coreDNS问题

Subtitle

2022/10/03

Table of Contents

coreDNS问题

dnsmasq监听tunl0问题

tunl0未启动时（机器刚重启时可能会这样），会导致dnsmasq无法启动，可能造成域名无法解析.

考虑取消监听tunl0机制，tunl0未启动时，/etc/resolv.conf 会写入空的 nameserver ，还会导致dnsmasq无法启动。既然支持/custom/resolv.conf 这种形式，那么就可以和 dnsmasq 没有任何关系，initContainer中 将 /etc/resolv.conf 中 的 127.0.0.1 及 172 网段ip删除即可.

取消 tunl0 步骤：

1. 重制 coredns 镜像，修正 /custom/resolv.conf，不包含dnsmasq相关地址
2. 更新coredns，检查是否正常
3. dnsmasq取消监听tunl0（直接使用puppet base模块中的处理）

proxy问题

开启coreDNS log插件，可以看到以下日志：

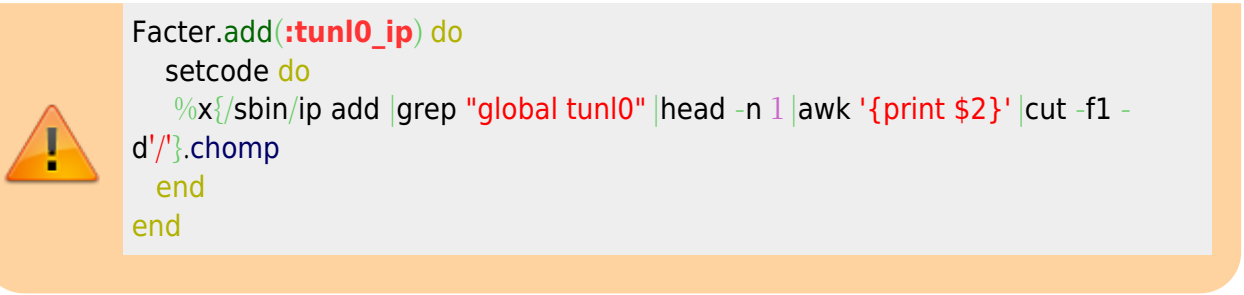
```
2018/10/29 15:23:30 [ERROR] 2 10142s.redis.cloud.com. A: unreachable backend: read udp
127.0.0.1:48803-&gt;127.0.0.1:53: i/o timeout
2018/10/29 15:23:30 [ERROR] 2 10142s.redis.cloud.com. AAAA: unreachable backend: read udp
127.0.0.1:46454-&gt;127.0.0.1:53: i/o timeout
2018/10/29 15:23:30 [ERROR] 2 internal.scloud.cloud.com. AAAA: unreachable backend: read udp
127.0.0.1:52533-&gt;127.0.0.1:53: i/o timeout
2018/10/29 15:23:30 [ERROR] 2 internal.scloud.cloud.com. AAAA: unreachable backend: read udp
127.0.0.1:40510-
```

因为 proxy配置为 . /etc/resolv.conf ，而node上使用了dnsmasq，监听127.0.0.1，/etc/resolv.conf中第一行即 127.0.0.1。因此这里需要dnsmasq监听tunl0 IP，然后让coreDNS直接 proxy tunl0 IP（**如何将tunl0 IP传入环境变量？**）

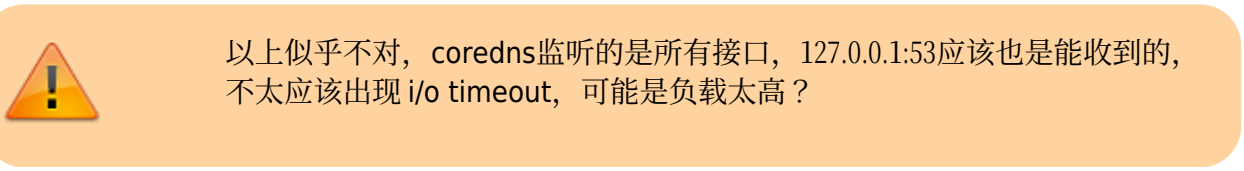
Update

考虑到dnsPolicy为Default的情况，pod会继承node的/etc/resolv.conf，非hostNetwork的pod将无法使用127.0.0.1作为dns服务器，因此考虑node上的/etc/resolv.conf使用tunl0 IP替换127.0.0.1，这样一来，以下coredns方案可以放弃了（**也可以继续使用，只会在/custom/resolv.conf中加入重复的tunl0 IP，应该不会有影响**），直接使用/etc/resolv.conf就可以了。参考.dnsPolicy问题

puppet 添加自定义变量取tunl0值



```
Facter.add(:tunl0_ip) do
  setcode do
    %x{/sbin/ip add |grep "global tunl0" |head -n 1 |awk '{print $2}' |cut -f1 -d'/'}.chomp
  end
end
```



以上似乎不对，coredns监听的是所有接口，127.0.0.1:53应该也是能收到的，不太应该出现 i/o timeout，可能是负载太高？

initContainer方式

使用initContainer生成custom resolv.conf

```
--- a/kubernetes/files/etc/kubernetes/manifests/coredns/coredns.yaml
+++ b/kubernetes/files/etc/kubernetes/manifests/coredns/coredns.yaml
@@ -57,6 +57,7 @@ data:
   Corefile: |
     .:53 {
         errors
+        log
         health
         kubernetes cluster.local in-addr.arpa ip6.arpa {
             pods insecure
@@ -64,7 +65,7 @@ data:
             fallthrough in-addr.arpa ip6.arpa
         }
         prometheus :9153
-        proxy . /etc/resolv.conf
+        proxy . /custom/resolv.conf
         cache 30
         reload
     }
@@ -104,6 +105,18 @@ spec:
         effect: NoSchedule
       - key: "CriticalAddonsOnly"
         operator: "Exists"
+      initContainers:
+      - name: resolv
+        image: alpine:3.8
+        imagePullPolicy: IfNotPresent
+        command:
+        - sh
+        - -c
+      # 使用calico时pod default gw全部为169.254.1.1，因此不能用route命令取tunl0地址。使用flannel时可以用route
+        - "echo $MY_POD_IP |awk -F '.' '{print \"nameserver \"$1\".\"$2\".\"$3\".1\"}' |head -n 1 >
```

```
/custom/resolv.conf; grep '^nameserver ' /etc/resolv.conf |grep -v '127.0.0.1' >>
/custom/resolv.conf"
+       env:
+       - name: MY_POD_IP
+         valueFrom:
+           fieldRef:
+             fieldPath: status.podIP
+       volumeMounts:
+       - name: resolv
+         mountPath: /custom
+
      containers:
      - name: coredns
        image: coredns:1.1.3
@@ -119,6 +132,9 @@ spec:
        - name: config-volume
          mountPath: /etc/coredns
          readOnly: true
+       - name: resolv
+         mountPath: /custom
+         readOnly: true
        ports:
        - containerPort: 53
          name: dns
@@ -148,6 +164,8 @@ spec:
          readOnlyRootFilesystem: true
      dnsPolicy: Default
      volumes:
+       - name: resolv
+         emptyDir: {}
        - name: config-volume
          configMap:
            name: coredns
```

loop插件

以上问题 或者使用coreDNS loop插件（https://coredns.io/plugins/loop/）解决。

为了不对localdns造成太大负担，使用dnsmasq可能是更好的方案

dnsmasq配置

```
# 指定IP地址，可以多次指定。
# interface 选项和 listen-address 选项可以同时使用。
# 下面两行与指定 interface 选项的作用类似。
listen-address=127.0.0.1
# 指定接口，指定后同时附加 lo 接口，可以使用'*'通配符。
# 不能使用接口别名（例如："eth1:0"），请用 listen-address 选项替代。
#interface=wlp2s0
interface=tunl0
# 通常情况下即使设置了 interface 选项（例如：interface=wlp2s0 
```

```
# 将仍然绑定到通配符地址（例如：*:53 ）。
# 开启此项将仅监听指定的接口。
# 适用于在同一主机的不同接口或 IP 地址上运行多个 dns 服务器。
bind-interfaces
```

参考: http://blog.51cto.com/laoding/1883469

禁用AAAA查询

- 容器不发起AAAA Query ?
- coreDNS禁用AAAA ?

参考

- https://mp.weixin.qq.com/s/z7PaTk7pu5ZMyXYbg9lY-Q

~~AAAA解析是相当快的，如果域名没有做AAAA记录，也不会延迟。关键问题应该是让容器不发起AAAA查询。~~（如果查询域名的权威dns不在本地，可能会很慢（家里测试很慢，但是IDC机房的机器没感觉慢），可能是需要向上游dns查询结果，本地虚拟机用 time dig AAAA baidu.com普遍1s以上，因此非常有必要禁用AAAA查询）

- https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/#pods-dns-config
- option single-request
  - https://unix.stackexchange.com/questions/373881/how-to-disable-dns-aaaa-queries-on-an-ipv4-only-host

single-request是不并行发起A和AAAA查询，用于某些不能正确处理并行查询的dns服务器，coredns没有此问题，不需要加此选项。此选项也没有说不发起AAAA查询，只是不并行了

```
single-request (since glibc 2.10)
    Sets RES_SNGLKUP in _res.options.  By default, glibc
    performs IPv4 and IPv6 lookups in parallel since
    version 2.9.  Some appliance DNS servers cannot handle
    these queries properly and make the requests time out.
    This option disables the behavior and makes glibc
    perform the IPv6 and IPv4 requests sequentially (at the
    cost of some slowdown of the resolving process).
```

https://community.hpe.com/t5/Networking/Can-I-disable-IPv6-lookup-when-querying-DNS/m-p/4714807#M46999

IPv6 is embedded in very unexpected ways, haphazard is my take on it. I recently had to eliminate all IPv6 client and server responses for an (probably misguided) auditor. If you're running 11.11, you can probably remove the IPv6 patches and products with success.

nsswitch.conf MUST have:

ipnodes: files

to avoid DNS issues. Some DNS servers do not support IPv6 and some are just not configured for IPv6 support, all causing DNS resolution delays.

But for 11.23 and 11.31, I discovered that getipnodebyname, getipnodebyaddr, gethostint, getaddrinfo all have IPv6 embedded with nothing to turn that 'feature' off.

icmp will try IPv6 and if fails, IPv4 only when there is a DNS server declared in resolv.conf.

查询过多

k8s生成的/etc/resolv.conf内容如下

```
nameserver 169.169.0.2
search intra.svc.cluster.local svc.cluster.local cluster.local
options ndots:5
```

其中ndots含义是查询域名中如果点的数量少于5，那么加上search中定义的后缀去查询

search Search list for host-name lookup.

The search list is normally determined from the local domain name; by default, it contains only the local domain name. This may be changed by listing the desired domain search path following the search keyword with spaces or tabs separating the names. Resolver queries having fewer than ndots dots (default is 1) in them will be attempted using each component of the search path in turn until a match is found. For environments with multiple subdomains please read options ndots:n below to avoid man-in-the-middle attacks and unnecessary traffic for the root-dns-servers. Note that this process may be slow and will generate a lot of network traffic if the servers for the listed domains are not local, and that queries will time out if no server is available for one of the domains. The search list is currently limited to six domains with a total of 256 characters.

coredns日志中有大量 外部域名被加上后缀 后查询失败的记录

kubernetes issue：https://github.com/kubernetes/kubernetes/issues/14051#issuecomment-140813186

水平扩容

有专门的dns水平扩容文档，需要部署其他容器来做dns扩容

（https://kubernetes.io/docs/tasks/administer-cluster/dns-horizontal-autoscaling/），为什么不直接使用HPA?

dns扩容用的是 https://github.com/kubernetes-incubator/cluster-proportional-autoscaler 这个项目，查看介绍，大意是coredns数量根据集群大小来计算，比如增加了节点，那么相应的coredns replicas也要成比例增加。HPA一般基于cpu使用量。

感觉HPA基本就够用了

- https://my.oschina.net/jxcdwangtao/blog/1581879
- https://github.com/coredns/deployment/issues/98

压力测试

缓存

域名解析方式

此问题为云计算的sdns系统bug导致，非coredns问题

域名cname到一个做了泛解析的域名上，这种方式是否会存在问题（dig时间歇性的answer section中不包含A记录）？

```
*.b.com -> A -> 10.0.0.1

a.c.cn -> CNAME -> random.b.com
```

未查出问题，改为cname到一个有明确解析的域名上了。

- https://github.com/coredns/coredns/issues/1864

Printed on: 2022/10/03 08:57