load Average

   1.1：   
   Load 是对系统工作量的度量，简单的说是进程队列的长度。Load Average 就是一段时间（1分钟、5分钟、15分钟）内平均Load。

   1.2：查看指令：  
   w or uptime or top  
     
   load average: 0.02,   0.27,    0.17  
   1 per/minute 5 per/minute 15 per/minute

1.3：如何判断系统是否已经Over Load？  
对一般的系统来说，根据cpu数量去判断。如果平均负载始终在1.2一下，而你有2颗cup的机器。那么基本不会出现cpu不够用的情况。也就是Load平均要小于Cpu的数量  
1.4：Load与容量规划（Capacity Planning）  
       一般是会根据15分钟那个load 平均值为首先。

1.5：Load误解：  
1：系统load高一定是性能有问题。  
    真相：Load高也许是因为在进行cpu密集型的计算  
        2：系统Load高一定是CPU能力问题或数量不够。  
    真相：Load高只是代表需要运行的队列累计过多了。但队列中的任务实际可能是耗Cpu的，也可能是耗i/0及其他因素的。  
3：系统长期Load高，首先增加CPU  
    真相：Load只是表象，不是实质。增加CPU个别情况下会临时看到Load下降，但治标不治本。

2：在Load average 高的情况下如何鉴别系统瓶颈。  
   是CPU不足，还是io不够快造成或是内存不足？

查看系统负载vmstat

procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----  
r b swpd free buff cache si so bi bo in cs us sy id wa  
0 0 100152 2436 97200 289740 0 1 34 45 99 33 0 0 99 0

procs  
r 列表示运行和等待cpu时间片的进程数，如果长期大于1，说明cpu不足，需要增加cpu。  
b 列表示在等待资源的进程数，比如正在等待I/O、或者内存交换等。  
cpu 表示cpu的使用状态  
us 列显示了用户方式下所花费 CPU 时间的百分比。us的值比较高时，说明用户进程消耗的cpu时间多，但是如果长期大于50%，需要考虑优化用户的程序。  
sy 列显示了内核进程所花费的cpu时间的百分比。这里us + sy的参考值为80%，如果us+sy 大于 80%说明可能存在CPU不足。  
wa 列显示了IO等待所占用的CPU时间的百分比。这里wa的参考值为30%，如果wa超过30%，说明IO等待严重，这可能是磁盘大量随机访问造成的，也可能磁盘或者磁盘访问控制器的带宽瓶颈造成的(主要是块操作)。  
id 列显示了cpu处在空闲状态的时间百分比  
system 显示采集间隔内发生的中断数  
in 列表示在某一时间间隔中观测到的每秒设备中断数。  
cs列表示每秒产生的上下文切换次数，如当 cs 比磁盘 I/O 和网络信息包速率高得多，都应进行进一步调查。  
memory  
swpd 切换到内存交换区的内存数量(k表示)。如果swpd的值不为0，或者比较大，比如超过了100m，只要si、so的值长期为0，系统性能还是正常  
free 当前的空闲页面列表中内存数量(k表示)  
buff 作为buffer cache的内存数量，一般对块设备的读写才需要缓冲。  
cache: 作为page cache的内存数量，一般作为文件系统的cache，如果cache较大，说明用到cache的文件较多，如果此时IO中bi比较小，说明文件系统效率比较好。  
swap  
si 由内存进入内存交换区数量。  
so由内存交换区进入内存数量。  
IO  
bi 从块设备读入数据的总量（读磁盘）（每秒kb）。  
bo 块设备写入数据的总量（写磁盘）（每秒kb）  
这里我们设置的bi+bo参考值为1000，如果超过1000，而且wa值较大应该考虑均衡磁盘负载，可以结合iostat输出来分析。

 查看磁盘负载iostat

每隔2秒统计一次磁盘IO信息，直到按Ctrl+C终止程序，-d 选项表示统计磁盘信息， -k 表示以每秒KB的形式显示，-t 要求打印出时间信息，2 表示每隔 2 秒输出一次。第一次输出的磁盘IO负载状况提供了关于自从系统启动以来的统计信息。随后的每一次输出则是每个间隔之间的平均IO负载状况。

# iostat -x 1 10  
Linux 2.6.18-92.el5xen 02/03/2009  
avg-cpu:   %user %nice %system %iowait   %steal %idle  
            1.10 0.00 4.82 39.54 0.07 54.46  
Device:       rrqm/s wrqm/s r/s w/s rsec/s wsec/s avgrq-sz avgqu-sz await   svctm   %util  
   sda             0.00     3.50   0.40   2.50     5.60 48.00 18.48     0.00 0.97 0.97 0.28  
   sdb             0.00     0.00   0.00   0.00     0.00     0.00     0.00     0.00 0.00 0.00 0.00  
   sdc             0.00     0.00   0.00   0.00     0.00     0.00     0.00     0.00 0.00 0.00 0.00  
   sdd             0.00     0.00   0.00   0.00     0.00     0.00     0.00     0.00 0.00 0.00 0.00  
   sde             0.00     0.10   0.30   0.20     2.40     2.40     9.60     0.00 1.60 1.60 0.08  
   sdf              17.40     0.50 102.00   0.20 12095.20     5.60 118.40     0.70 6.81 2.09   21.36  
   sdg          232.40     1.90 379.70   0.50 76451.20 19.20 201.13     4.94 13.78 2.45   93.16  
   rrqm/s: 每秒进行 merge 的读操作数目。即 delta(rmerge)/s  
   wrqm/s:   每秒进行 merge 的写操作数目。即 delta(wmerge)/s  
   r/s:           每秒完成的读 I/O 设备次数。即 delta(rio)/s  
   w/s:       每秒完成的写 I/O 设备次数。即 delta(wio)/s  
   rsec/s: 每秒读扇区数。即 delta(rsect)/s  
   wsec/s: 每秒写扇区数。即 delta(wsect)/s  
   rkB/s:   每秒读K字节数。是 rsect/s 的一半，因为每扇区大小为512字节。(需要计算)  
   wkB/s: 每秒写K字节数。是 wsect/s 的一半。(需要计算)  
   avgrq-sz: 平均每次设备I/O操作的数据大小 (扇区)。delta(rsect+wsect)/delta(rio+wio)  
   avgqu-sz: 平均I/O队列长度。即 delta(aveq)/s/1000 (因为aveq的单位为毫秒)。  
   await: 平均每次设备I/O操作的等待时间 (毫秒)。即 delta(ruse+wuse)/delta(rio+wio)  
   svctm: 平均每次设备I/O操作的服务时间 (毫秒)。即 delta(use)/delta(rio+wio)  
   %util:    一秒中有百分之多少的时间用于 I/O 操作，或者说一秒中有多少时间 I/O 队列是非空的。即 delta(use)/s/1000 (因为use的单位为毫秒)  
    
   如果 %util 接近 100%，说明产生的I/O请求太多，I/O系统已经满负荷，该磁盘  
   可能存在瓶颈。  
   idle小于70% IO压力就较大了,一般读取速度有较多的wait.  
    
   同时可以结合vmstat 查看查看b参数(等待资源的进程数)和wa参数(IO等待所占用的CPU时间的百分比,高过30%时IO压力高)  
    
   另外还可以参考  
   一般:  
   svctm < await (因为同时等待的请求的等待时间被重复计算了)，  
   svctm的大小一般和磁盘性能有关:CPU/内存的负荷也会对其有影响，请求过多也会间接导致 svctm 的增加。  
   await: await的大小一般取决于服务时间(svctm) 以及 I/O 队列的长度和 I/O 请求的发出模式。  
   如果 svctm 比较接近 await，说明I/O 几乎没有等待时间；  
   如果 await 远大于 svctm，说明 I/O队列太长，应用得到的响应时间变慢，  
   如果响应时间超过了用户可以容许的范围，这时可以考虑更换更快的磁盘，调整内核 elevator算法，优化应用，或者升级 CPU。  
   队列长度(avgqu-sz)也可作为衡量系统 I/O 负荷的指标，但由于 avgqu-sz 是按照单位时间的平均值，所以不能反映瞬间的 I/O 洪水。  
    
     别人一个不错的例子.(I/O 系统 vs. 超市排队)  
   举一个例子，我们在超市排队 checkout 时，怎么决定该去哪个交款台呢? 首当是看排的队人数，5个人总比20人要快吧?除了数人头，我们也常常看看前面人购买的东西多少，如果前面有个采购了一星期食品的大妈，那么可以考虑换个队排了。还有就是收银员的速度了，如果碰上了连钱都点不清楚的新手，那就有的等了。另外，时机也很重要，可能 5分钟前还人满为患的收款台，现在已是人去楼空，这时候交款可是很爽啊，当然，前提是那过去的 5 分钟里所做的事情比排队要有意义(不过我还没发现什么事情比排队还无聊的)。  
   I/O 系统也和超市排队有很多类似之处:  
   r/s+w/s 类似于交款人的总数  
   平均队列长度(avgqu-sz)类似于单位时间里平均排队人的个数  
   平均服务时间(svctm)类似于收银员的收款速度  
   平均等待时间(await)类似于平均每人的等待时间  
   平均I/O数据(avgrq-sz)类似于平均每人所买的东西多少  
   I/O 操作率 (%util)类似于收款台前有人排队的时间比例。  
   我们可以根据这些数据分析出 I/O 请求的模式，以及 I/O 的速度和响应时间。  
   下面是别人写的这个参数输出的分析  
   # iostat -x 1  
   avg-cpu:   %user %nice %sys %idle  
   16.24 0.00 4.31 79.44  
   Device: rrqm/s wrqm/s r/s w/s   rsec/s   wsec/s rkB/s wkB/s avgrq-sz avgqu-sz await   svctm   %util  
   /dev/cciss/c0d0  
   0.00   44.90   1.02 27.55 8.16   579.59     4.08 289.80 20.57 22.35 78.21 5.00   14.29  
   /dev/cciss/c0d0p1  
   0.00   44.90   1.02 27.55 8.16   579.59     4.08 289.80 20.57 22.35 78.21 5.00   14.29  
   /dev/cciss/c0d0p2  
   0.00 0.00   0.00   0.00 0.00 0.00     0.00     0.00     0.00     0.00 0.00 0.00 0.00  
   上面的 iostat 输出表明秒有 28.57 次设备 I/O 操作: 总IO(io)/s = r/s(读) +w/s(写) = 1.02+27.55 = 28.57 (次/秒) 其中写操作占了主体 (w:r = 27:1)。  
   平均每次设备 I/O 操作只需要 5ms 就可以完成，但每个 I/O 请求却需要等上 78ms，为什么? 因为发出的 I/O 请求太多 (每秒钟约 29 个)，假设这些请求是同时发出的，那么平均等待时间可以这样计算:  
   平均等待时间 = 单个 I/O 服务时间 \* ( 1 + 2 + ... + 请求总数-1) / 请求总数  
   应用到上面的例子: 平均等待时间 = 5ms \* (1+2+...+28)/29 = 70ms，和 iostat 给出的78ms 的平均等待时间很接近。这反过来表明 I/O 是同时发起的。  
   每秒发出的 I/O 请求很多 (约 29 个)，平均队列却不长 (只有 2 个 左右)，这表明这 29 个请求的到来并不均匀，大部分时间 I/O 是空闲的。  
   一秒中有 14.29% 的时间 I/O 队列中是有请求的，也就是说，85.71% 的时间里 I/O 系统无事可做，所有 29 个 I/O 请求都在142毫秒之内处理掉了。  
   delta(ruse+wuse)/delta(io) = await = 78.21 => delta(ruse+wuse)/s=78.21 \* delta(io)/s = 78.21\*28.57 =2232.8，表明每秒内的I/O请求总共需要等待2232.8ms。所以平均队列长度应为 2232.8ms/1000ms = 2.23，而iostat 给出的平均队列长度 (avgqu-sz) 却为 22.35，为什么?! 因为 iostat 中有 bug，avgqu-sz值应为 2.23，而不是 22.35。