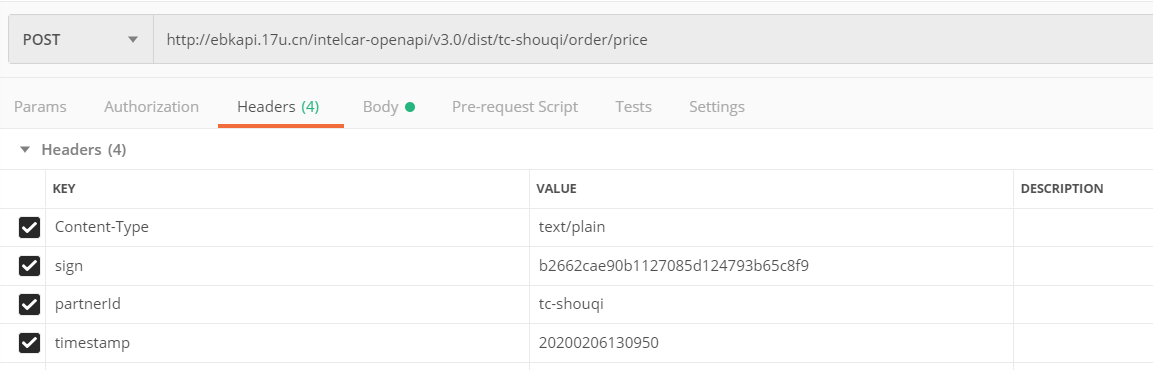
OpenAPI





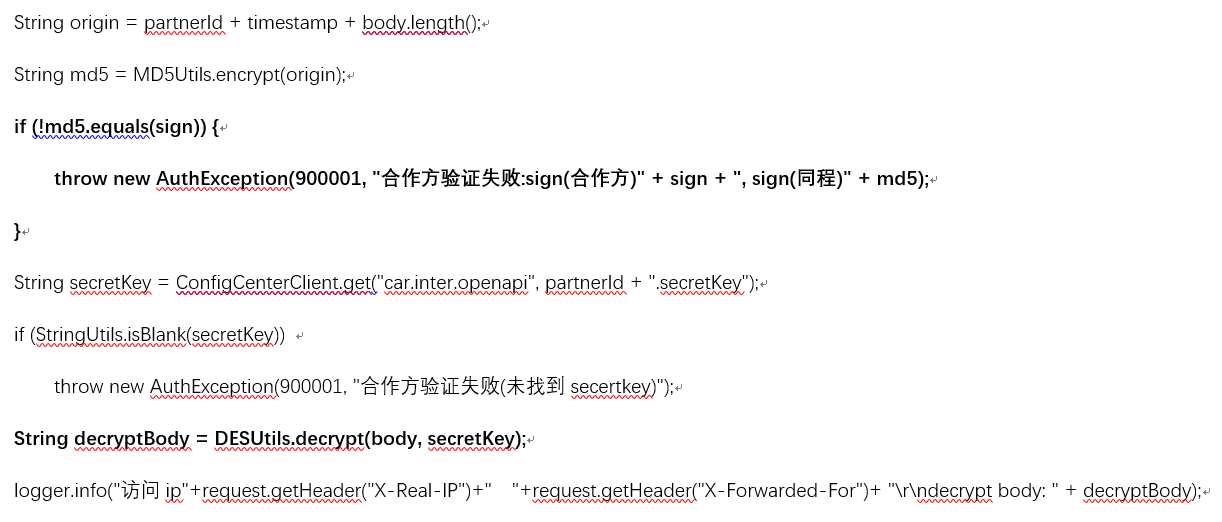
des对称加密

des对称加密，是一种比较传统的加密方式，其**加密运算、解密运算使用的是同样的密钥**，信息的发送者和信息的接收者在进行信息的传输与处理时，必须共同持有该密码（称为对称密码），是一种[对称加密算法](https://baike.baidu.com/item/%E5%AF%B9%E7%A7%B0%E5%8A%A0%E5%AF%86%E7%AE%97%E6%B3%95/211953)。

md5是一种**不可逆的加密，一定记住是不可逆的**

“md5是一种**信息摘要算法，它可以从一个字符串或一个文件中按照一定的规则生成一个特殊的字符串，并且一个文件所对应的MD5摘要是固定的，当文件内容变化后，其MD5值也会不一样，**因此，在应用中经常使用MD5值来验证一段数据有没有被篡改

String origin = partnerId + timestamp + body.length();



**cpu 100%怎样定位**

# CPU100%问题如何快速定位

**简要步骤如下：**

（1）找到最耗CPU的进程；

（2）找到最耗CPU的线程；

（3）查看堆栈，定位线程在干嘛，定位对应代码；

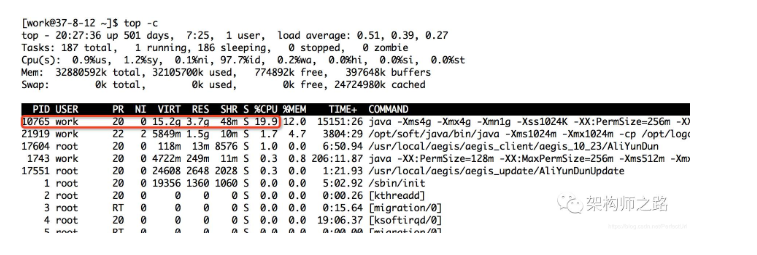
服务器CPU突然告警，如何定位是哪个服务进程导致CPU过载，哪个线程导致CPU过载，哪段代码导致CPU过载？

**步骤一、找到最耗CPU的进程**

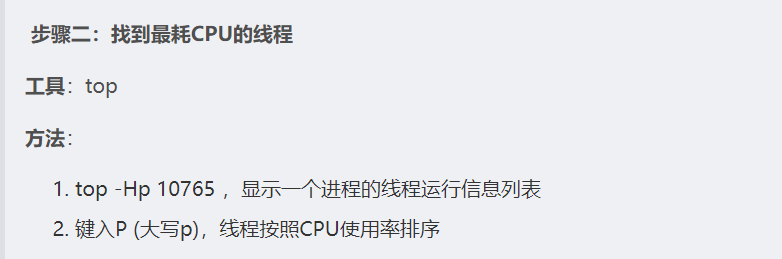
工具：top

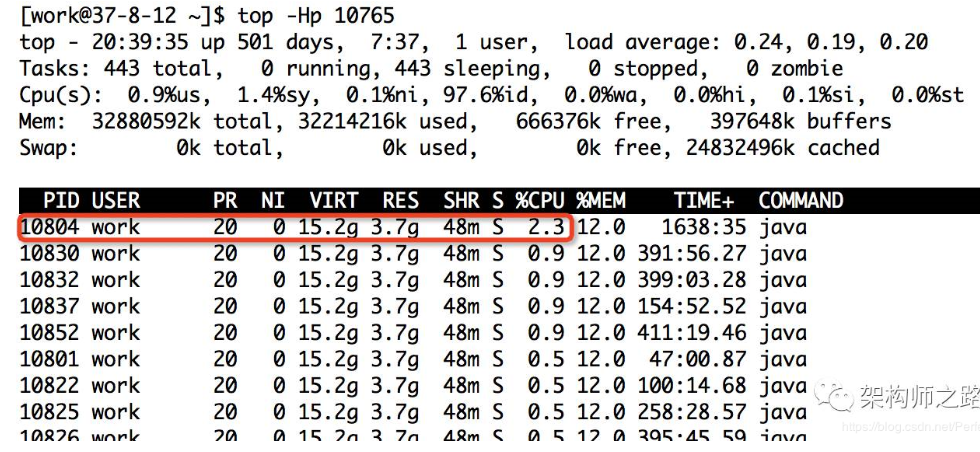
方法：

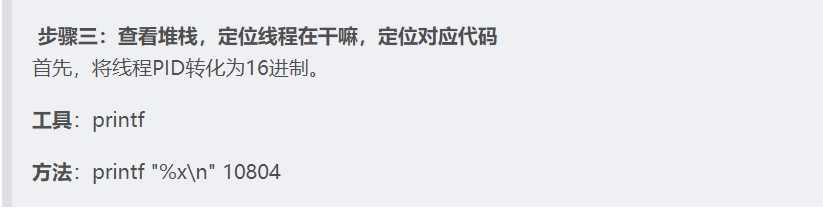
* 执行top -d 1 -c，每秒刷新一次，显示进程运行信息列表 **执行top -c ，显示进程运行信息列表**
* 键入P (大写p)，进程按照CPU使用率排序

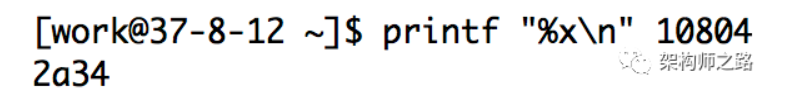


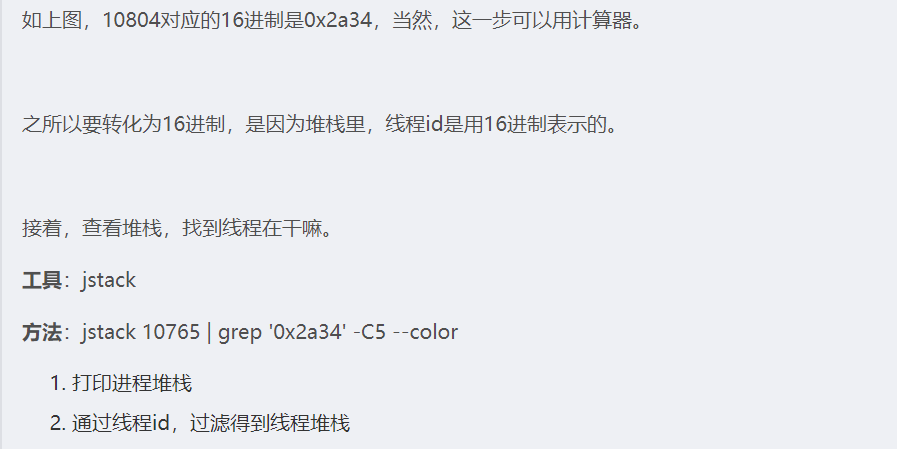
如上图，最耗CPU的进程PID为10765











**步骤四：查看堆栈，找到线程在干嘛**

工具：jstack/grep

方法：jstack 1802 | grep ‘722’ -C5

* 打印进程堆栈
* 通过线程id，过滤得到线程堆栈



如上图，找到了耗CPU高的线程对应的线程名称“http-nio-10287-exec-10”，以及看到了该线程正在执行代码的堆栈。