PostgreSQL 網絡延遲 瓶頸定量分析

yq.aliyun.com/articles/35176

摘要： 在使用sysbench或者pgbench測試數據庫性能時，連unix socket, loop address性能差異是非常大的，特別是非常小的事務，例如基於KEY的查詢，或者select 1這樣的簡單查詢。原因是這種查詢在數據庫端的處理非常快，從而網絡延遲在整個耗時佔比上就會比較大。還有一種場景.

在使用sysbench或者pgbench測試數據庫性能時，連unix socket, loop address性能差異是非常大的，特別是非常小的事務，例如基於KEY的查詢，或者select 1這樣的簡單查詢。
原因是這種查詢在數據庫端的處理非常快，從而網絡延遲在整個耗時佔比上就會比較大。
還有一種場景結果集比較大，網絡延遲在整個耗時佔比上也會比較大。
那麼如何來定量分析呢？
.1. 分析包的大小，可以通過tcpdump抓包，取得數據庫請求過程中傳輸的包大小和數量。
.2. 或者從PostgreSQL源碼中，根據實際的查詢計算對應的包大小，例如libpq。

以select 1;為例分析。

如何計算出一個請求在數據庫處理的耗時，以及在網絡傳輸段的耗時？

網絡傳輸端的耗時，可以通過前面拿到的包大小，然後使用網絡延遲分析工具例如 qperf 得到。
例子：
假設通過tcpdump得到的 select 1; 的包大小為16字節（去除TCP包頭）。
使用qperf測試單會話下的16字節包tcp延遲。

啟動qperf服務端

```
yum install -y qperf

qperf -lp 8888 &
```

測試迴環地址8-16字節 TCP包延遲

```
qperf 127.0.0.1 -lp 8888 -t 6 -oo msg_size:8:64:*2 -v tcp_lat &
latency        =  5.16 us
    msg_rate       =   194 K/sec
    msg_size       =    16 bytes
    time           =     6 sec
    loc_cpus_used  =  86.8 % cpus
    rem_cpus_used  =  86.8 % cpus
```

測試select 1的qps

```
vi test.sql
select 1;

pgbench -M prepared -n -r -P 1 -f ./test.sql -c 1 -j 1 -T 10 -h 127.0.0.1
tps = 36938.282331 (including connections establishing)
```

根據以上QPS以及qperf測出來的網絡延遲，計算出select 1的數據庫處理時延，注意需要計算回包的時間，所以TCP延遲*2

```
(1000000/36938.03) - (5.16*2) = 16.75 us
```

同局域網主機的延遲測試

```
qperf xxx.xxx.xxx.xxx -lp 8888 -t 6 -oo msg_size:8:64:*2 -v tcp_lat &
tcp_lat:
    latency        =  13.6 us
    msg_rate       =  73.8 K/sec
    msg_size       =    16 bytes
    time           =     6 sec
    loc_cpus_used  =  8.67 % cpus
    rem_cpus_used  =   9.5 % cpus
```

使用以上值，以及前面得到的數據庫處理延遲，計算理論上在這台機器連接到數據庫服務器進行tps測試的結果應該是

```
1000000 / (16.75 + 13.6*2) = 22753  tps
```

實際測試得到的tps, 基本一致

```
pgbench -M prepared -n -r -P 1 -f ./test.sql -c 1 -j 1 -T 10 -h xxx.xxx.xxx.xxx
tps = 24724.666217 (including connections establishing)
```

並發情況下的網絡延遲分析。
啟動多個qperf服務端

```
qperf -lp 8888 &
qperf -lp 8889 &
qperf -lp 8890 &
qperf -lp 8891 &
qperf -lp 8892 &
qperf -lp 8893 &
qperf -lp 8894 &
qperf -lp 8895 &
qperf -lp 8896 &
qperf -lp 8897 &
qperf -lp 8898 &
qperf -lp 8899 &
qperf -lp 8900 &
qperf -lp 8901 &
qperf -lp 8902 &
qperf -lp 8903 &
qperf -lp 8904 &
qperf -lp 8905 &
qperf -lp 8906 &
qperf -lp 8907 &
qperf -lp 8908 &
qperf -lp 8909 &
qperf -lp 8910 &
qperf -lp 8911 &
qperf -lp 8912 &
qperf -lp 8913 &
qperf -lp 8914 &
qperf -lp 8915 &
```

```
qperf -lp 8916 &
qperf -lp 8917 &
qperf -lp 8918 &
qperf -lp 8919 &
qperf -lp 8920 &
qperf -lp 8921 &
qperf -lp 8922 &
qperf -lp 8923 &
qperf -lp 8924 &
qperf -lp 8925 &
qperf -lp 8926 &
qperf -lp 8927 &
qperf -lp 8928 &
qperf -lp 8929 &
qperf -lp 8930 &
qperf -lp 8931 &
qperf -lp 8932 &
qperf -lp 8933 &
qperf -lp 8934 &
qperf -lp 8935 &
qperf -lp 8936 &
qperf -lp 8937 &
qperf -lp 8938 &
qperf -lp 8939 &
qperf -lp 8940 &
qperf -lp 8941 &
qperf -lp 8942 &
qperf -lp 8943 &
qperf -lp 8944 &
qperf -lp 8945 &
qperf -lp 8946 &
qperf -lp 8947 &
qperf -lp 8948 &
qperf -lp 8949 &
qperf -lp 8950 &
qperf -lp 8951 &
```

並發測試延遲

```
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8888 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8889 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8890 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8891 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8892 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8893 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8894 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8895 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8896 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8897 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8898 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8899 tcp_lat &
```

```
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8900 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8901 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8902 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8903 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8904 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8905 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8906 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8907 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8908 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8909 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8910 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8911 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8912 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8913 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8914 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8915 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8916 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8917 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8918 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8919 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8920 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8921 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8922 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8923 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8924 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8925 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8926 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8927 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8928 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8929 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8930 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8931 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8932 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8933 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8934 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8935 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8936 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8937 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8938 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8939 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8940 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8941 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8942 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8943 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8944 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8945 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8946 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8947 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8948 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8949 tcp_lat &
```

```
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8950 tcp_lat &
qperf 127.0.0.1 -t 6 -oo msg_size:16:16:*2 -v -lp 8951 tcp_lat &
```

迴環地址，64個並發的延遲約11.8us

```
latency        =   11.8 us
```

測試64個並發的tps，並計算出數據庫端的耗時(64核的機器)

```
pgbench -M prepared -n -r -P 1 -f ./test.sql -c 64 -j 64 -T 10 -h 127.0.0.1

tps = 1575989.161924 (including connections establishing)
```

計算出數據庫的RT
與單進程的RT基本一致，說明現在PostgreSQL在高並發下的處理能力已經非常強大了，充分利用了CPU的多核，並且性能是線性的。

```
(1000000/(1575989.161924/64)) - (11.8*2) = 17 us
```

在遠端主機測試網絡延遲
從測試結果來看，已經大大超出了數據庫本地處理的時間，網絡成了最大的瓶頸

```
qperf xxx.xxx.xxx.xxx -t 6 -oo msg_size:16:16:*2 -v -lp 8888 tcp_lat &
...
qperf xxx.xxx.xxx.xxx -t 6 -oo msg_size:16:16:*2 -v -lp 8951 tcp_lat &

latency        =  61.8 us
```

推算出64並發的TPS

```
(1000000/(17 + 61.8*2)) * 64 = 455192
```

推算出來的TPS與實際測出來的TPS基本一致

```
pgbench -M prepared -n -r -P 1 -f ./test.sql -c 64 -j 64 -T 10 -h xxx.xxx.xxx.xxx
tps = 466737.781999 (including connections establishing)
```

以上是網絡延遲的定量分析，網絡延遲在高並發的數據庫應用中，影響還是非常大的。

參考
man tcpdump
man qperf
http://blog.yufeng.info/archives/2234