



# 如何精确监控DB响应延时

 老叶茶馆

+ 关注


发布于 2021-08-26 15:13:48 906


# 文章被收录于专栏： MySQL修行 | 老叶茶馆


 运行总次数： 0

代码可运行

关联问题 换一批 ×

 如何通过数据库自带的监控工具来精确监控DB响应延时？

 有哪些第三方工具可以精确监控DB响应延时？

 怎样设置阈值能精确监控DB响应延时？

本文首发于 「老叶茶馆」 微信公众号。

作者：任坤，现居珠海，先后担任专职 Oracle 和 MySQL DBA，现在主要负责 MySQL、MongoDB 和 Redis 维护工作

原创内容未经授权不得使用，转载请联系小编并注明来源。

## 1、背景

日常工作中遇到被问的最多的就是“现在应用有点慢，帮忙看看db是不是有问题”，第一反应是要去看一下监控告警。

问题是线上的监控指标很多，OS的包括cpu、内存、IO、网络，DB主要有TPS/QPS、活跃连接数、锁等待(细分的更多)，就算把所有监控信息都放到 grafana 单个页面展示，挨个查看也要耗费一点时间。

更关键的是，即便上述指标都正常，也不等同于DB无恙，即这些指标只能算是DB健康的必要条件，而不是充分条件。在某些场景下，即便这些指标都很平稳，开发可能依然会不断的质疑你，这时要如何快速自证清白？

首先来梳理一下DB响应流程：

从应用程序的角度观察，DB响应速度 = 网络延时 + 处理延时，其中处理延时的时间从请求抵达DB 服务器 开始，到服务器将响应结果发出结束。

DB服务器任何一个环节出现问题，都会增大处理延时，进而触发上述场景。

为此，我们只需要监控每个db请求【进入db服务器，db响应结束】这段时间的耗时，便可计算出每个db请求的处理延时，进而判定db服务器是否健康。 tcprstat 是专门为统计处理延时而生的工具。

## 2、原理

官档原文：原文。

javascript | 运行次数：0

AI代码解释

```
1 | Here “response time” means, for a given TCP connection, the time elapsed fr
```

通过( `src_ip, src_port, dst_ip, dst_port` )四元组可以唯一标识1个 `tcp` 连接，对于每个连接，计算其最后1个入包和第1个出包的时间差，以此得出1个请求的处理延时，然后将所有连接的请求处理延时聚集统计并输出。

这里有个前提，服务端IO模型必须是同步阻塞模式，即当前 `request` 的响应完成后，才能接受下一个 `request` ，好在目前的主流DB都符合这个要求。

`tcprstat` 在启动时会创建一个hash表，默认2053个bucket，每个bucket挂载一条单向链表，当出现hash冲突时，遍历该bucket下的链表直至找到匹配的item。

借助libpcap捕获数据包，首先将其还原成ip包(struct ip)，根据 `(ip->ip_p == IPPROTO_TCP)` 过滤掉非tcp包， 并且去除只包含控制信息的tcp包。tcprstat会记录每个符合条件数据包的时间戳tv，以及对应的四元组 `(src_ip, src_port, dst_ip, dst_port)` ，对四元组取模，以此在hash表中定位查找。

对应的 [数据结构](#) 很简单:

javascript | 运行次数：0

AI代码解释

```
1 | struct session {
2 |     uint32_t laddr, raddr;
3 |     uint16_t lport, rport;
4 |     struct timeval tv;
5 |     struct session *next;
6 | }
```

- 如果该包是入包，将其插入到hash表，若对应的item已经存在，覆盖其已有tv值。
- 如果该包是出包，根据其四元组从hash表取出对应item并将其从hash表移除，将两个包的tv相减便得出该请求的处理延时。

`tcprstat`将每个请求的处理延时保存到1个长整型数组中，每次输出都要对这个指针数组进行遍历，比如计算avg。

javascript | 运行次数：0

AI代码解释

```
1 | for (i = 0; i < n; i++)
2 |     avg += stats[i];
3 | avg /= n;
```



而计算99\_avg时，则先对指针数组调用qsort()进行升序排列，并只计算前99%的元素，排除最高的1%

javascript | 运行次数：0

AI代码解释

```
1 | n = ( n * percentile ) / 100
2 | for (i = 0; i < n; i++)
3 |     avg += stats[i];
4 | avg /= n;
```

计算min和max同理。

3、安装

安装很简单，直接将二进制文件下载就会执行

javascript | 运行次数：0

AI代码解释

```
1 | wget http://github.com/downloads/Lowercases/tcprstat/tcprstat-static.v0.3.1
2 | cp -a tcprstat-static.v0.3.1.x86_64 /usr/bin/tcprstat
3 | chmod +x /usr/bin/tcprstat
```

该工具默认直接查询机器的网络接口，在 bonding 模式的网卡下会报错，可改用ip列表。

javascript | 运行次数：0

AI代码解释

```
1 | [root@ ~]# tcprstat -p 3306
2 | pcap: SIOCGIFFLAGS: bonding_masters: No such device
3 |
4 | [root@ ~]# tcprstat -p 3306 -n 0 -t 1 -l `ls /sbin/ip | grep inet | egrep 'et
5 | timestamp      count    max      min      avg      med      stddev   95_max   95_
6 | 1488445537      0        0        0        0        0        0        0        0
7 | 1488445538      1       663      663      663      663      0        0        0
8 | 1488445539      0        0        0        0        0        0        0        0
9 | 1488445540      2       44       27       35       44      10       27       27
10 | 1488445541      3      604      46      264     142     243     142     94
```

以图形化的方式展现，当应用出现性能问题时可快速排查是否由DB引发，只有 avg/99\_avg 出现剧烈波动时，才能证明db服务器响应有问题。

4、总结

可以做如下结论：`tcprstat.avg/99_avg` 平稳是db健康的充分必要条件。虽然作者原本是为了监控mysql开发的此工具，但是对于 `mongo` 、 `redis` 同样适用，只需要修改监控端口即可。

全文完。

tcp/ip

编程算法

数据库

云数据库 SQL Server

sql

本文分享自 老叶茶馆 微信公众号，[前往查看](#)

如有侵权，请联系 [cloudcommunity@tencent.com](mailto:cloudcommunity@tencent.com) 删除。

本文参与 [腾讯云自媒体同步曝光计划](#) ，欢迎热爱写作的你一起参与！

### 推荐阅读

#### 编辑精选文章

- ☰ QQ 25年技术巡礼 | 技术探索下的清新...
- ☰ 【万字长文】论如何构建一个资金账户...
- ☰ 因为一部遮天，我用三种语言实现了腾...

#### 相关讨论

- ☰ 日志延时是多少？
- ☰ 推流延时性问题？
- ☰ cmq订阅模式怎么发送延时消息？

#### 相关

- ☰
- ☰
- ☰

### 龙马卫士(WLM9000B-3100S)防火墙日志处理工具使用文档

👁 104

💬 2

### 收集各类安全设备、Nginx日志实现日志统一管理及告警

👁 1.7K

💬 0

### Linux 网络命令必知必会之 tcpdump，一份完整的抓包指南请查收！

👁 7.2K

💬 0

HOT

云数据库采购特惠，高性能、高可...

#### 云数据库 SQL Server

产品介绍

产品文档

#### 云服务器

产品介绍

产品文档

HOT

#### 云数

### pcap文件格式及文件解析[通俗易懂]

👁 10.7K

💬 0

## 无心插柳还是有意为之：TCP反射DDoS攻击手法深入分析

 11.2K

 0

## 基于开源蜜罐的实践与功能扩展

 1.7K

 0

## TCP经典异常问题探讨与解决

 1K

 0

广告

TDSQL-C MySQL版 免费体验15天



## 【译】使用 SO\_REUSEPORT 套接字开发高并发服务

 882

 0

## [译] 利用 eBPF 支撑大规模 K8s Service

 992

 0

## CentOS7下部署OSSEC开源主机入侵检测系统(HIDS)并接入到GrayLog

 3.6K

 1

## OSSIM传感器代理传送机制分析

 728

 0

## 从STGW流量下降探秘内核收包机制

 3.7K

 1

## Linux流量分析：tcpdump&wireshark

 182

 0

## Python：基于Scapy的深度包分析与网络攻击防御方案



 271

 0

### TCP协议的定义和丢包时的重传机制

 2.1K

 0

### Linux用户态协议栈与DPDK构建高性能应用

 629

 0

### 高性能网络编程 – 白话TCP 三次握手过程

 318

 0

### Python的无状态SYN快速扫描

 2.7K

 0

### Windows eBPF 程序的开发步骤

 211

 0

### 不要启用 net.ipv4.tcp\_tw\_recycle

 7.8K

 0