图床项目wrk+lua性能测试

零声教育 Darren QQ326873713

[零声教育 C/C++Linux服务器开发/高级架构师]

1 wrk安装和使用说明

1.1 安装wrk

wrk源码已经放到图床目录：**tuchuang/wrk_0voice**，相对于其他版本我以添加了dkjson.lua(用于json序列化和反序列化)

cd到wrk_0voice目录，进行安装

```
cd wrk_0voice
make
```

可能有一些包没有，导致git，make命令不能顺利执行，根据提示安装即可。

如果报错：unzip: Command not found，则安装

```
sudo apt-get install unzip
```

1.2 基本参数说明

以下是使用wrk查看到的一些基本参数信息

-c	**--connections**	**N**	**跟服务器建立并保持的TCP连接数量**
-d	--duration	T	压测**时间**
-t	--threads	N	使用多少个**线程**进行压测
-s	--script	S	指定**Lua脚本路径**
-H	--header	H	为每一个HTTP请求添加**HTTP头**
--latency			在压测结束后，打印**延迟统计信息**

-c	--connections	N	跟服务器建立并保持的TCP连接数量
--timeout		T	**超时**时间
-v	--version		打印正在使用的wrk的详细版本信息

- **N**代表**数字**参数，支持国际单位 (1k, 1M, 1G)
- **T**代表**时间**参数，支持时间单位 (2s, 2m, 2h)

wrk **-c** 20 **-t** 2 **-d** 20s **--latency** http://192.168.1.34

建立20个**TCP连接**，使用**两个线程**，用时**20秒**，对http://192.168.1.34进行压测。

1.3 测试范例

这里只是范例，不能直接用于图床的测试，图床测试见 本文第三章节《3 wrk+lua性能测试》

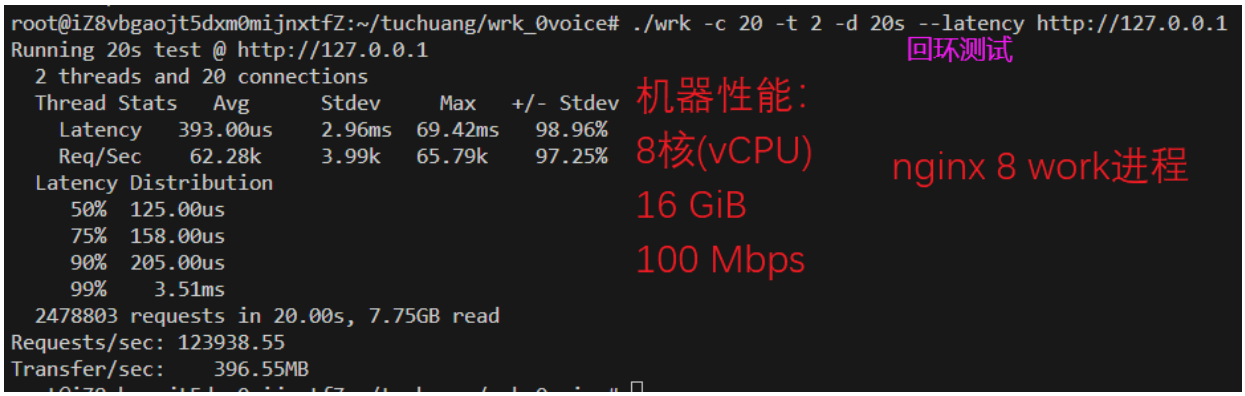
```
lqf@ceph-admin:~/flamegraph/wrk$ ./wrk -c 20 -t 2 -d 20s --latency
http://192.168.1.22
Running 20s test @ http://192.168.1.22
  2 threads and 20 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
              （平均值）（标准差）（最大值）（正负一个标准差所占比例）
    Latency     9.35ms    1.91ms  81.96ms   96.92%
    （延迟）
    Req/Sec     1.08k    73.16     1.52k    88.50%
    （处理中的请求数）
  Latency Distribution（延迟分布）
     50%    9.03ms
     75%    9.45ms
     90%    9.97ms
     99%   17.26ms（99分位的延迟）
  43019 requests in 20.03s, 170.79MB read（20.03s秒内共处理完成了43019个请求，读取了
170.79MB数据）
Requests/sec:   2148.24     （平均每秒处理完成2148.24个请求）
Transfer/sec:      8.53MB （平均每秒读取数据8.53MB）

lqf@ceph-admin:~/flamegraph/wrk$ ./wrk -c 20 -t 2 -d 20s --latency
http://120.27.131.197
Running 20s test @ http://120.27.131.197
  2 threads and 20 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    44.10ms   69.76ms 509.32ms   92.73%
    Req/Sec   382.88     80.68   470.00     88.14%
  Latency Distribution
```

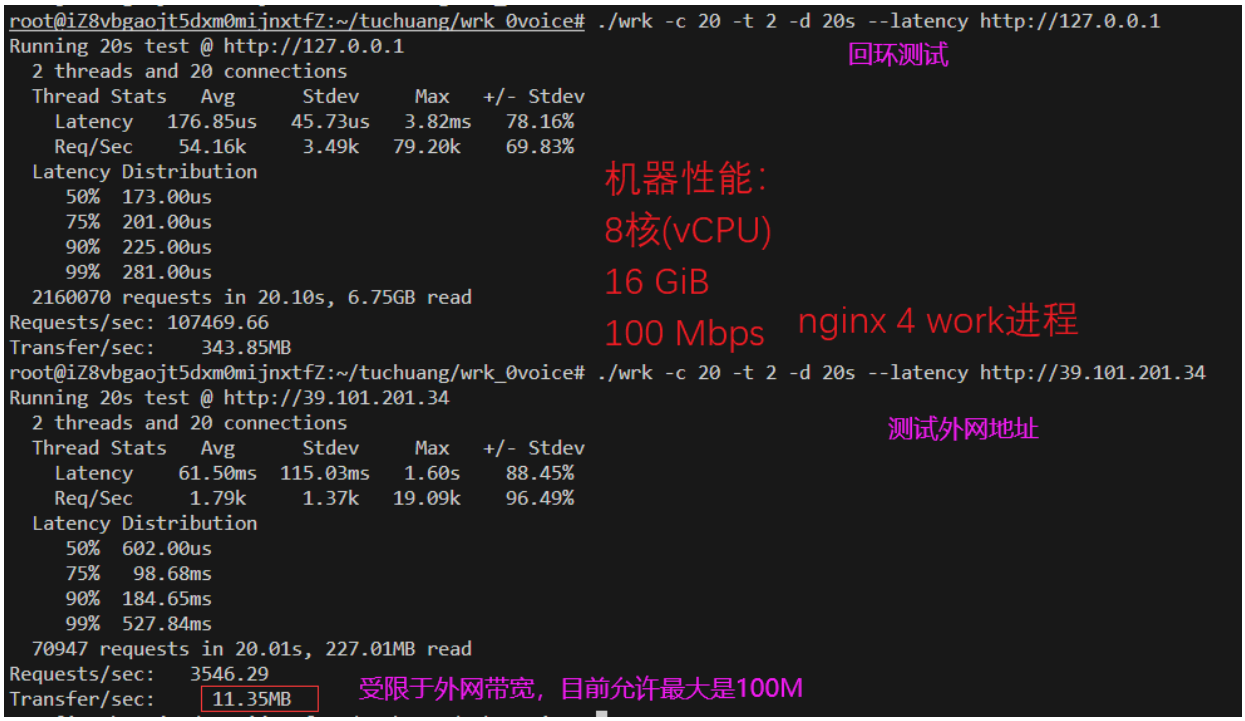
```
     50%   23.98ms
     75%   26.97ms
     90%   58.56ms
     99%  404.74ms
  14363 requests in 20.02s, 5.03MB read
Requests/sec:    717.38
Transfer/sec:    257.11KB
```

- Latency **延迟**时间
- Req/Sec **每秒处理**的请求数
- - **平均值**(Avg)，
- - **标准偏差**(Stdev)，
- - **最大值**(Max)
- - 正负一个**标准差占比**(+/-) Stdev

一般主要关注**Avg和Max。Stdev**如果太大说明样本本身**离散程度**比较高，有可能系统性能波动很大。

再看另一组测试数据。





2 lua环境和测试

2.1 安装lua

步骤1. 下载lua安装包、解压、编译、安装

```
curl -R -O http://www.lua.org/ftp/lua-5.3.0.tar.gz
# 如果上述命令不能正常下载lua则使用：wget http://www.lua.org/ftp/lua-5.3.0.tar.gz --no-check-certificate

tar zxf lua-5.3.0.tar.gz
cd lua-5.3.0
make linux
sudo make install
```

make的时候如果报错根据提示安装对应缺省的库，比如“lua.c:80:10: fatal error: readline/readline.h”，则安装sudo apt-get install libreadline-dev

成功安装后，运行 Lua 解释器，如下所示：

```
lua -v
```

步骤 2. 创建示例 Lua 程序。

现在我们创建一个示例程序**hello.lua**, 添加以下文件：

```
print("Hello lua")
```

Save 和 close 该文件，然后运行您的程序：

```
lua hello.lua
```

打印

Hello lua

脚本语言 lua、python

redis lua

3 wrk+lua性能测试

3.1 注册测试

在wrk_0voice目录

```
./wrk -c 100 -t 100 -d 5s --latency -s  scripts/reg.lua  http://127.0.0.1/api/reg
```

reg.lua是提前写好的（原来的wrk不带reg.lua）， http://127.0.0.1/api/reg如果是跨机器测试则填写目标ip的地址。

reg.lua大致的原理：

- 通过随机字符串给nickName和userName赋值。
- 然后发起post请求

tc_http_server.conf配置不同的线程池、连接池数量：

- ThreadNum
- tuchuang_master_maxconncnt
- tuchuang_slave_maxconncnt
- token_maxconncnt
- ranking_list_maxconncnt

```

```

```
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 10s --latency -s
scripts/reg.lua  http://127.0.0.1/api/reg
Running 10s test @ http://127.0.0.1/api/reg
  100 threads and 100 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    28.59ms    9.06ms 236.44ms   83.79%
    Req/Sec    35.01      8.08    60.00     82.66%
  Latency Distribution
     50%   26.83ms
     75%   31.31ms
     90%   37.27ms
     99%   64.63ms
  35515 requests in 10.10s, 9.55MB read
Requests/sec:   3516.45
Transfer/sec:      0.95MB
```

可以对比有索引数据库和没有索引数据库的区别。

服务端的**线程池和连接池数量设置**为50.

./wrk -c 50 -t 50 -d 5s --latency -s scripts/reg.lua http://127.0.0.1/api/reg

无索引性能:

```
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 50 -t 50 -d 5s --latency -s
scripts/reg.lua   http://127.0.0.1/api/reg
Running 5s test @ http://127.0.0.1/api/reg
  50 threads and 50 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    60.91ms   99.17ms 617.43ms   92.32%
    Req/Sec    29.60     11.18    60.00     84.26%
  Latency Distribution
     50%   31.50ms
     75%   45.48ms
     90%   79.76ms
     99%  538.10ms
  6800 requests in 5.10s, 1.83MB read
Requests/sec:   1333.01
Transfer/sec:    367.05KB
```

有索引性能

```
```

```
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 50 -t 50 -d 5s --latency -s
scripts/reg.lua   http://127.0.0.1/api/reg
Running 5s test @ http://127.0.0.1/api/reg
  50 threads and 50 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    44.02ms   84.29ms 616.73ms   93.00%
    Req/Sec    47.46     12.27    70.00     68.27%
  Latency Distribution
     50%   18.63ms
     75%   24.61ms
     90%   63.60ms
     99%  468.35ms
  10871 requests in 5.10s, 2.92MB read
Requests/sec:   2131.13
Transfer/sec:    586.80KB
```

3.2 登录测试

使用wrk_0voice/scripts/login.lua， 服务端单线程模式处理

使用wrk_0voice/scripts/loginm.lua 服务端多线程模式处理

需要根据自己的用户名和密码修改

lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 10s --latency -s scripts/login.lua http://127.0.0.1/api/login

```
//服务端单线程处理
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 5s --latency -s
scripts/login.lua    http://127.0.0.1/api/login
Running 5s test @ http://127.0.0.1/api/login
  100 threads and 100 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    51.64ms  136.28ms   1.06s    95.32%
    Req/Sec    42.86     12.64   121.00     54.99%
  Latency Distribution
     50%   23.10ms
     75%   29.33ms
     90%   35.23ms
     99%  855.05ms
  20573 requests in 5.10s, 6.38MB read
Requests/sec:   4034.40
Transfer/sec:      1.25MB

//服务端多线程处理
```

```
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 5s --latency -s
scripts/loginm.lua    http://127.0.0.1/api/loginm
Running 5s test @ http://127.0.0.1/api/loginm
  100 threads and 100 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    15.42ms    3.73ms  80.28ms   85.17%
    Req/Sec    65.06      9.94   141.00     84.03%
  Latency Distribution
     50%   15.23ms
     75%   16.85ms
     90%   18.54ms
     99%   26.05ms
  32356 requests in 5.10s, 10.03MB read
Requests/sec:   6343.81
Transfer/sec:      1.97MB
```

3.3 我的文件列表

使用wrk_0voice/scripts/myfiles.lua， 服务端单线程模式处理

使用wrk_0voice/scripts/myfilesm.lua 服务端多线程模式处理

需要根据自己的用户名和token修改， 目前设置最多拉取10个文件信息，这里需要token，浏览器登录图床的时候通过F12观察调试窗口获取。

lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 5s --latency -s scripts/myfiles.lua http://127.0.0.1/api/myfiles&cmd=normal

后台单线程模式

```
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 5s --latency -s
scripts/myfiles.lua    http://127.0.0.1/api/myfiles&cmd=normal
[1] 269509
lqf@ubuntu:~/tuchuang/wrk_0voice$ Running 5s test @ http://127.0.0.1/api/myfiles
  100 threads and 100 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   229.44ms  287.80ms   1.36s    86.17%
    Req/Sec     9.69      2.90    20.00     88.34%
  Latency Distribution
     50%   98.86ms
     75%  127.05ms
     90%  728.88ms
```

```
     99%    1.19s
  3382 requests in 5.10s, 8.01MB read
Requests/sec:    663.09
Transfer/sec:      1.57MB
```

后台多线程模式测试

```
lqf@ubuntu:~/tuchuang/wrk_0voice$ ./wrk -c 100 -t 100 -d 5s --latency -s
scripts/myfilesm.lua    http://127.0.0.1/api/myfilesm&cmd=normal
[1] 269610
lqf@ubuntu:~/tuchuang/wrk_0voice$ Running 5s test @ http://127.0.0.1/api/myfilesm
  100 threads and 100 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   117.06ms  240.20ms   1.26s    87.55%
    Req/Sec    45.60     12.04   161.00     75.21%
  Latency Distribution
     50%   21.97ms
     75%   28.25ms
     90%  495.33ms
     99%    1.01s
  17957 requests in 5.10s, 42.50MB read
  Socket errors: connect 0, read 0, write 0, timeout 5
Requests/sec:   3521.17
Transfer/sec:      8.33MB
```

3.4 其他

其他api可以根据以上测试范例以及深入学习lua进一步测试，lua用来测试真是一把利器。

自己一定要动手写测试范例。

4 补充

mysq修改客户端最大连接数

```
# 若因达到最大连接数导致mysql服务不可用，则需重启mysql服务，如果起不来只能重启机器。反之，直接登录mysql即可。
service mysqld restart
# 重新登录mysql
```

```
mysql -uroot -p
# 查看当前mysql最大连接数限制（未改过默认151）
mysql> show variables like 'max_connections';
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 151   |
+-----------------+-------+
1 row in set (0.00 sec)
# 设置最大连接数
mysql> set global max_connections=1000;
Query OK, 0 rows affected (0.00 sec)
#再次确认
mysql> show variables like 'max_connections';
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| max_connections | 1000  |
+-----------------+-------+
1 row in set (0.00 sec)
mysql> exit
```

5 参考教程

Lua 基础语法 (yuque.com)