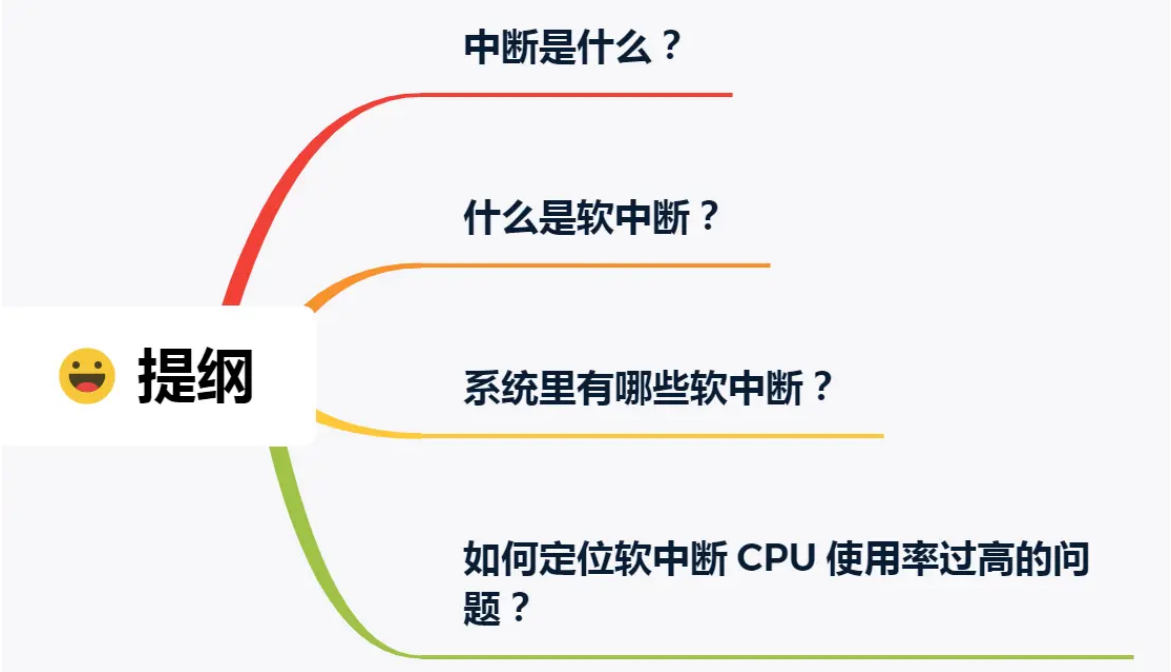
2.6 什么是软中断?

今日的技术主题：**什么是软中断?** 。



中断是什么?

先来看看什么是中断？在计算机中，中断是系统用来响应硬件设备请求的一种机制，操作系统收到硬件的中断请求，会打断正在执行的进程，然后调用内核中的中断处理程序来响应请求。

这样的解释可能过于学术了，容易云里雾里，我就举个生活中取外卖的例子。

小林中午搬完砖，肚子饿了，点了份白切鸡外卖，这次我带闪了，没有被某团大数据杀熟。虽然平台上会显示配送进度，但是我也不能一直傻傻地盯着呀，时间很宝贵，当然得去干别的事情，等外卖到了配送员会通过「电话」通知我，电话响了，我就会停下手中地事情，去拿外卖。

这里的打电话，其实就是对应计算机里的中断，没接到电话的时候，我可以做其他只有接到了电话，也就是发生中断，我才会停下当前的事情，去进行另一个事情，外卖。

操作系统收到了中断请求，会打断其他进程的运行，所以**中断请求的响应程序，也就是中断处理程序，要尽可能快的执行完，这样可以减少对正常进程运行调度地影响。**

而且，中断处理程序在响应中断时，可能还会「临时关闭中断」，这意味着，如果当前中断处理程序没有执行完之前，系统中其他的中断请求都无法被响应，也就说中断有可能会丢失，所以中断处理程序要短且快。

还是回到外卖的例子，小林到了晚上又点起了外卖，这次为了犒劳自己，共点了两份外卖，一份小龙虾和一份奶茶，并且是由不同地配送员来配送，那么问题来了，当第一份外卖送到时，配送员给我打了长长的电话，说了一些杂七杂八的事情，比如给个好评等等，但如果这时另一位配送员也想给我打电话。

很明显，这时第二位配送员因为我在通话中（相当于关闭了中断响应），自然就无法打通我的电话，他可能尝试了几次后就走掉了（相当于丢失了一次中断）。

什么是软中断?

前面我们也提到了，中断请求的处理程序应该要短且快，这样才能减少对正常进程运行调度地影响，而且中断处理程序可能会暂时关闭中断，这时如果中断处理程序执行时间过长，可能在还未执行完中断处理程序前，会丢失当前其他设备的中断请求。

那 Linux 系统**为了解决中断处理程序执行过长和中断丢失的问题，将中断过程分成了两个阶段，分别是「上半部和下半部分」**。

- **上半部用来快速处理中断**，一般会暂时关闭中断请求，主要负责处理跟硬件紧密相关或者时间敏感的事情。
- **下半部用来延迟处理上半部未完成的工作**，一般以「内核线程」的方式运行。

前面的外卖例子，由于第一个配送员长时间跟我通话，则导致第二位配送员无法拨通我的电话，其实当我接到第一位配送员的电话，可以告诉配送员说我现在下楼，剩下的事情，等我们见面再说（上半部），然后就可以挂断电话，到楼下后，在拿外卖，以及跟配送员说其他的事情（下半部）。

这样，第一位配送员就不会占用我手机太多时间，当第二位配送员正好过来时，会有很大几率拨通我的电话。

再举一个计算机中的例子，常见的网卡接收网络包的例子。

网卡收到网络包后，通过 DMA 方式将接收到的数据写入内存，接着会通过**硬件中**

目录 侧边栏 夜间 技术群 资料 支持我 上一篇 下一篇

上部分要做的事情很少，会先禁止网卡中断，避免频繁硬中断，而降低内核的工作效率。接着，内核会触发一个**软中断**，把一些处理比较耗时且复杂的事情，交给「软中断处理程序」去做，也就是中断的下半部，其主要是需要从内存中找到网络数据，再按照网络协议栈，对网络数据进行逐层解析和处理，最后把数据送给应用程序。

所以，中断处理程序的上部分和下半部可以理解为：

- **上半部直接处理硬件请求，也就是硬中断**，主要是负责耗时短的工作，特点是快速执行；
- **下半部是由内核触发，也就说软中断**，主要是负责上半部未完成的工作，通常都是耗时比较长的事情，特点是延迟执行；

还有一个区别，硬中断（上半部）是会打断 CPU 正在执行的任务，然后立即执行中断处理程序，而软中断（下半部）是以内核线程的方式执行，并且每一个 CPU 都对应一个软中断内核线程，名字通常为「ksoftirqd/CPU 编号」，比如 0 号 CPU 对应的软中断内核线程的名字是 `ksoftirqd/0`

不过，软中断不只是包括硬件设备中断处理程序的下半部，一些内核自定义事件也属于软中断，比如内核调度等、RCU 锁（内核里常用的一种锁）等。

系统里有哪些软中断?

在 Linux 系统里，我们可以通过查看 `/proc/softirqs` 的内容来知晓「软中断」的运行情况，以及 `/proc/interrupts` 的内容来知晓「硬中断」的运行情况。

接下来，就来简单的解析下 `/proc/softirqs` 文件的内容，在我服务器上查看到的文件内容如下：

目录
侧边栏
夜间
技术群
资料
支持我
上一篇
下一篇

```
$ cat /proc/softirqs
                    CPU0       CPU1       CPU2       CPU3
          HI:           0          0          0          0
       TIMER:   493685121  462502251  464481108  457141523
      NET_TX:   669467141   41820907   52777974   47467679
      NET_RX:   237983180  786176603 1521503296 3393422033
       BLOCK:        9586   35962255         18          1
BLOCK_IOPOLL:           0          0          0          0
     TASKLET:   448616917   26303934   33607570   30310010
       SCHED:   159888159  166811996  170089808  142212962
     HRTIMER:     1298467    1304037    1179765    1050725
         RCU:   779556545  617440054  561737120  529046379
```

你可以看到，每一个 CPU 都有自己对应的不同类型软中断的**累计运行次数**，有 3 点需要注意下。

第一点，要注意第一列的内容，它是代表着软中断的类型，在我的系统里，软中断包括了 10 个类型，分别对应不同的工作类型，比如 `NET_RX` 表示网络接收中断， `NET_TX` 表示网络发送中断、 `TIMER` 表示定时中断、 `RCU` 表示 RCU 锁中断、 `SCHED` 表示内核调度中断。

第二点，要注意同一种类型的软中断在不同 CPU 的分布情况，正常情况下，同一种中断在不同 CPU 上的累计次数相差不多，比如我的系统里， `NET_RX` 在 CPU0 、CPU1、CPU2、CPU3 上的中断次数基本是同一个数量级，相差不多。

第三点，这些数值是系统运行以来的累计中断次数，数值的大小没什么参考意义，但是系统的**中断次数的变化速率**才是我们要关注的，我们可以使用 `watch -d cat /proc/softirqs` 命令查看中断次数的变化速率。

前面提到过，软中断是以内核线程的方式执行的，我们可以用 `ps` 命令可以查看到，下面这个就是在我的服务器上查到软中断内核线程的结果：

```
$ ps aux | grep softirq
root         4  0.0  0.0      0     0 ?        S    Nov28   3:09 [ksoftirqd/0]
root         9  0.0  0.0      0     0 ?        S    Nov28   1:42 [ksoftirqd/1]
root        13  0.0  0.0      0     0 ?        S    Nov28   0:42 [ksoftirqd/2]
root        17  0.0  0.0      0     0 ?        S    Nov28   0:35 [ksoftirqd/3]
root     28380  0.0  0.0 109168   876 pts/1    S+   04:00   0:00 grep softirq
```

可以发现，内核线程的名字外面都有有中括号，这说明 ps 无法获取它们的命令行参
以一般来说，名字在中括号里的都可以认为是内核线程。

如何定位软中断 CPU 使用率过高的问题?

要想知道当前的系统的软中断情况，我们可以使用 `top` 命令查看，下面是一台服务器上的 top 的数据：

```
# top 运行后按数字 1 ,即可显示所有 CPU 核心
$ top
top - 17:50:58 up 3 days, 12:10,  1 user,  load average: 0.00, 0.00, 0.00
Tasks: 122 total,   1 running,  71 sleeping,   0 stopped,   0 zombie
%Cpu0  :  0.0 us,  0.0 sy,  0.0 ni, 96.7 id,  0.0 wa,  0.0 hi,  3.3 si,  0.0 st
%Cpu1  :  0.0 us,  0.0 sy,  0.0 ni, 95.6 id,  0.0 wa,  0.0 hi,  4.4 si,  0.0 st
...
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    7 root      20   0       0      0      0 S   0.3  0.0   0:01.64 ksoftirqd/0
   16 root      20   0       0      0      0 S   0.3  0.0   0:01.97 ksoftirqd/1
...
```

上图中的黄色部分 `si` ，就是 CPU 在软中断上的使用率，而且可以发现，每个 CPU 使用率都不高，两个 CPU 的使用率虽然只有 3% 和 4% 左右，但是都是用在软中断上了。

另外，也可以看到 CPU 使用率最高的进程也是软中断 `ksoftirqd` ，因此可以认为此时系统的开销主要来源于软中断。

如果要知道是哪种软中断类型导致的，我们可以使用 `watch -d cat /proc/softirqs` 命令查看每个软中断类型的中断次数的变化速率。

```
Every 2.0s: cat /proc/softirqs

                    CPU0       CPU1       CPU2       CPU3
          HI:          0          0          0          0
       TIMER:   49584326   46432325   46662048   45966802
      NET_TX:   67111373    4182095    5277903    4746768
      NET_RX:   24692151   79487637  153705558  341012307
       BLOCK:       9586    3603899         18          1
BLOCK_IOPOLL:          0          0          0          0
     TASKLET:   44940352    2630393    3360757    3031001
       SCHED:   16046774   16742677   17068908   14259575
     HRTIMER:     130206     130922     118286     105320
         RCU:   78236495   61967224   56415017   53178901
```

如果发现 `NET_RX` 网络接收中断次数的变化速率过快，接下来就可以使用 `sar -n DEV` 查看网卡的网络包接收速率情况，然后分析是哪个网卡有大量的网络包进来。

```
$ sar -n DEV 1
04:41:31 PM     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s
04:41:32 PM        lo      0.00      0.00      0.00      0.00      0.00      0.00      0.00
04:41:32 PM      eth0   1289.13   6706.52     85.57   9923.21      0.00      0.00      0.00
04:41:32 PM      eth1  61951.09      0.00  82156.42      0.00      0.00      0.00 123892.39
04:41:32 PM      eth2      1.09      0.00      0.07      0.00      0.00      0.00      0.00
04:41:32 PM      eth3      0.00      0.00      0.00      0.00      0.00      0.00      0.00
04:41:32 PM      eth4      0.00      0.00      0.00      0.00      0.00      0.00      0.00
04:41:32 PM      eth5      0.00      0.00      0.00      0.00      0.00      0.00      0.00
```

接着，在通过 `tcpdump` 抓包，分析这些包的来源，如果是非法的地址，可以考虑加防火墙，如果是正常流量，则要考虑硬件升级等。

总结

为了避免由于中断处理程序执行时间过长，而影响正常进程的调度，Linux 将中断处理程序分为上半部和下半部：

- 上半部，对应硬中断，由硬件触发中断，用来快速处理中断；
- 下半部，对应软中断，由内核触发中断，用来异步处理上半部未完成的工作；

Linux 中的软中断包括网络收发、定时、调度、RCU 锁等各种类型，可以通过查看 /proc/softirqs 来观察软中断的累计中断次数情况，如果要实时查看中断次数的变化率，可以使用 watch -d cat /proc/softirqs 命令。

每一个 CPU 都有各自的软中断内核线程，我们还可以用 ps 命令来查看内核线程，一般名字在中括号里面到，都认为是内核线程。

如果在 top 命令发现，CPU 在软中断上的使用率比较高，而且 CPU 使用率最高的进程也是软中断 ksoftirqd 的时候，这种一般可以认为系统的开销被软中断占据了。

这时我们就可以分析是哪种软中断类型导致的，一般来说都是因为网络接收软中断导致的，如果是的话，可以用 sar 命令查看是哪个网卡的有大量的网络包接收，再用 tcpdump 抓网络包，做进一步分析该网络包的源头是不是非法地址，如果是就需要考虑防火墙增加，如果不是，则考虑硬件升级等。

目录
侧边栏
夜间
技术群
资料
支持我
上一篇
下一篇

关注作者

哈喽，我是小林，就爱图解计算机基础，如果觉得文章对你有帮助，欢迎微信搜索「小林coding」，关注后，回复「网络」再送你图解网络 PDF

上次更新: 9/16/2022, 6:14:00 PM

← 2.5 CPU 是如何执行任务的?

2.7 为什么 0.1 + 0.2 不等于 0.3？ →

评论

Powered by GitHub & Vssue

发现新内容可用

刷新

使用 GitHub 帐号登录后发表评论

使用 GitHub 登录

登录后查看评论