

搜狗搜索测试分享

上·网·从·搜·狗·开·始

主要内容

- 搜索引擎的常见问题
 - 网络连接与容灾
 - 协议容错与兼容
 - 内存使用与线程安全
- 有趣的Bug

搜索引擎问题的由来

- 搜索引擎的特点
 - 庞大的分布式系统
 - 海量数据
 - 极高的性能要求
 - 极高的稳定性要求
- 大量自主开发的网络相关方法、数据库
 - 对通用协议的二次开发
 - 高性能数据库
 - 连接池特殊处理

网络连接与容灾

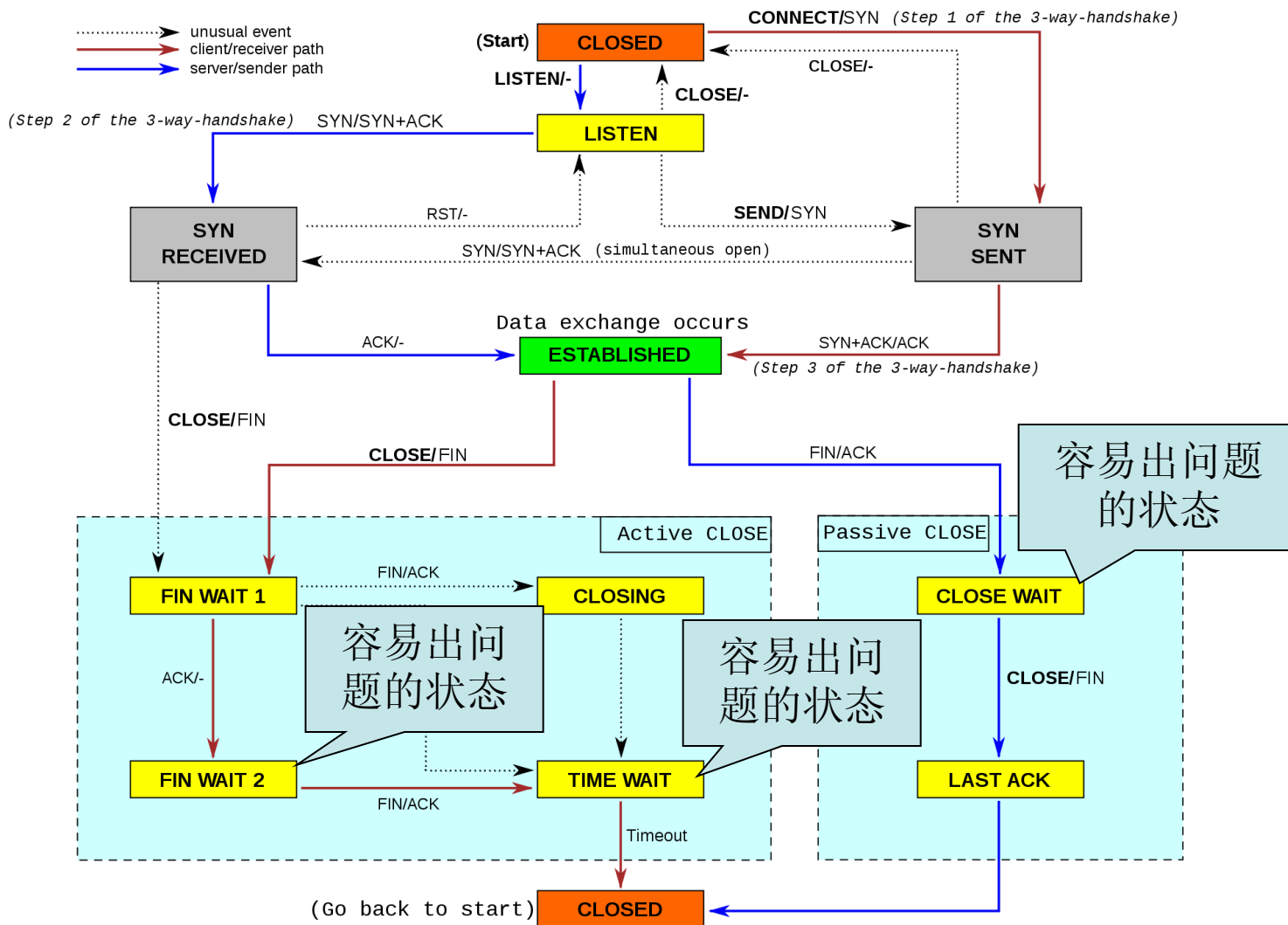
- 常见灾难
 - 后端服务down
 - 后端服务返回慢、返回异常
 - 内部网络闪断
 - 服务器宕机、掉电
 - 大流量冲击
- 灾难的扩大化与雪崩效应

网络连接与容灾

- 搜索引擎使用的网络协议
 - 不同层次的模块使用各种不同的协议
 - HTTP、TCP、UDP等等
 - 最常用的是TCP协议
- TCP带来的问题
 - 连接管理
 - 超时控制

网络连接常见问题

- 连接管理
 - 半关闭状态（客户端已关闭但服务器端未关闭）
 - TIME_WAIT状态连接过多
 - 脏连接放入连接池



网络连接常见问题

- 超时处理

- DNS超时

- 1.未配置DNS
- 2.请求DNS时，DNS服务器长时间不返回。

- 连接超时

- 1.目标ip非法
- 2.目标ip无路由
- 3.目标ip down机
- 4.目标ip设置了iptables等

- 发送超时

- 接收超时

- 队列超时

- 1.发送队列
- 2.接收队列
- 3.待处理队列

- 超时出错的后果

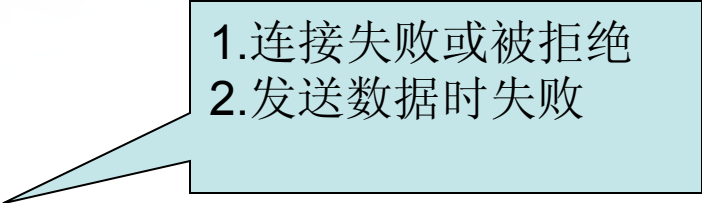
- 1.响应极慢
- 2.无法恢复
- 3.影响面扩大

网络连接常见问题

- 阻塞
 - 服务端阻塞
 - 客户端阻塞
 - 数据丢失 (UDP)
- 阻塞的测试模拟方法

网络连接常见问题

- 重连



1. 连接失败或被拒绝
2. 发送数据时失败

- 发现异常

- 重连频率

- 重试策略

网络连接常见问题

- 回顾之前提到的灾难

- 后端服务程序down
- 后端服务返回慢、返回异常
- 内部网络闪断
- 服务器down机、掉电
- 大流量冲击

1.重连
2.是否需要阻塞

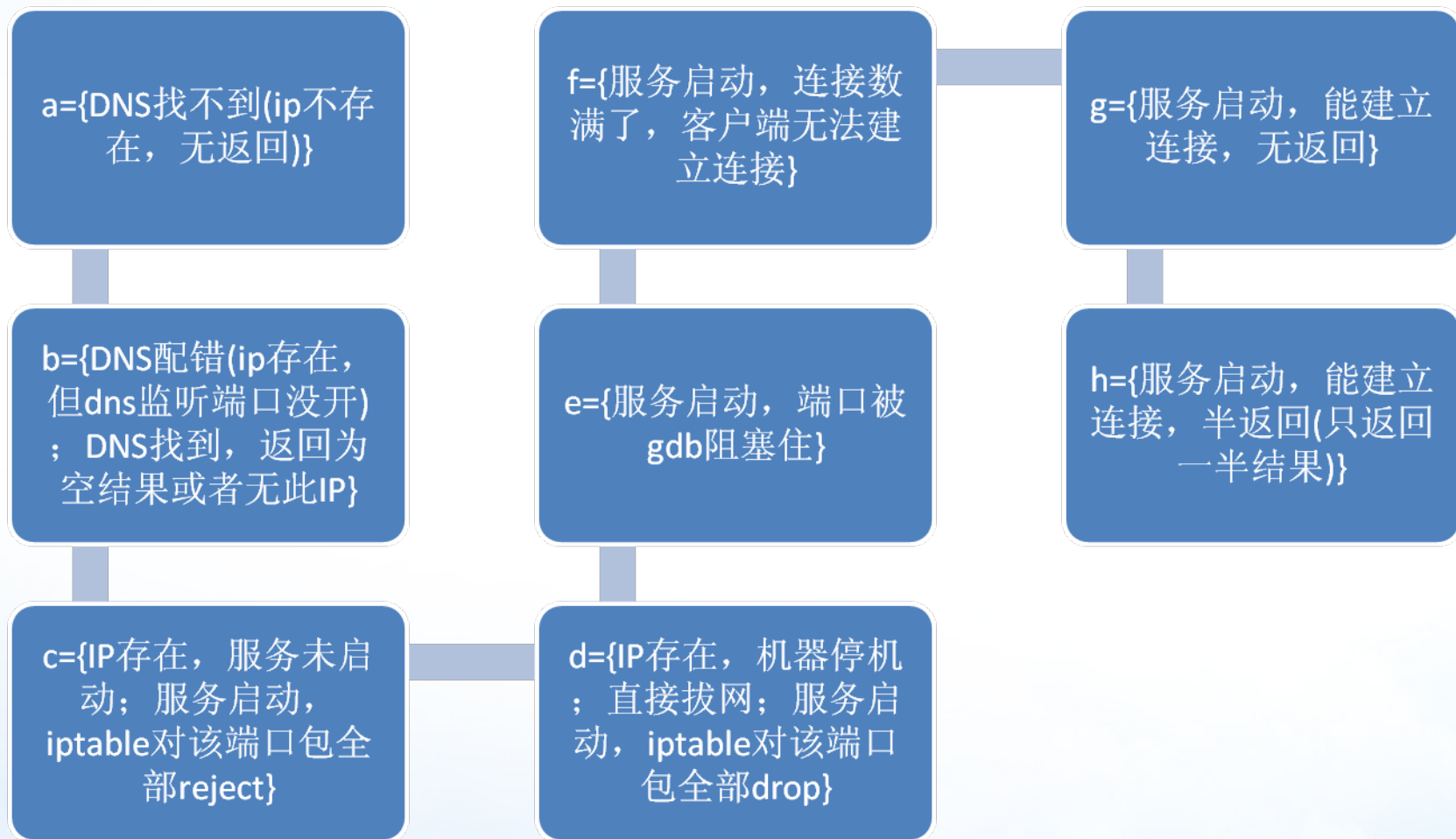
1.请求超时
2.脏连接关闭
3.重连

1.DNS超时
2.连接、超时
3.脏连接关闭
4.重连

1.连接超时
2.脏连接关闭
3.重连

1.连接超时
2.队列超时

网络连接常见问题



协议容错与兼容

- 协议容错

- 数据头与数据体一致性判断

1. 数据类型判断
2. 变长数据长度判断

- 字段一致性

1. 字段数量
2. 字段名称
3. 字段顺序
4. 字段长度或类型

- Key与value的一致性校验

- 各字段取值范围

协议容错与兼容

- 协议兼容

- 旧代码对新协议的兼容

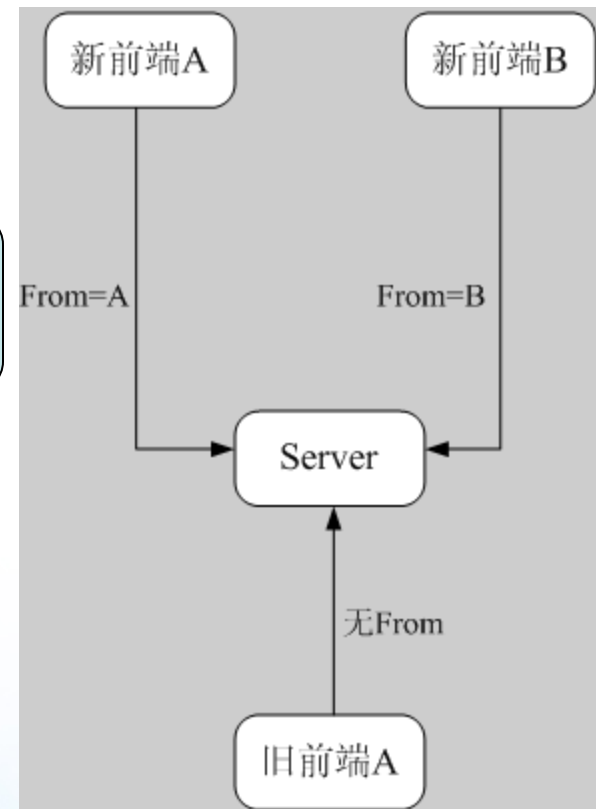
1. 新增了字段
2. 删除了字段
3. 字段新的取值

- 新代码对旧协议的兼容

缺少字段的
默认值

- 协议版本号

- 缓存兼容性



内存使用和线程安全

- 内存使用常见问题

- 错误使用

- 1.未申请成功就使用
- 2.未初始化就使用，缺省值真的不一定为0

- 内存泄漏

- 1.容错分支最易漏
- 2.内存碎片

- 内存越界

- 1.静态数组越界
- 2.指针移动越界
- 3.真的不一定core

- 释放了还用

- 1.Double free
- 2.Free后没有将指针置为NULL产生野指针
- 3.Return了指向“栈内存”的指针或引用，该内存存在函数体结束时被自动销毁

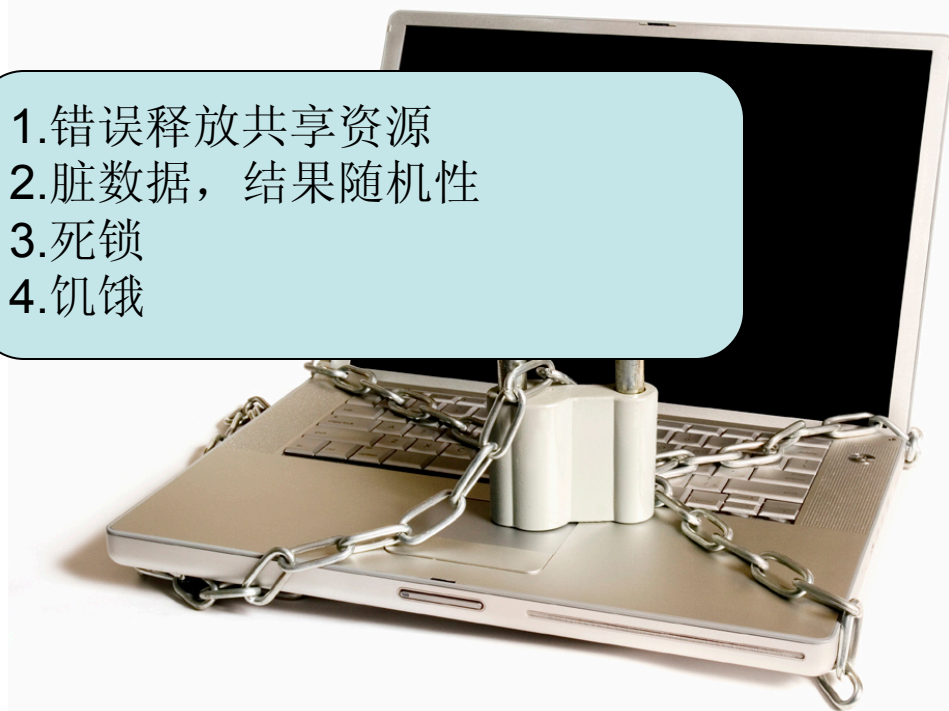
内存使用和线程安全

- 线程安全

- 加锁不当

1. 错误释放共享资源
2. 脏数据，结果随机性
3. 死锁
4. 饥饿

- 编译优化

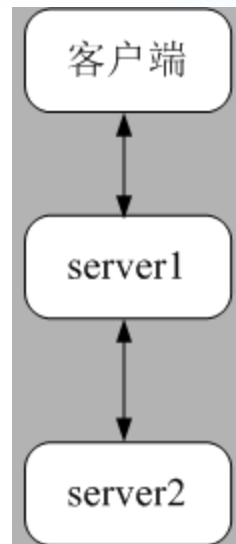


有趣的缺陷



性能测试压力不是越大越好

- 失败的一次性能测试
 - 测试时50qps，平均响应时间5ms
 - 上线后实际压力2q/s，平均响应时间200ms以上
- 问题分析
 - 配置、请求类型、数据量、硬件都完全一样
 - 性能差时系统资源很空闲
 - 定位到nagle算法



性能测试压力不是越大越好

- 什么是nagle算法
- 带来的问题
 - 会与TCP的延迟确认机制发生冲突

性能测试压力不是越大越好

- 回到之前的测试
 - 测试时压力远高于线上实际压力
 - 缓冲区被迅速塞满，发送延迟很小
 - 使用TCP套接字的 `TCP_NODELAY`选项关闭nagle
- 高交互性实时系统
 - 认真分析系统特性
 - 准确预测上线流量并精确测试

字符串不能乱做加法

- BUG: 查询结果怎么串了?
- 背景介绍:
 - 被测模块可写入数据并在查询时读取
 - 使用hash表存储, 将查询词与查询类型合并签名做key
 - 查询词与查询类型合并时, 使用了按位相加。
 - 计算hash签名时, 使用了**strcpy**

字符串不能乱做加法

- BUG分析

- 跟踪缓存中串了的数据
- 合并查询词与查询类型的代码
- 有可能加出\0
- 使用strcpy时，错误的被截断后计算hash
- 导致hash的key严重冲突

```
if (len >= sizeof(word)) {  
    len = sizeof(word) - 1;  
    for (size_t i = 0; i < len; i+  
+)  
        {  
            word[i] += key.prd;  
        }
```

查询词：搜狗搜索

CB D1 B9 B7 CB D1 CB F7

查询类型：2F

D1: 11010001

2F: 00101111

相加后出现一个字节的0

连接池的BUG

- 查询又串了.....
 - 压力测试中手工查询
 - 出现一次之后就总是串
- Bug分析
 - 检查log发现串之前出现了一次请求超时
 - 请求超时后，将连接放回了客户端连接池
 - 此时返回了数据，后续线程使用该连接
 - 发现连接可读，读取了缓冲区里前一个请求的返回
 - 然后又放回了连接池.....


黑名单匹配的BUG

- BUG现象

- 查询一些正常数字时被告知命中黑名单无结果

- 背景

- 黑名单按精确匹配
- 黑名单格式



查询词	权重	级别
Abc	20	0
Bdd	10	1
.....		

- Bug原因

- 错误的将整行进行了匹配

谢谢观赏