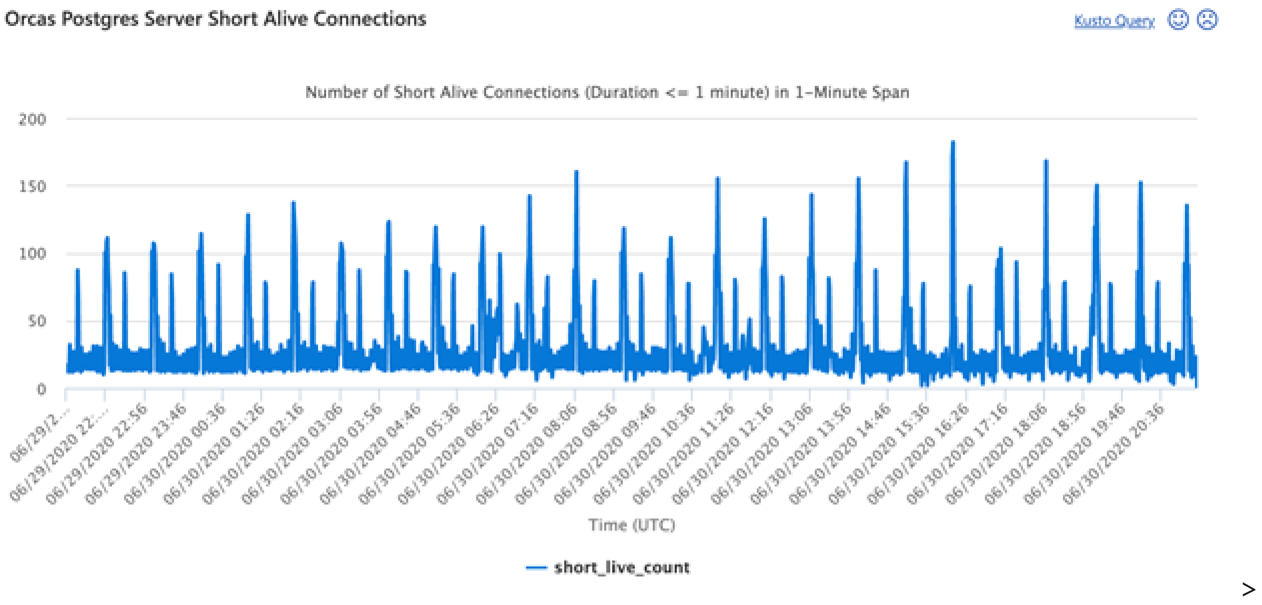
Too many short lived database connections

Last updated by | Lisa Liu | Nov 6, 2020 at 10:34 AM PST

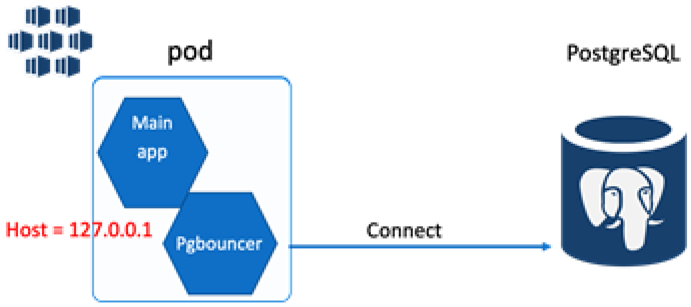
Problem

Opening a new connection to the database is a time-consuming operation. If you find many the short alive connections at ASC (Connection pooling section) like this, it indicates the application is making a new connection upon every request which would add extra latency to the transaction.

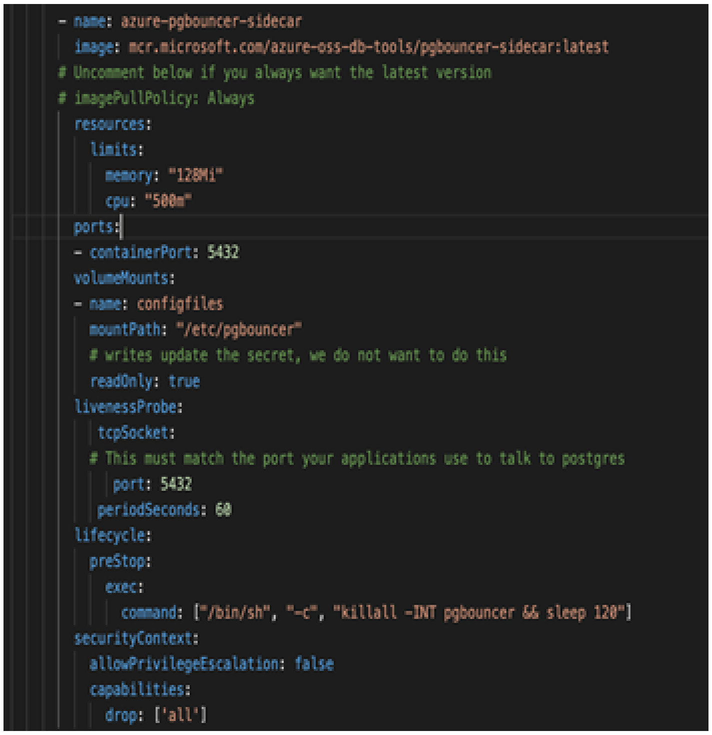


Solution

Leverage connection pooling such as ProxySQL or PgBouncer. In AKS, you can deploy these connection pooling libraries as a sidecar container within the same pod as the main application so the communication would be within the localhost. The only change required in the main application is to replace the database host with ‘127.0.0.1’.



This is an example of deployment manifest for PgBouncer.



```
- name: azure-pgbouncer-sidecar
  image: mcr.microsoft.com/azure-oss-db-tools/pgbouncer-sidecar:latest
# Uncomment below if you always want the latest version
# imagePullPolicy: Always
  resources:
    limits:
      memory: "128Mi"
      cpu: "500m"
  ports:
  - containerPort: 5432
  volumeMounts:
  - name: configfiles
    mountPath: "/etc/pgbouncer"
    # writes update the secret, we do not want to do this
    readOnly: true
  livenessProbe:
     tcpSocket:
    # This must match the port your applications use to talk to postgres
       port: 5432
     periodSeconds: 60
  lifecycle:
    preStop:
      exec:
        command: ["/bin/sh", "-c", "killall -INT pgbouncer && sleep 120"]
  securityContext:
    allowPrivilegeEscalation: false
    capabilities:
      drop: ['all']
```

This article has more details.

How good have you found this content?