用 sysbench 对 TiDB 进行性能测试的介绍

笔记本：	rds-test		
创建时间：	2018/3/2 9:01	更新时间：	2018/3/2 16:15
作者：	jianghj@gmail.com		
URL：	https://github.com/akopytov/sysbench		

sysbench是基于LuaJIT的脚步化多线程的基准测试工具，最常用于数据库基准测试，但也可用于创建不涉及数据库服务器的任意复杂的工作负载。
sysbench内置了一些测试cpu、文件系统、内存、线程调度器、POSIX mutex等等的 benchmark，可以直接调用进行测试。还有一些可定制的lua脚本，比如完成oltp测试的lua脚本就是执行一些SQL语句，完成点查、范围查、排序查、SELECT DISTINCT、更新、删除、插入等一系列组合的测试。

特点：

- 关于比率和响应时间有广泛的统计数据可用，包括响应时间百分比和直方图;
- 即使有数千个并发线程，开销也很低。sysbench能够每秒生成和跟踪数以亿计的事件（即请求）;
- 通过在用户提供的Lua脚本中实现预定义的钩子，可以轻松创建新的基准;
- 可以作为一个通用的Lua解释器，只需在脚本中用 #!/usr/bin/sysbench 替换 #!/usr/bin/lua。

默认自带下列基准测试：

- oltp_*.lua: a collection of OLTP-like database benchmarks
- fileio: a filesystem-level benchmark
- cpu: a simple CPU benchmark
- memory: a memory access benchmark
- threads: a thread-based scheduler benchmark
- mutex: a POSIX mutex benchmark

响应时间直方图的例子：

```
Threads started!

[ 10s ] thds: 256 tps: 1137.91 qps: 23012.82 (r/w/o: 16150.14/4562.45/2300.22) lat (ms,95%): 287.38 err/s:
0.00 reconn/s: 0.00
[ 20s ] thds: 256 tps: 1202.42 qps: 24049.33 (r/w/o: 16838.93/4806.07/2404.33) lat (ms,95%): 282.25 err/s:
0.00 reconn/s: 0.00
[ 30s ] thds: 256 tps: 1221.92 qps: 24430.12 (r/w/o: 17096.70/4889.48/2443.94) lat (ms,95%): 277.21 err/s:
0.00 reconn/s: 0.00
Latency histogram (values are in milliseconds)
       value  ------------- distribution ------------- count
      54.828 |                                         1
      63.323 |                                         1
      65.645 |                                         1
      68.053 |                                         2
      70.548 |                                         2
      71.830 |                                         3
      75.817 |                                         2
      77.194 |                                         1
      78.597 |                                         5
      80.025 |                                         3
      81.479 |                                         3
      82.959 |                                         3
      84.467 |                                         6
      86.002 |                                         3
      87.564 |                                         10
      89.155 |                                         7
      90.775 |                                         13
      92.424 |                                         14
      94.104 |                                         15
      95.814 |                                         12
      97.555 |                                         13
      99.327 |                                         13
     101.132 |                                         16
```

```
102.969 |                                        12
104.840 |*                                       24
106.745 |*                                       28
108.685 |*                                       32
110.659 |*                                       39
112.670 |*                                       52
114.717 |*                                       47
116.802 |**                                      78
118.924 |**                                      74
121.085 |**                                      67
123.285 |**                                      83
125.525 |***                                     96
127.805 |****                                    134
130.128 |****                                    140
132.492 |****                                    146
134.899 |*****                                   181
137.350 |******                                  196
139.846 |*******                                 235
142.387 |******                                  224
144.974 |*******                                 241
147.608 |********                                276
150.290 |*********                               325
153.021 |**********                              405
155.801 |**********                              377
158.632 |***********                             431
161.514 |***********                             429
164.449 |**************                          533
167.437 |**************                          554
170.479 |**************                          567
173.577 |*****************                       657
176.731 |*******************                     727
179.942 |**********************                  802
183.211 |*******************                     758
186.540 |**********************                  832
189.929 |***************************             1009
193.380 |***************************             1011
196.894 |**************************              965
200.472 |*****************************           1086
204.114 |*******************************         1164
207.823 |*******************************         1194
211.599 |*********************************       1270
215.443 |*********************************       1331
219.358 |*********************************       1316
223.344 |*********************************       1366
227.402 |****************************************1425
231.534 |*********************************       1292
235.740 |********************************        1260
240.024 |********************************        1251
244.385 |*****************************           1095
248.825 |*****************************           1070
253.346 |*****************************           1074
257.950 |************************                906
262.636 |***********************                 856
267.408 |*******************                     755
272.267 |**************                          574
277.214 |**************                          552
282.251 |***********                             441
287.379 |*********                               342
292.601 |*******                                 254
297.917 |*******                                 253
303.330 |*****                                   161
308.842 |***                                     114
314.453 |***                                     100
320.167 |**                                      65
325.984 |**                                      54
331.907 |*                                       34
337.938 |*                                       31
344.078 |*                                       47
350.330 |*                                       32
356.695 |*                                       27
```

```
     363.176 |*                                        22
     369.775 |*                                        20
     376.494 |*                                        23
     383.334 |                                         12
     390.299 |                                         8
     397.391 |                                         6
     404.611 |                                         11
     411.963 |                                         4
     419.448 |                                         5
     427.069 |                                         4
     434.829 |                                         1
     442.730 |                                         1
     458.964 |                                         1
    1129.239 |                                         1
    1149.757 |                                         1
    1170.648 |                                         5
    1191.918 |                                         3
    1213.575 |                                         4
    1235.625 |                                         6
    1258.076 |                                         10
    1304.208 |                                         3
    1327.905 |                                         1
    5312.729 |                                         1
```

sysbench默认带了如下数据库测试脚本：

```
[tidb@rds1 sysbench]$ ll /usr/share/sysbench/
总用量 64
-rwxr-xr-x. 1 root root  1452 1月  17 19:08 bulk_insert.lua
-rw-r--r--. 1 root root 13816 1月  17 19:08 oltp_common.lua
-rwxr-xr-x. 1 root root  1290 1月  17 19:08 oltp_delete.lua
-rwxr-xr-x. 1 root root  2415 1月  17 19:08 oltp_insert.lua
-rwxr-xr-x. 1 root root  1265 1月  17 19:08 oltp_point_select.lua
-rwxr-xr-x. 1 root root  1649 1月  17 19:08 oltp_read_only.lua
-rwxr-xr-x. 1 root root  1824 1月  17 19:08 oltp_read_write.lua
-rwxr-xr-x. 1 root root  1118 1月  17 19:08 oltp_update_index.lua
-rwxr-xr-x. 1 root root  1127 1月  17 19:08 oltp_update_non_index.lua
-rwxr-xr-x. 1 root root  1440 1月  17 19:08 oltp_write_only.lua
-rwxr-xr-x. 1 root root  1919 1月  17 19:08 select_random_points.lua
-rwxr-xr-x. 1 root root  2118 1月  17 19:08 select_random_ranges.lua
drwxr-xr-x. 4 root root  4096 2月   8 17:48 tests
```

可以直接调用脚本对数据库进行压测：

```
$ sysbench /usr/share/sysbench/oltp_read_only.lua \
--mysql-host=127.0.0.1 \
--mysql-port=3306 \
--mysql-user=root \
--mysql-password='000000' \
--mysql-db=sbtest \
--db-driver=mysql \
--tables=10 \
--table-size=1000000 \
--report-interval=10 \
--threads=128 \
--time=120 \
run
```

一个30秒的oltp测试结果展示：

```
Initializing worker threads...

Threads started!

[ 10s ] thds: 256 tps: 1137.91 qps: 23012.82 (r/w/o: 16150.14/4562.45/2300.22) lat (ms,95%): 287.38 err/s:
0.00 reconn/s: 0.00
```

```
[ 20s ] thds: 256 tps: 1202.42 qps: 24049.33 (r/w/o: 16838.93/4806.07/2404.33) lat (ms,95%): 282.25 err/s:
0.00 reconn/s: 0.00
[ 30s ] thds: 256 tps: 1221.92 qps: 24430.12 (r/w/o: 17096.70/4889.48/2443.94) lat (ms,95%): 277.21 err/s:
0.00 reconn/s: 0.00

SQL statistics:
    queries performed:
        read:                            502334
        write:                           143524
        other:                           71762
        total:                           717620
    transactions:                        35881  (1128.64 per sec.)
    queries:                             717620 (22572.85 per sec.)
    ignored errors:                      0      (0.00 per sec.)
    reconnects:                          0      (0.00 per sec.)

General statistics:
    total time:                          31.7877s
    total number of events:              35881

Latency (ms):
         min:                                    54.55
         avg:                                   214.67
         max:                                  5344.86
         95th percentile:                       282.25
         sum:                                7702745.81

Threads fairness:
    events (avg/stddev):           140.1602/2.99
    execution time (avg/stddev):   30.0889/0.11
```

*. 对磁盘io进行压测的例子：
先准备要测试的文件：

```
[tidb@rds1 sysbench]$ sysbench fileio --file-num=16 --file-total-size=2G prepare
sysbench 1.0.12 (using bundled LuaJIT 2.1.0-beta2)

16 files, 131072Kb each, 2048Mb total
Creating files for the test...
Extra file open flags: 3
Creating file test_file.0
Creating file test_file.1
Creating file test_file.2
Creating file test_file.3
Creating file test_file.4
Creating file test_file.5
Creating file test_file.6
Creating file test_file.7
Creating file test_file.8
Creating file test_file.9
Creating file test_file.10
Creating file test_file.11
Creating file test_file.12
Creating file test_file.13
Creating file test_file.14
Creating file test_file.15
2147483648 bytes written in 5.52 seconds (371.28 MiB/sec).
```

在性能测试服务器【Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz，40核/256G内存/SSD硬盘】上的测试结果展示：

```
[tidb@rds1 sysbench]$ sysbench fileio --time=60 --events=100000000 --threads=16 --file-num=16 --file-total-
size=2G --file-test-mode=rndrd --file-extra-flags=direct --file-fsync-freq=0 --file-block-size=16384 run
sysbench 1.0.12 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 16
Initializing random number generator from current time
```

```
Extra file open flags: 3
16 files, 128MiB each
2GiB total file size
Block size 16KiB
Number of IO requests: 100000000
Read/Write ratio for combined random IO test: 1.50
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random read test
Initializing worker threads...

Threads started!


File operations:
    reads/s:                      68278.84
    writes/s:                     0.00
    fsyncs/s:                     0.00

Throughput:
    read, MiB/s:                  1066.86
    written, MiB/s:               0.00

General statistics:
    total time:                          60.0003s
    total number of events:              4097004

Latency (ms):
         min:                                    0.03
         avg:                                    0.23
         max:                                   19.05
         95th percentile:                        0.32
         sum:                               957873.75

Threads fairness:
    events (avg/stddev):           256062.7500/426.01
    execution time (avg/stddev):   59.8671/0.00
```

作为对比，同样的测试，在一台很普通的机器【Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz，4核/8G内存/机械硬盘】上的测试结果：

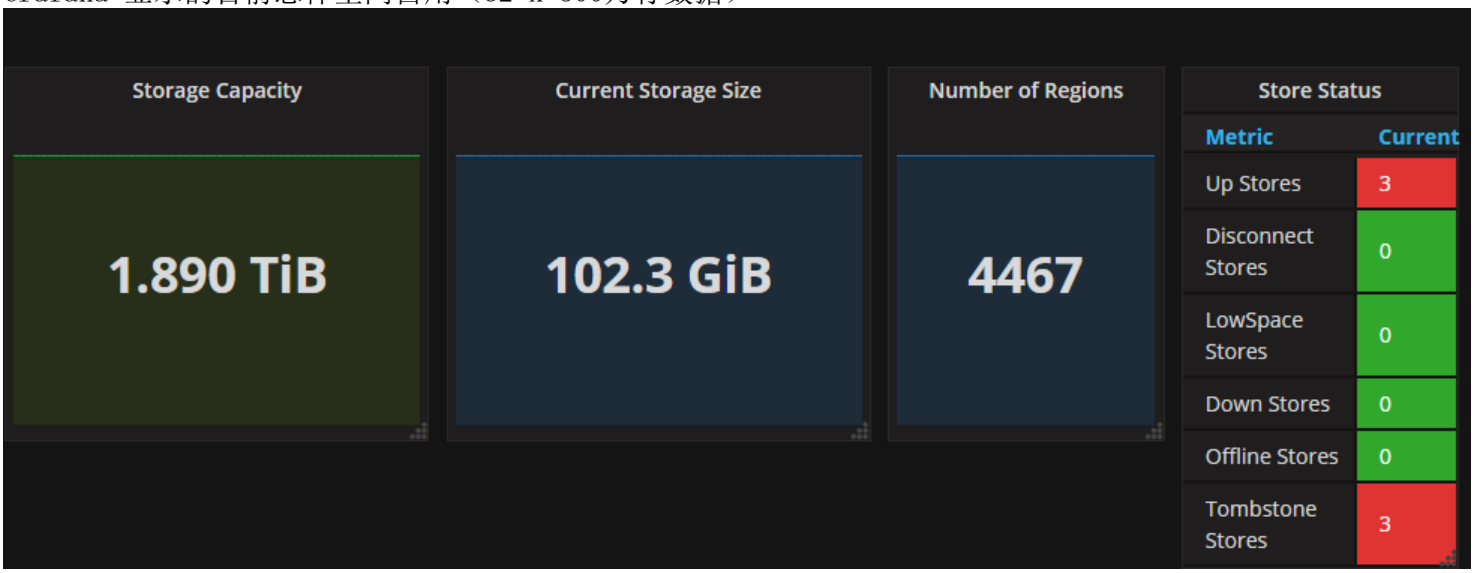
```
[root@rdsclient testsysbench]# sysbench fileio --time=60 --events=100000000 --threads=16 --file-num=16 --
file-total-size=2G --file-test-mode=rndrd --file-extra-flags=direct --file-fsync-freq=0 --file-block-
size=16384 run
sysbench 1.0.12 (using bundled LuaJIT 2.1.0-beta2)

Running the test with following options:
Number of threads: 16
Initializing random number generator from current time


Extra file open flags: 3
16 files, 128MiB each
2GiB total file size
Block size 16KiB
Number of IO requests: 100000000
Read/Write ratio for combined random IO test: 1.50
Calling fsync() at the end of test, Enabled.
Using synchronous I/O mode
Doing random read test
Initializing worker threads...

Threads started!


File operations:
    reads/s:                      326.51
```

```
    writes/s:                     0.00
    fsyncs/s:                     0.00

Throughput:
    read, MiB/s:                  5.10
    written, MiB/s:               0.00

General statistics:
    total time:                          60.0693s
    total number of events:              19614

Latency (ms):
         min:                                    0.19
         avg:                                   48.96
         max:                                  782.61
         95th percentile:                      164.45
         sum:                                960341.01

Threads fairness:
    events (avg/stddev):           1225.8750/48.05
    execution time (avg/stddev):   60.0213/0.02
```

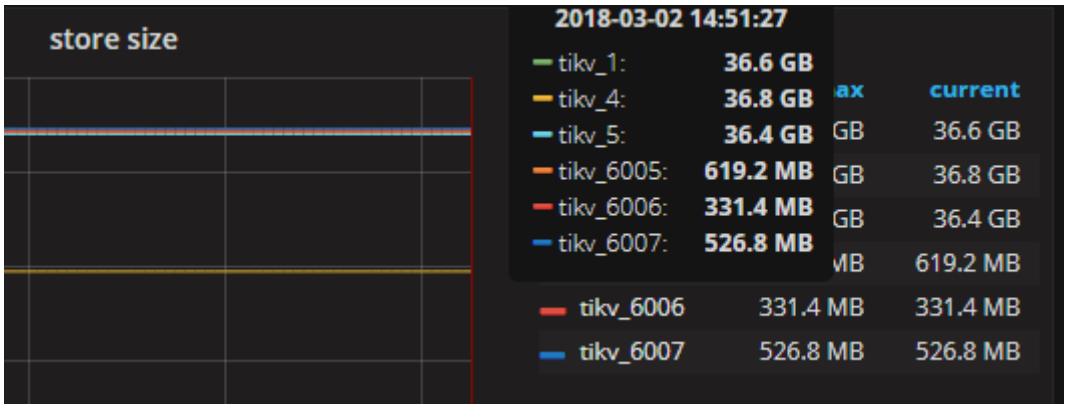
*. 本次TiDB sysbench测试步骤

```
*. 修改 pingcap/tidb-bench/sysbench/conf.sh 中的参数
*. （可选）如果修改了表数或表中的数据行数，可能需要重新准备数据
    cleanup.sh
    parallel-prepare.sh
*. （可选）如果修改了tidb的参数，需要重启 TiDB，在 172.20.129.11 的 /tidb/tidb-ansible-master 目录执行：
    ansible-playbook stop.yml
    ansible-playbook start.yml
**. 【重要】 在所有机器上修改 ~/bin/testname 文件，写入当前测试的名字，比如：tidb_16x100w_256t
*. 在所有机器上以“tidb” 用户（比如可以用：su - tidb转换到tidb用户）同时运行 nmon.sh，启动nmon监控（默认配置10分钟，到时会自动停止）。
*. 紧接着（尽量缩短时间），在所有运行sysbench客户端的机器上以“tidb” 用户sysbench目录下同时运行测试脚本，比如
./oltp.sh | tee $(teen)
*. 大约10分钟后（测试时间，可配置），测试结束，在 windows 机器上保存测试结果的目录中运行 cptr.bat，把测试结果一键拷贝到 windows机器：
   例如： cptr 20180211-102 （会调用 pscp 拷贝所有测试机 ~/nmon 目录中名字符合 *20180211-102* 的所有 .nmon 文件和 .log文件）
*. 如需要分析结果可用 nmon analyser v52_1.xlsm 把 .nmon 文件转换成 Excel 图表。
*. 测试数据量大的时候要注意监控硬盘的空间，如果不够，需要扩容TiKV。
```

Grafana 显示的目前总体空间占用（32 x 500万行数据）



每个TiKV的占用情况：



*. 测试过程中的一些非正式记录：
准备 16个表*100万行数据，
调用parallel-prepare并行准备数据：TiDB 需要6分钟(带宽平均80MBit/s)，Mysql主需要不到一分钟，算上三个从都同步完的时间总共也就2分钟的事(带宽平均445MBit/s)。
调用prepare串行准备数据：TiDB 需要20分钟以上（带宽用到20MBit/s），Mysql不到5分钟(带宽用到91MBit/s)

关掉 Mysql 半同步，采用异步复制，性能能提高 20-30%:
20180212-101010_rdsclient_mysql_16_100w_256t_async：tps：1404.02

再设置 innodb_flush_log_at_trx_commit = 0，又提高10%以上，
再设置 sync_binlog = 0，改变不明显。
再设置 thread_cache_size = 64，改变不明显。
再设置 innodb_buffer_pool_size = 64G，再上面基础上又提高了将近一倍，达到 tps: 2866

用 Mysql master所在的机器作为客户端，lo（127.0.0.1）的网卡流量跑到2G多（原来的客户端是千兆网卡），性能又提高了一倍以上，tps达到6735.16，CPU才用到40%多。

tidb_32x100w_256thread 每节点一个tikv（共三个） 4*sysbench 1*tidb tps:2779，只相当于PingCap结果4577的60%。TiDB所在的机器CPU占用达80%以上。

升级到 5.7.1-TiDB-v1.1.0-beta-5-gba04a34 后性能有下降（单TiDB server)：2473，
在v1.1.0-beta-5下采用4个TiDB Server，性能达到 4852，性能已经超过 PingCap的测试报告，再考虑到v1.1.0-beta版性能下降的因素，起码在这种测试条件组合下，性能指标满意。
单纯TiDB Server的机器CPU占用50%多，TiDB + TiKV的机器CPU占用将近80%。

每个节点新增一个TiKV实例，tps最终结果 3911，因为最后1分多钟性能总是严重下降。经过较长时间的稳定后，性能也比单TiKV要低。

*. PingCap 的测试报告
https://github.com/pingcap/docs-cn/blob/master/benchmark/sysbench.md

end