

**Visitors**

	148,192		178
	3,069		169
	1,749		159
	1,715		126
	1,054		108
	368		106

Pageviews: 245,328  
Flags Collected: 78

 **FLAG** counter

A calendar for May 2018. The days of the week are labeled at the top: 日 (Sunday), 一 (Monday), 二 (Tuesday), 三 (Wednesday), 四 (Thursday), 五 (Friday), 六 (Saturday). The dates are arranged in a grid. The date 4 is highlighted in red.

日	一	二	三	四	五	六
29	30	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

- Linux(42)
- C(37)
- 网络编程(31)
- Socket(28)
- 环境配置(22)
- 物联网(22)
- 服务器(21)
- 年度月度总结(11)
- 感悟人生(9)
- java(9)
- 更多

- Egret(1)
- Netty网络编程(4)
- React Native(1)
- Socket系列(30)
- Spring XXX

随笔- 126 文章- 0 评论- 192

我以前也没有搞过高并发的网络，最近看了一些博客文章，准备用一个最简单的模型来实现这个百万QPS的模拟。后台我是用Java写的，Tomcat服务器。这种带业务的请求，单机绝对不可能到达百万QPS(1m-qps)。网上所说的1m-qps测试环境是一个静态的网页来测试的。既然带业务的http单机不能达到要求，那就需要用到集群了。就我目前所学到的知识，基本就只会搭建下面如图所示的集群架构了。



Windows网络编程(3)  
博客导航(2)  
感悟人生(10)  
后端架构之路(22)  
年度月度总结(11)  
学习笔记(4)  
置顶-实时更新(1)

## 📁 随笔档案

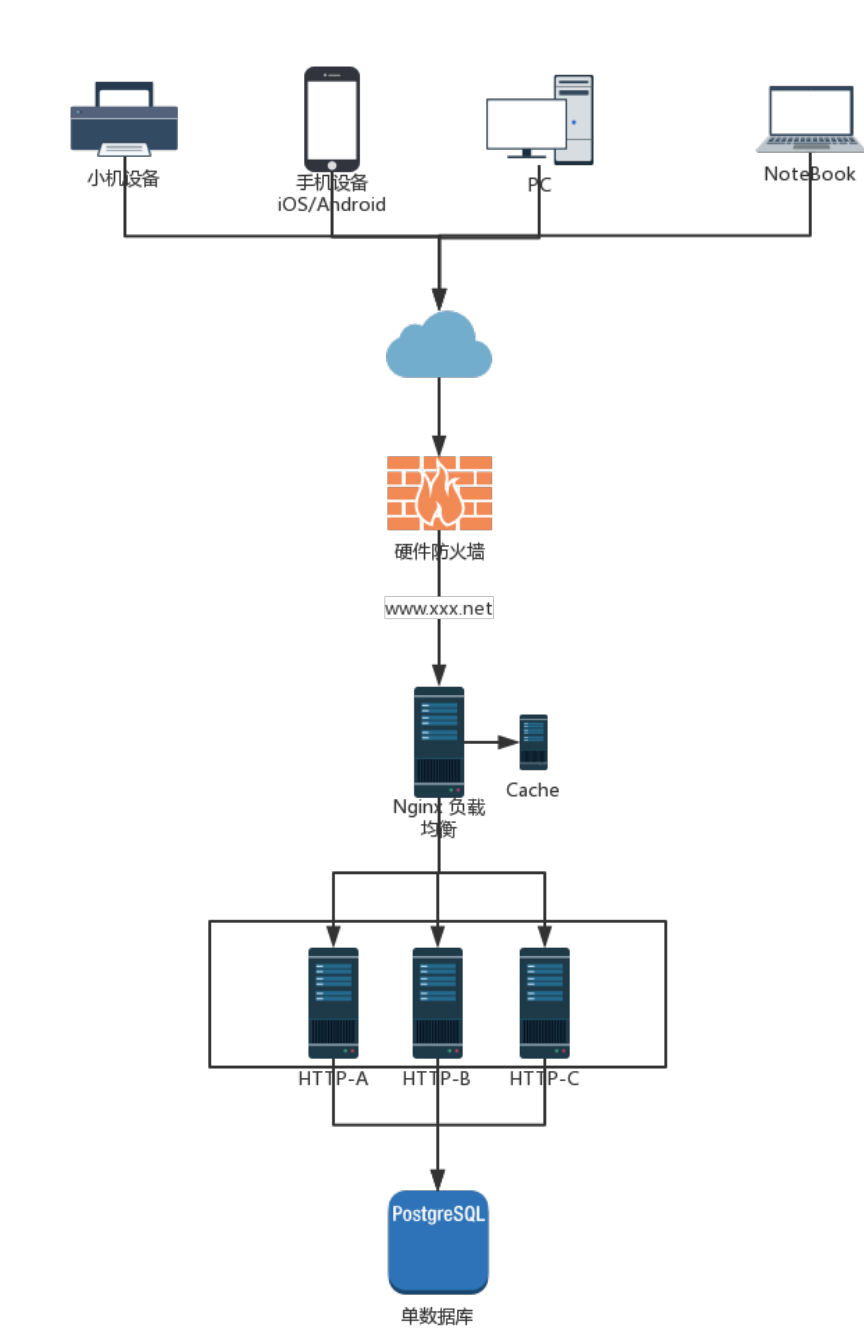
2018年4月 (1)  
2018年3月 (3)  
2018年2月 (5)  
2018年1月 (9)  
2017年12月 (6)  
2017年10月 (1)  
2017年8月 (1)  
2017年6月 (1)  
2017年3月 (1)  
2017年2月 (1)  
2016年11月 (2)  
2016年6月 (1)  
2016年5月 (1)  
2016年3月 (3)  
2015年12月 (4)  
2015年11月 (2)  
2015年6月 (4)  
2015年5月 (5)  
2015年1月 (2)  
2014年12月 (1)  
2014年11月 (1)  
2014年10月 (1)  
2014年9月 (7)  
2014年8月 (21)  
2014年7月 (12)  
2014年6月 (1)  
2014年5月 (5)  
2014年4月 (7)  
2014年3月 (6)  
2014年2月 (1)  
2014年1月 (1)  
2013年10月 (1)  
2013年7月 (2)  
2013年6月 (1)  
2013年5月 (1)  
2013年4月 (1)  
2013年1月 (3)

## 📊 积分与排名

积分 - 150474  
排名 - 1852

## 📖 阅读排行榜

1. Nginx + Tomcat 动静分离实现负载均衡(27371)
2. 搭建WebRtc环境(19681)
3. udp穿透简单讲解和实现(Java)(12782)
4. 微信电脑网页二维码扫描登录简单实现(11363)
5. JavaWEB springmvc 使用定时任务(10266)
6. Linux下C++连MySQL数据库(8417)
7. Linux下C语言使用openssl库进行加密(7380)
8. struts2漏洞-第一次入侵经历(7088)



一台性能比较好的机器按照nginx作为负载均衡，剩下的一些普通的机器就在后面。数据库目前只用到一个，后续如何优化和扩展不在本次讨论。

如果使用上面的架构，出去数据库后，整个HTTP的性能瓶颈就在最前面的Nginx负载均衡上了。其他的HTTP-jsp-tomcat机器如果性能不够，可以通过多加几台机器进行水平扩展。而最前面的Nginx就不行了。所以本次所要测试的就是如何让nginx支持1m-qps，并且机器正常工作。nginx只是做数据转发而已。

硬件：服务器88核、128G内存

软件：Debian 8， Nginx 1.11， wrk压力测试工具

### 三、操作

1. 首先设置一些Linux系统参数 在/etc/sysctl.conf 中增加如下配置

```
1 vm.swappiness = 0
2 net.ipv4.neigh.default.gc_stale_time=120
3 net.ipv4.conf.all.rp_filter=0
4 net.ipv4.conf.default.rp_filter=0
5 net.ipv4.conf.default.arp_announce = 2
6 net.ipv4.conf.all.arp_announce=2
7 net.ipv4.tcp_max_tw_buckets = 100
8 net.ipv4.tcp_syncookies = 0
9 net.ipv4.tcp_max_syn_backlog = 3240000
10 net.ipv4.tcp_window_scaling = 1
11 #net.ipv4.tcp_keepalive_time = 60
12 net.ipv4.tcp_synack_retries = 2
13 net.ipv6.conf.all.disable_ipv6 = 1
14 net.ipv6.conf.default.disable_ipv6 = 1
15 net.ipv6.conf.lo.disable_ipv6 = 1
16 net.ipv4.conf.lo.arp_announce=2
17 fs.file-max = 40000500
18 fs.nr_open = 40000500
19 net.ipv4.tcp_tw_reuse = 1
20 net.ipv4.tcp_tw_recycle = 1
21 net.ipv4.tcp_keepalive_time = 1
22 net.ipv4.tcp_keepalive_intvl = 15
23 net.ipv4.tcp_keepalive_probes = 3
```



9. 使用硬盘，安装双系统，Win7+CentOS(5334)
10. Nginx 单机百万QPS环境搭建(5073)
11. Zlib库的安装与使用(4795)
12. Socket网络编程--Libev库学习(1)(4420)
13. 学习笔记之gethostbyaddr函数(4143)
14. Netty5 + WebSocket 练习(4119)
15. Socket网络编程--简单Web服务器(1)(3706)

```
24
25 net.ipv4.tcp_fin_timeout = 5
26 net.ipv4.tcp_mem = 768432 2097152 15242880
27 net.ipv4.tcp_rmem = 4096 4096 33554432
28 net.ipv4.tcp_wmem = 4096 4096 33554432
29 net.core.somaxconn = 6553600
30 net.ipv4.ip_local_port_range = 2048 64500
31 net.core.wmem_default = 183888608
32 net.core.rmem_default = 183888608
33 net.core.rmem_max = 33554432
34 net.core.wmem_max = 33554432
35 net.core.netdev_max_backlog = 2621244
36 kernel.sem=250 65536 100 2048
37 kernel.shmmni = 655360
38 kernel.shmmax = 34359738368
39 kerntl.shmall = 4194304
40 kernel.msgmni = 65535
41 kernel.msgmax = 65536
42 kernel.msgmnb = 65536
43
44 net.netfilter.nf_conntrack_max=1000000
45 net.nf_conntrack_max=1000000
46 net.ipv4.netfilter.ip_conntrack_max=1000000
47 kernel.perf_cpu_time_max_percent=60
48 kernel.perf_event_max_sample_rate=6250
49
50 net.ipv4.tcp_max_orphans=1048576
51 kernel.sched_migration_cost_ns=5000000
52 net.core.optmem_max = 25165824
53
54 kernel.sem=10000 2560000 10000 256
```



还有在命令行中输入 `ulimit -n 20000500`

上面的所有参数，这里就不多讲了，具体想要了解的可以上网找资料

## 2.Nginx 安装 及其配置

nginx用最简单的apt-get 也可以，用源码按照也可以。没有什么特殊要求。下面的配置就有点要求了。 `$NGINX/conf/nginx.conf (/etc/nginx/conf/nginx.conf)`



```
1 worker_processes 88; #这个根据硬件有多少核CPU而定
2 pid logs/nginx.pid;
3
4 events {
5     worker_connections 1024;
6 }
7
8 http {
9     include mime.types;
10    default_type application/octet-stream;
11    sendfile on;
12    tcp_nopush on;
13
14    keepalive_timeout 65;
15
16    gzip off;
17    access_log off; #日志功能要关闭
18
19    server {
20        listen 888 backlog=168888;
21        server_name localhost;
22        root /dev/shm/;
23    }
24
25 }
```



上面这个是最简单的配置，具体的Nginx还有很多可以调优的，还有nginx负载均衡配置，请参考我的另外一片博客<[Nginx + Tomcat 动静分离实现负载均衡](http://www.cnblogs.com/wunaozai/p/5001742.html)> <http://www.cnblogs.com/wunaozai/p/5001742.html>>

```
1 worker_processes 4;
2
3 error_log /var/log/nginx/error.log info;
4
5 pid /var/run/nginx.pid;
6
7
8 events{
9     use epoll;
10    worker_connections 409600;
11    multi_accept on;
12    accept_mutex off;
13 }
14
15
16 http{
17     sendfile on;
18     tcp_nopush on;
19     tcp_nodelay on;
20     open_file_cache max=200000 inactive=200s;
21     open_file_cache_valid 300s;
22     open_file_cache_min_uses 2;
23     keepalive_timeout 5;
24     keepalive_requests 20000000;
25     client_header_timeout 5;
26     client_body_timeout 5;
27     reset_timedout_connection on;
28     send_timeout 5;
29
30     #日志
31     access_log off;
32     #access_log /var/log/nginx/access.log;
33     #error_log /var/log/nginx/error.log;
34     #gzip 压缩传输
35     gzip off;
36     #最小1K
37     #gzip_min_length 1k;
38     #gzip_buffers 16 64K;
39     #gzip_http_version 1.1;
40     #gzip_comp_level 6;
41     #gzip_types text/plain application/x-javascript text/css application/xml
application/javascript;
42     #gzip_vary on;
43     #负载均衡组
44     #静态服务器组
45     #upstream static.zh-jieli.com {
46     #     server 127.0.0.1:808 weight=1;
47     #}
48     #动态服务器组
49     upstream zh-jieli.com {
50         server 127.0.0.1:8080;
51     }
52     #配置代理参数
53     #proxy_redirect off;
54     #proxy_set_header Host $host;
55     #proxy_set_header X-Real-IP $remote_addr;
56     #proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
57     #client_max_body_size 10m;
```

```
58 #client_body_buffer_size 128k;
59 #proxy_connect_timeout 65;
60 #proxy_send_timeout 65;
61 #proxy_read_timeout 65;
62 #proxy_buffer_size 4k;
63 #proxy_buffers 4 32k;
64 #proxy_busy_buffers_size 64k;
65 #缓存配置
66 #proxy_cache_key '$host:$server_port$request_uri';
67 #proxy_temp_file_write_size 64k;
68 ##proxy_temp_path /dev/shm/JieLiERP/proxy_temp_path;
69 ##proxy_cache_path /dev/shm/JieLiERP/proxy_cache_path levels=1:2
keys_zone=cache_one:200m inactive=5d max_size=1g;
70 #proxy_ignore_headers X-Accel-Expires Expires Cache-Control Set-Cookie;
71
72 server{
73     listen 88 backlog=163840;
74     server_name test2;
75     root /dev/shm/;
76 }
77
78 server {
79     listen 443 ssl;
80     server_name test;
81     location / {
82         index index;
83     }
84
85     location ~ .*$ {
86         index index;
87         proxy_pass http://zh-jieli.com;
88     }
89     ssl on;
90     ssl_certificate keys/client.pem;
91     ssl_certificate_key keys/client.key.unsecure;
92 }
93 }
```



### 3. 创建一个简单的html文件

看到 root /dev/shm/ 了吗， 这个就是http服务器根目录了。 echo "a" > /dev/shm/a.html

### 4. 安装wrk

git clone https://github.com/wg/wrk.git 然后 make

## 四、运行

启动nginx

启动wrk 进行并发请求测试



```
./wrk -t88 -c10000 -d20s "http://127.0.0.1:888/a.html"
```



top结果图



```
top - 14:04:39 up 29 days, 23:27, 12 users, load average: 34.47, 35.58, 102.46
Tasks: 1350 total, 90 running, 1260 sleeping, 0 stopped, 0 zombie
%Cpu(s): 15.4 us, 65.6 sy, 0.0 ni, 1.5 id, 0.5 wa, 0.0 hi, 16.9 si, 0.0 st
KiB Mem : 13189204+total, 11453526+free, 12304640 used, 5052144 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 11741756+avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
 84700 root        20   0 6826076 160872 2860  S   28.9   0.1    8:43.33 wrk
 84680 nobody     20   0  25336   3224  2324  R   73.8   0.0    0:13.18 nginx
 84656 nobody     20   0  25336   3220  2348  R   73.1   0.0    0:12.21 nginx
 84668 nobody     20   0  25336   3216  2348  R   73.1   0.0    0:12.40 nginx
 84672 nobody     20   0  25336   3228  2348  R   73.1   0.0    0:12.12 nginx
 84689 nobody     20   0  25336   3212  2348  R   72.5   0.0    0:13.48 nginx
 84627 nobody     20   0  25336   3212  2348  R   71.5   0.0    0:11.31 nginx
 84690 nobody     20   0  25336   3192  2320  R   71.5   0.0    0:13.63 nginx
 84683 nobody     20   0  25336   3224  2344  R   70.8   0.0    0:13.40 nginx
 84684 nobody     20   0  25336   3224  2348  R   70.8   0.0    0:13.23 nginx
 84618 nobody     20   0  25336   3208  2348  R   70.5   0.0    0:10.73 nginx
 84687 nobody     20   0  25336   3200  2336  R   70.2   0.0    0:12.89 nginx
 84637 nobody     20   0  25336   3212  2348  R   69.5   0.0    0:12.49 nginx
 84653 nobody     20   0  25336   3224  2348  R   69.5   0.0    0:12.04 nginx
 84694 nobody     20   0  25336   3276  2348  R   69.5   0.0    0:13.48 nginx
 84679 nobody     20   0  25336   3228  2340  R   69.2   0.0    0:13.39 nginx
 84681 nobody     20   0  25336   3220  2340  R   69.2   0.0    0:12.71 nginx
 84616 nobody     20   0  25336   3212  2348  R   68.9   0.0    0:11.07 nginx
 84666 nobody     20   0  25336   3220  2340  R   68.9   0.0    0:11.92 nginx
 84667 nobody     20   0  25336   3212  2344  R   68.9   0.0    0:12.04 nginx
 84645 nobody     20   0  25336   3216  2348  R   68.5   0.0    0:11.77 nginx
 84665 nobody     20   0  25336   3220  2348  R   68.5   0.0    0:11.83 nginx
 84685 nobody     20   0  25336   3216  2348  R   68.2   0.0    0:13.26 nginx
 84644 nobody     20   0  25336   3216  2348  R   67.9   0.0    0:11.51 nginx
 84613 nobody     20   0  25336   3208  2348  R   67.5   0.0    0:10.60 nginx
 84628 nobody     20   0  25336   3212  2348  R   67.5   0.0    0:11.26 nginx
```

wrk 结果图

```
root@j1app2:~/workspace/wrk/wrk# ./wrk -t88 -c10000 -d20s "http://127.0.0.1:888/a.html"
Running 20s test @ http://127.0.0.1:888/a.html
 88 threads and 10000 connections
  Thread Stats   Avg      Stdev     Max   +/-  Stdev
    Latency    10.96ms   12.65ms  252.56ms   92.86%
    Req/Sec    13.24k    1.47k   61.95k    90.19%
 23430090 requests in 20.10s, 5.15GB read
Requests/sec: 1165578.52
Transfer/sec: 262.28MB
```

基本在120万QPS。nginx设置一些调优参数，加上如果不在本机上运行wrk的话，150万QPS基本是没有问题的。

五、要达到1m-qps需要注意

Linux 参数一定要进行调整（坑：这个坑是比较小的，基本提到高并发Linux，就会有提到要修改这些参数了）

Nginx 里 access\_log off; 日志功能一定要关闭（坑：这个影响还是比较大的，由于Nginx日志是写在磁盘的，而我服务器的磁盘是普通的磁盘，所以这里会成为磁盘IO瓶颈，所以一开始用最简单的配置，然后根据硬件的不同修改参数。有些参数在其他机器很快，在我的机器就很慢）

测试并发工具最好用我上面的wrk进行测试（坑：一开始用apache的ab进行测试，后面发现ab测试高性能时不行，最后使用wrk工具）

最好在本机测试wrk（坑：从上面的wrk图可以看到，传输的数据在每秒262MB左右，普通的100M网卡是远远达不到的。虽然我的a.html只有一个字节，但是http头部信息太多了。这里还有一个小坑，就是我的服务器是有千兆网卡的，但是我用另外一台测试机也是千兆网卡，但是中间的交换机或者路由器或者网线都达不到要求，所以这里会成为网络瓶颈）

上面那几个就是我遇到的坑了，特别是最后两个，一度还怀疑nginx性能问题。因为网上很多评测都说Nginx做代理可以达到百万QPS，我总是测试不到，基本在2~3万QPS而已。最后发现是测试工具和网卡原因。

六、最后多说两句

其实整个安装搭建测试环境都比较简单，这篇博客最主要的点还是最后的几点注意，由于以前没有搞过这方面，所以没有经验，这里记录下来，以后提醒自己。前端的负载均衡器保证没有问题后，接下来的问题是后面的HTTP服务集群了。我现在的Java功能后台一台普通的服



务器(Tomcat)就只能支持一万多的简单请求(因为jvm和web框架等种种原因)，然后如果是那种需要简单查询数据库的功能API就只有2~3K的QPS，最后那些需要事务操作的一台服务器仅仅只能支持120~150QPS。第一个还好，通过加机器就可以水平扩展，但是后面那两个跟数据库相关的就比较麻烦了，各种新的名词(NoSQL、缓存、读写分离、分库分表、索引、优化SQL语句等)都来了。能力有限，后续要学的东西太多了。

----- 2016-11-22 19:27:18 增加-----

单机进行测试的时候好多配置都没有用到，原因是，所有的请求和应答基本都是在内存处理的，没有创建socket连接，没有经过网卡网线路由器交换机等。这两天测试发现经过nginx代理后连接并发数上不去，查了一天，发现问题是nginx的keepalive参数没有打开。这个参数没有打开的话，会造成每次代理，都会创建一个http-socket，处理完就关闭，这样会比较占用资源，同时连接数上不去。加上keepalive参数后，外网的N个并发请求就可以通过一条socket发送多次请求。(这个描述比较不清楚, 如果对http1.0 http1.1 http2.0 里面的关于HTTP协议连接问题了解的话， 上面的描述就不难理解。)

集群的upstream设置为






```
1 # 这里要设置keepalive 表示本机与后端机器的连接数
2 # 同时这里还有一些其他设置，如权重，负载类型等
3 upstream wunaozai.cnblogs.com {
4     server 192.168.25.106:888;
5     server 192.168.25.100:888;
6     server 192.168.9.201:888;
7     keepalive 1000;
8 }
```





server代理设置为





```
1 server {
2     listen 88 backlog=168888;
3     server_name localhost2;
4     location ~ .*$ {
5         index index;
6         proxy_pass http://wunaozai.cnblogs.com;
7         proxy_set_header Connection "keep-alive";
8         proxy_http_version 1.1;
9         proxy_ignore_client_abort on;
10        #下面的timeout跟自己的业务相关设置对应的timeout
11        proxy_connect_timeout 600;
12        proxy_read_timeout 600;
13        proxy_send_timeout 600;
14    }
15 }
```





参考资料：

<http://datacratic.com/site/blog/1m-qps-nginx-and-ubuntu-1204-ec2>

<http://serverfault.com/questions/408546/how-to-achieve-500k-requests-per-second-on-my-webserver>



<https://lowlatencyweb.wordpress.com/2012/03/20/500000-requestssec-modern-http-servers-are-fast/>

<http://blog.jobbole.com/87531/>

<https://github.com/wg/wrk>

作者: [无脑仔的小明](#)

出处: <http://www.cnblogs.com/wunaozai/>

本文版权归作者和博客园共有，欢迎转载，但未经作者同意必须保留此段声明，且在文章页面明显位置给出原文连接，否则保留追究法律责任的权利。

如果文中有什么错误，欢迎指出。以免更多的人被误导。

分类: [学习笔记](#)

标签: [Linux](#), [Nginx](#), [Http服务器](#), [Tomcat](#)

好文要顶

关注我

收藏该文



无脑仔的小明

关注 - 17

粉丝 - 229

[+加关注](#)

1

推荐

0

反对

« 上一篇: [udp穿透简单讲解和实现\(Java\)](#)

» 下一篇: [简单搭建React-Native环境](#)

posted @ 2016-11-18 19:20 [无脑仔的小明](#) 阅读(5075) 评论(1) [编辑](#) [收藏](#)

### 评论

#1楼 2016-11-20 10:23 | [叛逆的小米](#)



头像是linux下的vim 我也很喜欢这个-。-

支持(0) 反对(0)

[刷新评论](#) [刷新页面](#) [返回顶部](#)



注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】超50万VC++源码：大型组态工控、电力仿真CAD与GIS源码库！

【活动】2050 大会 - 博客园程序员团聚 （5.25 杭州·云栖小镇）

【推荐】0元免费体验华为云服务

【活动】腾讯云云服务器新购特惠，5折上云

腾讯云

云数据库MySQL仅12元/月

满足入门学习、小规模应用、测试场景

立即购买

最新IT新闻:

- 创维激进的代价
- 小米上市估值或超百度，雷军如何穿越互联网黑暗森林？
- VS Code 1.23.0发布，带来众多更新
- Google发布开源容器运行时gVisor
- 李小加：在按正常程序尽快处理小米上市申请

» 更多新闻...

阿里云

517 通信节

短信服务低至3分/条

去省钱

广告





最新知识库文章:

- [如何成为优秀的程序员?](#)
- [菜鸟工程师的超神之路 -- 从校园到职场](#)
- [如何识别人的技术能力和水平?](#)
- [写给自学者的入门指南](#)
- [和程序员谈恋爱](#)
- » [更多知识库文章...](#)

Copyright ©2018 无脑仔的小明

