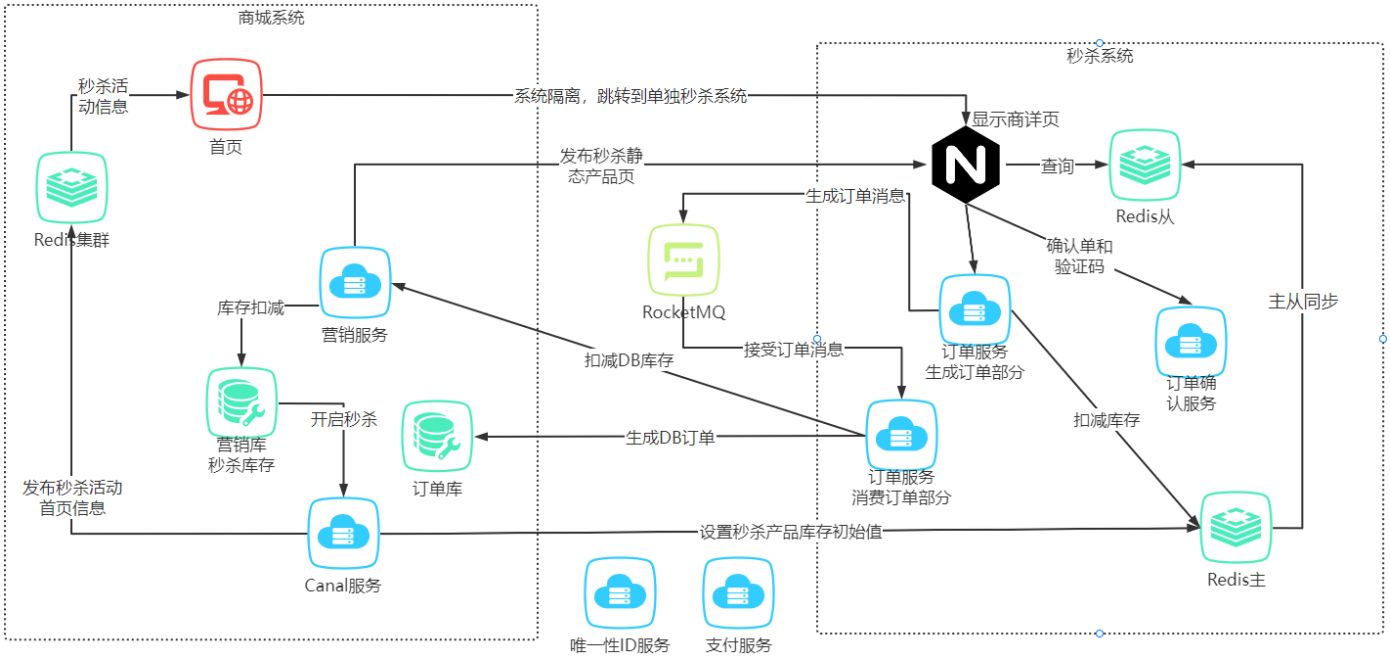
秒杀系统性能压测

秒杀系统快速回顾



1．运营人员在秒杀系统的运营后台，根据指定商品，创建秒杀活动，指定活动的开始时间、结束时间、活动库存等。

2．活动开始之前，由秒杀系统运营后台开启秒杀，会同时往商城系统的 Redis Cluster 集群写入首页秒杀活动信息和往秒杀系统的 Redis 主从集群写诸如秒杀商品库存等信息。

3．用户进入到秒杀商详页准备秒杀。

4. 商详页可以看到立即抢购的按钮，这里我们可以通过增加一些逻辑判断来限制按钮是否可以点击，比如是否还有活动库存，是否设置了预约等等。如果都没限制，用户可以点击抢购按钮，进入到订单确认服务，展示秒杀结算页。

5. 在结算页，用户可更改购买数量，切换地址、支付方式等。

6. 确认无误后，用户提交订单，在这里后端订单服务通过MQ异步完成数据库中库存的扣减和订单的生成。

7.订单完成后，根据用户选择的支付方式跳转到对应的页面，比如在线支付就跳转到收银台，货到付款的话，就跳到下单成功提示页。

为了应对秒杀的巨大的瞬时流量、热点数据问题等等问题，我们还实现了很多的举措，比如秒杀系统的隔离、商详页的静态化、流量管控等等。

我们对秒杀系统的压测主要集中在秒杀商详页和秒杀下单这两个一头一尾两个步骤上。

服务器基础配置

秒杀微服务服务器

首先：我们需要统计一下当前压测环境的服务器配置

```
# 物理核心数
cat /proc/cpuinfo | grep "physical id" | sort| uniq| wc -l
# CPU 核心数
cat /proc/cpuinfo | grep "cpu cores" |uniq
# CPU 处理线程数
cat /proc/cpuinfo | grep "processor" | wc -l
# CPU 主频
cat /proc/cpuinfo | grep MHz | uniq
# 内存配置
free -m
```

当前整体服务器配置如下：

相关服务器	物理 CPU	逻辑CPU	CPU 主频	内存
秒杀下单服务与从 Redis	1	4	2000HZ	4G
秒杀订单服务	1	4	2000HZ	4G
Nginx 服务器	1	4	2000HZ	4G
主 Redis	1	4	2000HZ	4G

基准测试

接下来：对主要的第三方组件，Redis 和 RocketMQ，做一下基准测试。

主Redis 基准测试

当前秒杀系统是采用一主一从的方式搭建的 Redis。主节点主要负责同步秒杀库存数据，以及实时扣减库存。我们先来测试一下主节点读写缓存的基准性能。

在对 Redis 进行压测前，需要注意一下 Redis 的几个对压测性能有比较大影响的配置。以下是redis.conf中几个重要的配置项：

```
# 日志备份采用appendonly
appendonly yes
# 日志刷盘设置为每秒刷盘 --有些压测环境会配置为不刷盘，这样可以提升压测性能
appendfsync everysec
```

在主节点上，我们使用 Redis 提供的redis-benchmark脚本进行基准测试。测试时，预设 100 个并发线程，进行 20W 次get 和 set 操作，测试的字节大小预设为 8 个字节。压测脚本如下：

```
/usr/local/redis-6.2.7/src/redis-benchmark -c 100 -n 200000 -t get,set -d 8
```

主要压测数据：

```
[root@192-168-65-190 src]# /usr/local/redis-6.2.7/src/redis-benchmark -c 100 -n 200000
-t get,set -d 8
====== SET ======
  200000 requests completed in 3.61 seconds
  100 parallel clients
  8 bytes payload
  keep alive: 1
  host configuration "save": 3600 1 300 100 60 10000
  host configuration "appendonly": yes
  multi-thread: no
.......
Summary:
  throughput summary: 55447.74 requests per second
  latency summary (msec):
          avg       min       p50       p95       p99       max
        1.231     0.216     0.999     2.359     2.831     8.415
====== GET ======
  200000 requests completed in 2.76 seconds
  100 parallel clients
  8 bytes payload
  keep alive: 1
  host configuration "save": 3600 1 300 100 60 10000
  host configuration "appendonly": yes
  multi-thread: no
.......
Summary:
  throughput summary: 72411.30 requests per second
  latency summary (msec):
```

```
          avg       min       p50       p95       p99       max
        0.718     0.208     0.743     0.935     1.183     2.159
```

从 Redis 基准测试

从节点主要负责同步秒杀库存数据。其作用主要是供 OpenResty 读取库存用，而 OpenResty 我们配置的是开启 7 个工作进程，加上主进程，一共是 8 个进程。所以我们在压测时，将并发数改为 8。

```
[root@192-168-65-28 src]# /usr/local/redis-6.2.7/src/redis-benchmark -c 8 -n 200000 -t
get -d 8
====== GET ======
  200000 requests completed in 3.36 seconds
  8 parallel clients
  8 bytes payload
  keep alive: 1
  host configuration "save": 3600 1 300 100 60 10000
  host configuration "appendonly": yes
  multi-thread: no
.....
Summary:
  throughput summary: 59541.53 requests per second
  latency summary (msec):
          avg       min       p50       p95       p99       max
        0.091     0.016     0.087     0.159     0.215     0.887
```

从经验判断，这个 Redis 服务的读写性能是相当低的，这跟当前电商项目采用的还是几年前的旧机器有关。这有可能会对后续整体性能产生比较大的影响。

为了让大家能够形成比较直观的对比，我们另外准备了一台阿里云上的云服务器，基础配置为 1CPU，2 核心，4 处理线程，主频2500HZ，内存32G。在上面进行同样的 Redis 压测，压测结果如下：

```
[root@iz8vb2n78bshjsty1jtrx8z src]# ./redis-benchmark -c 100 -n 200000 -t get,set -d 8
====== SET ======
  200000 requests completed in 1.27 seconds
  100 parallel clients
  8 bytes payload
  keep alive: 1
  host configuration "save": 900 1 300 10 60 10000
  host configuration "appendonly": yes
  multi-thread: no
.......
Summary:
  throughput summary: 157728.70 requests per second
  latency summary (msec):
          avg       min       p50       p95       p99       max
        0.338     0.104     0.319     0.431     0.727     3.831
====== GET ======
  200000 requests completed in 1.36 seconds
```

```
  100 parallel clients
  8 bytes payload
  keep alive: 1
  host configuration "save": 900 1 300 10 60 10000
  host configuration "appendonly": yes
  multi-thread: no
.......
Summary:
  throughput summary: 146627.56 requests per second
  latency summary (msec):
          avg       min       p50       p95       p99       max
        0.346     0.112     0.343     0.383     0.559     1.191
```

基准测试还只是测试 Redis 空转的读写性能，考虑到项目运行过程中，数据变化会更大，网络通信方面也会有性能损耗，实际项目中Redis 的并发读写性能还会更低。

那么 Redis 的读写性能会对业务具体产生什么样的影响呢？我们接下来对电商项目的商品详情页进行进一步压测。

Nginx静态网页基准测试

我们先来验证Redis 的读性能对秒杀的影响。以秒杀的商品详情页为例。在秒杀系统中，商品详情页处理是分为两个步骤，首先在秒杀开始时，通过 Freemark 生成商品对应的静态详情页，放到 OpenResty 当中。然后秒杀过程中，客户每次访问详情页，都会在 OpenResty 中直接通过lua脚本访问 Redis，实时获取商品库存。

我们先来测试 Redis 性能对于商品详情页的性能影响。尝试访问一个秒杀商品的详情页：http://192.168.65.28/seckill_3_29.html 。使用ab，模拟 200 个线程发送 100000 个请求。

```
[root@192-168-65-190 src]# ab -c 100 -n 30000 http://192.168.65.28/seckill_3_29.html
This is ApacheBench, Version 2.3 <$Revision: 1430300 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.65.28 (be patient)
Completed 3000 requests
Completed 6000 requests
Completed 9000 requests
Completed 12000 requests
Completed 15000 requests
Completed 18000 requests
Completed 21000 requests
Completed 24000 requests
Completed 27000 requests
Completed 30000 requests
Finished 30000 requests


Server Software:        openresty/1.21.4.1
Server Hostname:        192.168.65.28
```

```
Server Port:            80

Document Path:          /seckill_3_29.html
Document Length:        10209 bytes

Concurrency Level:      100
Time taken for tests:   2.712 seconds
Complete requests:      30000
Failed requests:        0
Write errors:           0
Total transferred:      313980000 bytes
HTML transferred:       306270000 bytes
Requests per second:    11063.60 [#/sec] (mean)
Time per request:       9.039 [ms] (mean)
Time per request:       0.090 [ms] (mean, across all concurrent requests)
Transfer rate:          113077.72 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    3   1.1      3      11
Processing:     0    6   1.6      6      14
Waiting:        0    3   1.1      3      11
Total:          2    9   1.9      9      20

Percentage of the requests served within a certain time (ms)
  50%      9
  66%     10
  75%     10
  80%     10
  90%     11
  95%     12
  98%     13
  99%     14
 100%     20 (longest request)
```

测试结果表明，这种情况下，单台秒杀Nginx能提供访问静态 Html 网页的 QPS 在 11K左右。

秒杀业务测试

接下来，我们开始针对秒杀业务进行业务测试。

秒杀详情页测试

秒杀业务入口为商品详情页。详情页需要在静态页面的基础上，再通过lua脚本访问 Redis，获取实时库存。我们同样使用ab来对商品详情页进行测试。测试地址为：http://192.168.65.28/product?flashPromotionId=3&promotionProductId=29&memberId=462245 这是一个发布后的商品秒杀页地址。

```
[root@192-168-65-190 src]# ab -c 200 -n 100000 http://192.168.65.28/product?
flashPromotionId=3\&promotionProductId=29\&memberId=462245
```

```
This is ApacheBench, Version 2.3 <$Revision: 1430300 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.65.28 (be patient)
Completed 10000 requests
Completed 20000 requests
Completed 30000 requests
Completed 40000 requests
Completed 50000 requests
Completed 60000 requests
Completed 70000 requests
Completed 80000 requests
Completed 90000 requests
Completed 100000 requests
Finished 100000 requests


Server Software:        openresty/1.21.4.1
Server Hostname:        192.168.65.28
Server Port:            80

Document Path:          /product?
flashPromotionId=3&promotionProductId=29&memberId=462245
Document Length:        9934 bytes

Concurrency Level:      200
Time taken for tests:   8.659 seconds
Complete requests:      100000
Failed requests:        0
Write errors:           0
Total transferred:      1009900000 bytes
HTML transferred:       993400000 bytes
Requests per second:    11548.80 [#/sec] (mean)
Time per request:       17.318 [ms] (mean)
Time per request:       0.087 [ms] (mean, across all concurrent requests)
Transfer rate:          113897.75 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    6   4.8      6    1005
Processing:     2   11   2.9     11      30
Waiting:        1    6   1.9      6      26
Total:          3   17   5.7     18    1014

Percentage of the requests served within a certain time (ms)
  50%     18
  66%     19
  75%     19
```

```
  80%     20
  90%     21
  95%     22
  98%     24
  99%     26
 100%   1014 (longest request)
```

商品详情页的 QPS 基本和静态网页保持在同一个水平，甚至略高。这其中是因为 Nginx 对静态网页提供了缓存能力。

商品库存数据获取

接下来，我们再针对商品详情页中最核心的从 Redis 读取库存的接口进行压测。测试地址：http://192.168.65.28/cache/stock?productId=29

```
[root@192-168-65-190 src]# ab -c 200 -n 100000 http://192.168.65.28/cache/stock?productId=29
This is ApacheBench, Version 2.3 <$Revision: 1430300 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.65.28 (be patient)
Completed 10000 requests
Completed 20000 requests
Completed 30000 requests
Completed 40000 requests
Completed 50000 requests
Completed 60000 requests
Completed 70000 requests
Completed 80000 requests
Completed 90000 requests
Completed 100000 requests
Finished 100000 requests


Server Software:        openresty/1.21.4.1
Server Hostname:        192.168.65.28
Server Port:            80

Document Path:          /cache/stock?productId=29
Document Length:        11 bytes

Concurrency Level:      200
Time taken for tests:   8.307 seconds
Complete requests:      100000
Failed requests:        0
Write errors:           0
Total transferred:      16600000 bytes
HTML transferred:       1100000 bytes
```

```
Requests per second:    12037.78 [#/sec] (mean)
Time per request:       16.614 [ms] (mean)
Time per request:       0.083 [ms] (mean, across all concurrent requests)
Transfer rate:          1951.44 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    7   2.0      7      25
Processing:     0   10   3.3     10     213
Waiting:        0    7   3.0      7     211
Total:          1   17   4.1     17     218

Percentage of the requests served within a certain time (ms)
  50%     17
  66%     18
  75%     19
  80%     19
  90%     20
  95%     22
  98%     24
  99%     25
 100%    218 (longest request)
```

可以看到，获取商品库存数据的 QPS 也保持在 11K 左右。所以，综合上面的测试，可以看到，在当前一台 Nginx 服务器的情况下，当前秒杀系统对商品详情页，可以支持 11K /S 左右的 QPS。

秒杀下单接口测试

接下来测试最为重要的秒杀下单接口。该接口整体的业务逻辑分为两个部分，首先是从缓存中进行库存扣减，然后异步发送消息到 MQ，完成下单操作。因此，在压测时，我们也针对两个部分分别进行测试。

MQ 生产者基准测试

由于在秒杀业务中，我们只需要将下单消息发到 RocketMQ 即可，所以，我们这里也只需要测试生产者发送消息的基准性能。

```
[root@192-168-65-164 benchmark]# pwd
/app/rocketmq/rocketmq-all-4.9.5-bin-release/benchmark
[root@192-168-65-164 benchmark]# ./producer.sh -n rocketmq1:9876
[root@192-168-65-164 benchmark]# Java HotSpot(TM) 64-Bit Server VM warning: ignoring
option PermSize=128m; support was removed in 8.0
Java HotSpot(TM) 64-Bit Server VM warning: ignoring option MaxPermSize=320m; support
was removed in 8.0
Java HotSpot(TM) 64-Bit Server VM warning: UseCMSCompactAtFullCollection is deprecated
and will likely be removed in a future release.
topic: BenchmarkTest threadCount: 64 messageSize: 128 keyEnable: false propertySize: 0
tagCount: 0 traceEnable: false aclEnable: false messageQuantity: 0
delayEnable: false delayLevel: 1
asyncEnable: false
Current Time: 2023-08-09 15:45:24,942 Send TPS: 22129 Max RT(ms): 719 Average RT(ms):
1.214 Send Failed: 0 Response Failed: 135
Current Time: 2023-08-09 15:45:34,942 Send TPS: 11927 Max RT(ms): 2979 Average RT(ms):
  2.738 Send Failed: 18 Response Failed: 224
Current Time: 2023-08-09 15:45:44,942 Send TPS: 14069 Max RT(ms): 2979 Average RT(ms):
  1.472 Send Failed: 81 Response Failed: 229
Current Time: 2023-08-09 15:45:54,942 Send TPS: 18185 Max RT(ms): 2989 Average RT(ms):
  1.542 Send Failed: 102 Response Failed: 299
```

默认参数下，我们这个 RocketMQ 集群的消息发送 TPS 大大低于理想的情况，并且测试中看到失败的消息非常多。那么在业务中 RocketMQ 的情况怎么样呢?

MQ 业务压测

我们将往 RocketMQ 发送下单消息的功能单独抽取出来，并在秒杀服务中提供了单独的测试接口进行性能测试。

```
    @ApiOperation("压测RockctMQ")
    @RequestMapping(value = "/pressureMQ",method = RequestMethod.POST)
    @ResponseBody
    public CommonResult pressureMQ(@RequestBody SecKillOrderParam secKillOrderParam,
                                   @RequestHeader("memberId") Long memberId) throws
BusinessException {
    }
```

在这个接口中会直接往 RocketMQ 中发送消息。

```
[root@192-168-65-190 ~]# curl -H 'Content-Type:application/json' -H 'memberId:1' -d
'{"flashPromotionId":3,"memberReceiveAddress":{"city":"长沙市","defaultStatus":
1,"detailAddress": "梅溪湖","id": 1,"memberId": 1,"name": "图灵Java","phoneNumber":
"13800000000","postCode": "414000","province": "湖南省","region": "岳麓区"},"orderId":
1,"orderItemId": 1,"payType": 3,"productId": 29}' -X POST
'http://192.168.65.133:9922/seckillOrder/pressureMQ'
{"code":200,"message":"操作成功","data":"success"}[root@192-168-65-190 ~]#
```

然后对这个接口进行压测。

```
[root@192-168-65-190 ~]# ab -c 20 -n 10000 -p "params" -T "application/json" -H
"Content-Type: application/json" -H "memberId: 1"
"http://192.168.65.133:9922/seckillOrder/pressureMQ"
This is ApacheBench, Version 2.3 <$Revision: 1430300 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.65.133 (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests


Server Software:
Server Hostname:        192.168.65.133
Server Port:            9922

Document Path:          /seckillOrder/pressureMQ
Document Length:        54 bytes

Concurrency Level:      20
Time taken for tests:   6.803 seconds
Complete requests:      10000
Failed requests:        16
   (Connect: 0, Receive: 0, Length: 16, Exceptions: 0)
Write errors:           0
Total transferred:      1598524 bytes
Total body sent:        5220000
HTML transferred:       548524 bytes
Requests per second:    1469.98 [#/sec] (mean)
Time per request:       13.606 [ms] (mean)
Time per request:       0.680 [ms] (mean, across all concurrent requests)
Transfer rate:          229.47 [Kbytes/sec] received
                        749.35 kb/s sent
                        978.82 kb/s total

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.2      0       6
```

```
Processing:     2   13  16.2     10     255
Waiting:        2   13  16.2     10     255
Total:          2   14  16.2     10     255

Percentage of the requests served within a certain time (ms)
  50%     10
  66%     13
  75%     15
  80%     16
  90%     20
  95%     27
  98%     52
  99%     85
 100%    255 (longest request)
```

这个压测的结果离理想情况差距非常大。这应该是受限于当前服务器的配置情况，最主要是在部署时，由于服务器内存不够，因此我们对每个 RocketMQ 的运行内存进行了限制。

鉴于这种情况，我们在实际的下单接口中，引入了异步线程池，来提升往 MQ 发消息的性能。

```
// 引入异步线程池
private static Executor sendMessage = new
ThreadPoolExecutor(2,4,60,TimeUnit.SECONDS,new ArrayBlockingQueue<>(50000));
// 异步发送消息
try {
            sendMessage.execute(new MessageTask(orderId,orderItemId,product,memberId,
                    secKillOrderParam,orderMessageSender,redisStockUtil,sendMessage));
        } catch (Exception e) {
            return CommonResult.failed("已经抢购完了，感谢参与本次活动");
        }
```

秒杀下单接口地址

接下来，再整合业务代码，对秒杀下单接口进行压测。压测之前，先测试一下接口功能是否正常。

```
[root@192-168-65-190 ~]# curl -H 'Content-Type:application/json' -H 'memberId:1' -d
'{"flashPromotionId":3,"memberReceiveAddress":{"city":"长沙市","defaultStatus":
1,"detailAddress": "梅溪湖","id": 1,"memberId": 1,"name": "图灵Java","phoneNumber":
"13800000000","postCode": "414000","province": "湖南省","region": "岳麓区"},"orderId":
1,"orderItemId": 1,"payType": 3,"productId": 29}' -X POST
'http://192.168.65.133:9922/seckillOrder/generateOrder'

{"code":200,"message":"下单中.....，后续请检查下单是否成功","data":{"orderItemList":
[{"id":1,"orderId":null,"orderSn":"1","productId":29,"productPic":"http://macro-
oss.oss-cn-
shenzhen.aliyuncs.com/mall/images/20180615/5acc5248N6a5f81cd.jpg","productName":"Apple
iPhone 8 Plus 64GB 红色特别版 移动联通电信4G手机","productBrand":"苹
果","productSn":"7437799","productPrice":4999.00,"productQuantity":1,"productSkuId":null
,"productSkuCode":null,"productCategoryId":19,"sp1":null,"sp2":null,"sp3":null,"promoti
onName":"秒杀特惠活
动","promotionAmount":0,"couponAmount":0,"integrationAmount":0,"realAmount":4999.00,"gif
tIntegration":5499,"giftGrowth":5499,"productAttr":null,"gmtCreate":null,"gmtModified":
null}],"orderStatus":"1","order":
{"id":1,"memberId":1,"couponId":null,"orderSn":"1","createTime":"2023-08-
10T07:41:03.511+00:00","memberUsername":null,"totalAmount":4999.00,"payAmount":4999.00,
"freightAmount":0,"promotionAmount":0,"integrationAmount":0,"couponAmount":null,"discou
ntAmount":500.00,"payType":3,"sourceType":0,"status":0,"orderType":1,"deliveryCompany":
null,"deliverySn":null,"autoConfirmDay":null,"integration":5499,"growth":5499,"promotio
nInfo":"秒杀特惠活
动","billType":null,"billHeader":null,"billContent":null,"billReceiverPhone":null,"billR
eceiverEmail":null,"receiverName":"图灵
Java","receiverPhone":"13800000000","receiverPostCode":"414000","receiverProvince":"湖南
省","receiverCity":"长沙市","receiverRegion":"岳麓区","receiverDetailAddress":"梅溪
湖","note":null,"confirmStatus":0,"deleteStatus":0,"useIntegration":null,"paymentTime":n
ull,"deliveryTime":null,"receiveTime":null,"commentTime":null,"modifyTime":null,"qrcode
":null,"gmtCreate":null,"gmtModified":null,"version":null}}}
```

然后使用ab进行压测。

```
[root@192-168-65-190 ~]# ab -c 1000 -n 10000 -p "params" -T "application/json" -H
"Content-Type: application/json" -H "memberId: 1"
"http://192.168.65.133:9922/seckillOrder/generateOrder"
This is ApacheBench, Version 2.3 <$Revision: 1430300 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 192.168.65.133 (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
```

```
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests


Server Software:
Server Hostname:        192.168.65.133
Server Port:            9922

Document Path:          /seckillOrder/generateOrder
Document Length:        1760 bytes

Concurrency Level:      1000
Time taken for tests:   2.466 seconds
Complete requests:      10000
Failed requests:        2959
   (Connect: 0, Receive: 0, Length: 2959, Exceptions: 0)
Write errors:           0
Total transferred:      13663165 bytes
Total body sent:        5250000
HTML transferred:       12613165 bytes
Requests per second:    4055.43 [#/sec] (mean)
Time per request:       246.583 [ms] (mean)
Time per request:       0.247 [ms] (mean, across all concurrent requests)
Transfer rate:          5411.13 [Kbytes/sec] received
                        2079.20 kb/s sent
                        7490.33 kb/s total

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0   26 143.6      1    1006
Processing:    65  199 164.5    129    1293
Waiting:        5  198 164.2    129    1293
Total:         65  224 222.3    131    1361

Percentage of the requests served within a certain time (ms)
  50%    131
  66%    163
  75%    294
  80%    328
  90%    442
  95%    567
  98%   1269
  99%   1292
 100%   1361 (longest request)
```

可以看到， 在各项服务基准测试都不太理想的情况下，秒杀下单接口依然能够保持 4000 左右的 TPS。

压测初步结论

可以看到，在各项服务基准测试都不太理想的情况下，秒杀下单接口依然能够保持 4000 左右的 TPS。这个 TPS 承担互联网一般的秒杀活动还是基本可以的。