq service in the k8s cluster and point all pods nameserver except dnsmasq to the dnsmasq service.
dnsmasq will forward requests to the correct nameserver based on domain suffix. So both internal and external vpn lookups will work.

1. setup a dnsmasq service.
   The pods can look something like this, make sure this is has at least 2 pods as it needs to be HA.

   ```
   apiVersion: v1
   kind: Pod
   metadata:
     name: dnsmasq
   spec:
     containers:
     - name: dnsmasq
       image: "andyshinn/dnsmasq:2.76"
       ports:
       - containerPort: 53
         hostPort: 53
         protocol: UDP
       - containerPort: 53
         hostPort: 53
         protocol: TCP
       args: [
   ```

```
        "-S", "/consul/10.3.20.86",
        "-S", "/consul/10.3.20.88",
        "-S", "/consul/10.3.20.90",
        "-S", "/your-vpn-domain.dom/10.3.128.22",
        "-S", "/your-vpn-domain.dom/10.3.128.23"
      ]
      securityContext:
        capabilities:
          add:
          - NET_ADMIN
```

2. Add a resolv-conf config map.

```
#!/bin/bash

DNS_IP=$(kubectl get svc --template '{{.spec.clust
DNS_POD=$(kubectl get pod -n kube-system | grep -v
^kube-dns  | head -1 | awk '{ print $1; }')
DOMAIN=$(kubectl describe -n kube-system pod/${DNS
sed -Ee 's/.*--domain=(.*)\..*/\1/')
SEARCH=$(kubectl exec -n kube-system  ${DNS_POD} -
/etc/resolv.conf | grep ^search | cut -d' '  -f2-)
VPN_SEARCH="your-vpn-domain.dom"

kubectl create -f - <<EOF
apiVersion: v1
kind: ConfigMap
metadata:
  name: resolv-conf
data:
  resolv.conf: |
    # This file is created by resolv-conf config m
service.
    search default.svc.${DOMAIN} svc.${DOMAIN} ${D
${VPN_SEARCH}
    nameserver ${DNS_IP}
    ndots:5
EOF
```

3. Mount the cfgmap in your services/pods. add this to your pods

```
  volumeMounts:
  - mountPath: /etc/resolv.conf
    name: resolv-conf
    subPath: resolv.conf
    readOnly: true
volumes:
  - name: resolv-conf
    configMap:
      name: resolv-conf
```

This solution can perhaps can be considered a bit ugly, but currently there aren't many other options. In the future I would hope to see a dns forward feature to Google Cloud or kube-dns.

It's kind of crazy that Google Cloud doesn't offer a DNS forward feature for specified domains/zones.

Share Improve this answer Follow

answered Apr 18, 2017 at 9:43
Raboo
145 ● 1 ● 11

---

2 Does this help blog.kubernetes.io/2017/04/…? – Kuberchaun Apr 18, 2017 at 22:28

Looks exactly like what the OP and I need. I'm gonna test it on GKE and report back after. – Raboo Apr 19, 2017 at 10:59

Thanks for the neat solution Raboo. I like how that fits well into my kubernetes workflow and doesn't require config management on my nodes. Thanks GoatWalker for the link. I'm glad it's baked in now. – Jake Apr 21, 2017 at 18:15