使用RRDtool 进行绘图  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
注 ：该教程参考了如下内容 ：  
A）官方文档 ：http://oss.oetiker.ch/rrdtool/doc/index.en.html  
B）abel 兄的大作 ：http://bbs.chinaunix.net/viewthread.php?tid=552224&highlight=rrdtool  
                http://bbs.chinaunix.net/viewthread.php?tid=552220&highlight=rrdtool  
作者 ：ailms <ailms{@}263{dot}net>  
版本 ：v1  
最后修改 ：2006/11/17 17:35  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  
**一）前言**  
  
使用RRDtool 我们最关心什么？当然是如何把数据画出来了。虽然前面谈了很多，但这些都是基础来的。掌握好了，可以让你在绘图时更加得心应手。  
  
本来还有 RPN （反向波兰表达式）一节的，但考虑一下，觉得还是放到后面，先从基本的绘图讲起。  
  
这一节的内容虽然很多，但基本都是实验性的内容，只要多试几次就可以了。  
  
**二、graph 操作的语法**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

rrdtool graph filename [option ...]   
   [data definition ...]  
   [data calculation ...]          
   [variable definition ...]  
   [graph element ...]  
   [print element ...]

其中的 data definiton、variable definition 、data calculation、分别是下面的格式

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

DEF:<vname>=<rrdfile>:<ds-name>:<CF>[:step=<step>][:start=<time>][:end=<time>][:reduce=<CF>]  
VDEF:vname=RPN expression  
CDEF:vname=RPN expression

其中 filename 就是你想要生成的图片文件的名称，默认是 png 。你可以通过选项修改图片的类型，可以有 PNG、SVG、EPS、PDF四种。  
  
A）DEF 是 Definition （定义）的意思。定义什么呢？你要绘图，总要有数据源吧？DEF 就是告诉 RRDtool 从那个 RRD 中取出指定  
  
    DS（eth0\_in、eth0\_out）的某个类型的统计值（还可以指定 resolution、start、end），并把这一切放入到一个变量 <vname>中 。  
      
    你可能会感到奇怪，为什么还有一个 CF 字段？因为 RRA 有多种  CF 类型，有些 RRA 可能用来保存平均值、有些 RRA 可能用于统计最大值、  
      
    最小值等等。所以你必须同时指定使用什么 CF 类型的 RRA的数据。至于 :start 和 :end 、:reduce 则用得比较少，最常用的就是 :step 了，  
      
    它可以让你控制 RRDtool 从那个 RRA 中取数据。  
  
B）VDEF 是 Variable Definition （变量定义）的意思。定义什么呢？记得 MRTG 在图表的下面有一个称之为 Legend 的部分吗？  
  
    那里显示了1个或者2个 DS （MRTG 没有 DS 一说，这里是借用 RRDtool 的）的 “最大值”、“平均值”、“当前值”。这些值是如何  
      
    计算的呢？  
      
    RRDtool 中用 VDEF 来定义。这个变量专门存放某个 DS 某种类型的值，例如 eth0\_in 的最大值、eht0\_out 的当前值等。当你需要象  
      
     MRTG  一样输出数字报表（Legend） 时，就可以在 GPRINT 子句（sub clause）中调用它。  
     
   同样它也需要用一个变量来存放数值。要注意的是，旧版 的 RRDtool 中是用另外一种格式来达到相同的目的。新版的 RRDtool 则推荐使用  
     
    VDEF   语句。但在使用过程中，却发现 VDEF 的使用反而造成了困扰。 例如你有5个 DS 要画，每个 DS 你都想输出最大值、最小值、平均值  
      
    、当前值。  如果使用 VDEF ，则需要 4 \* 5 = 20 个 VDEF 语句，这会造成极大的困扰。具体例子可以看第十一节“数字报表”部分。  
     
C）CDEF 是 Calculation Define 的意思。使用过MRTG 的都会体会到一点，MRTG 的计算能力实在太差了。例如你有两个 Target ，  
  
      一个是 eth0\_in ， 一个是 eth0\_out,如果要把它们相加起来，再除以8，得出 bytes 为单位的值，如何计算呢？或者说你只想看  
        
      eth0\_in 中超过 10Mb/s 的那部分， 其余的不关心，又如何实现呢？因为 MRTG 不能对它从 log 取出来的数据进行修改，只能原  
        
      原本本的表现，所以很难满足我们的要求。而使用 CDEF ， 这一切都可以很容易的实现。CDEF 支持很多数学运算，甚至还支持简  
        
      单的逻辑运算 if-then-else ，可以解决前面提到的第2个问题：如何只绘制你所关 心的数据。不过这一切都需要熟悉 RPN 的语法，  
        
      所以我们放到下一节介绍，这一节就介绍如何把 RRDtool 中的数据以图表的方式显示出来。  
        
        
**三）选项分类**  
  
本部分我们按照官方文档的方式，把选项分成几大类，分为  ：  
  
A）**Time range ：** 用于控制图表的X轴显示的起始/结束时间，也包括从RRA中提取指定时间的数据。  
  
B）**Labels ：**用于控制 X/Y 轴的说明文字。  
  
C）**Size ：**用于控制图片的大小。  
  
D）**Limits ：**用于控制 Y 轴的上下限。  
  
E）**Grid ：**用于控制 X/Y 轴的刻度如何显示。  
  
F）**Miscellaneous ：**其他选项。例如显示中文、水印效果等等。  
  
G）**Report ：**数字报表  
  
需要说明的是，本篇当中并不是列出了所有选项的用法，只是列出较为常用的选项，如果想查看所有选项的的用法，可以到官方站点下载文档，  
  
这里就不一一列出了，望各位见谅。  
  
  
**四）时间范围控制（Time Range）**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

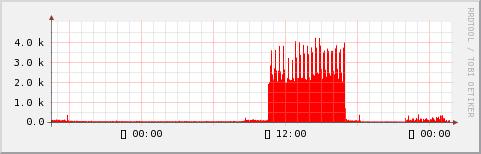
[-s|--start time] [-e|--end time] [-S|--step seconds]

既然要绘图，就应该有一个起始/结束的时间。Graph 操作中有 –s ，-e 选项。这两个选项即可以用于控制图表的 X 轴显示的时间范围，也可以用  
  
于控制 RRDtool 从 RRA 中提取对应时间的数据。如果没有指定 –-end ，默认为 now；如果没有指定 –-start，则默认为1天前。如果两者都没有  
  
指定，则图表默认显示从当前算起1天内的。数  
回头看一下 DEF 中，也有 :start ,:end , :step ，这些和 –-start、--end、--step 之间有什么关系呢？  
  
让我们先看 :step 和 –step之间的关系是如何的。  
  
下面以 eth0.rrrd 为例，假设要绘制的时间范围 **range 等于 end -start[**  
  
A）如果 0 <  ragne < 180000 （第一个 RRA 的时间覆盖范围） ，则默认从第1个RRA中取数据 ：  
  
     如果 DEF 中给出的 :step > 300 ，例如 1000 ，则从 resolution= 1000 的或者第一个高于 1000 的RRA 中挑选数据，由于 eth0.rrd 中没有  
      
                                                     resolution = 1000 的 RRA，则 RRDtool 会从 resolution = 1200 的第2 RRA 中取数据。  
  
     如果 DEF 中给出的 :step <= 300 ，例如 200 ，则 RRDtool 会忽略该设定，还是从第一个 RRA 中取数据。  
  
B）如果 180000 < range < 720000  ，由于第一个RRA只能保存2天的数据，所以默认从第2个RRA中取数据 ：  
  
     如果 DEF 中给出的 :step > 1200 ，例如 1800，则 RRDtool 会从 resolution = 7200 的第3 RRA 中取数据  
  
     如果 DEF 中给出 :step<= 1200，例如 300 ，则 RRDtool 会忽略，还是从第2个 RRA 中取数据  
  
C）如果 720000 < range <  4320000 ，则默认从第三个 RRA 中取数据 ：  
  
     如果 DEF 中给出的 :step > 7200 ，例如 10000 ，则从第4个 RRA中取数据  
  
     如果 DEF 中给出的 :step <= 7200 ， 例如 1200 ，则忽略该值，并还是从第3 个 RRA 中取数据  
  
D）如果 4320000 < range < 63072000 ，则默认从第4个 RRA 中取数据 ：  
  
     如果 DEF 中给出的 :step > 86400 ，则行为未知  
  
     如果 DEF 中给出的 :step <=86400 ，则从第4个 RRA 中取数据  
  
E）-S 选项可以直接控制 RRDtool 如何挑选 RRA 。  
  
    例如 -S 1200 ，即使DEF 中不加 :setp ，则 RRDtool 会从第2个 RRA 中取数据，即使 rang < 180000   
  
    如果 -S 和 :step 同时出现，则 :step 优先。  
  
F）DEF 中的 :start 和 :end 可以覆盖 –-start 和 –-end 的值。  
  
     默认情况下，如果 DEF 中不加 :start 和 :end ，则等于 –-start 和 –end  
  
    如果 DEF 中定义了 :start 和 :end ，则以 :start 和 :end 为准。  
  
**实例1 ：使用 –-start 指定 X 轴的起始时间**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

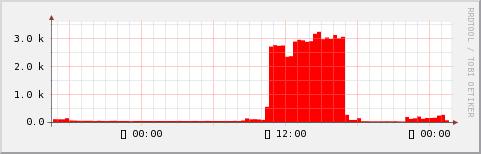
[root@dns1 bob]# rrdtool graph 1.png \          
> --start now-120000 \        # 表示起始时间是当前时间往前 120000 秒，也就是 33 个小时左右  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE \         # 从eth0.rrd 中取出 eth0\_in 的数据，CF 类型为 AVERAGE  
> AREA:value1#ff0000        # 用“方块”的形式来绘制 value1 ，注意这里是用 value1 ，不是用 eth0\_in  
481x154                        # 如果 RRDtool 有绘图方面的语句，则这里显示图片大小，否则为 0x0。  
[root@dns1 bob]#

  
  
可以看到 X 轴的文字有些是乱码，不过不要紧，你可以临时已用 env LANG=C rrdtool xxxx 来解决该问题，或者在后面用  
  
–n 来设定 RRDtool 使用中文字体，就不会出现这样的情况了  
  
  
**实例2 ：使用 :step 从第2个RRA中取数据**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

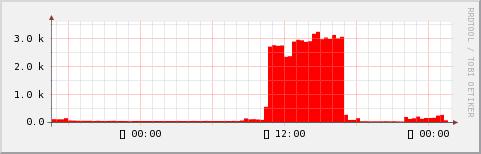
[root@dns1 bob]# rrdtool graph 2.png \  
> --start now-120000 \  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE:step=1000 \        # :step 指定 resolution=1000   
> AREA:value1#ff0000  
481x154  
[root@dns1 bob]#

这里是 :step=1000，但 RRDtool 会取 :step=1200 的 第2个 RRA 的数据来绘图,可以和上面的 1.png 比较，发现比较平滑。  
  
  
  
  
**实例3 ：使用 –S 从第2个RRA中取数据**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

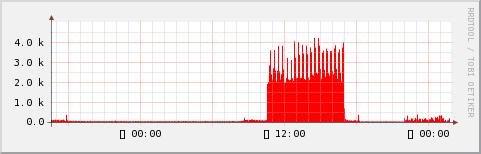
[root@dns1 bob]# rrdtool graph 4.png   
> -S 1200 \                # 使用 –S 控制 RRDtool 从 resolution=1200 的 RRA 中取数据  
> --start now-120000 \  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE \  
> AREA:value1#ff0000          
481x154  
[root@dns1 bob]#

  
  
可以看到和上面的图一样，说明 RRDtool 的确按照 -S 的设置从第2个RRA 中取数据了  
  
使用 –S 可以对 DEF 中所有的 DS 都使用相同的 resolution，等于在每个 DEF后都加上 :step=<value> ，value 是 –S 的值  
  
**实例4 ：同时使用 –S 和 :step**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

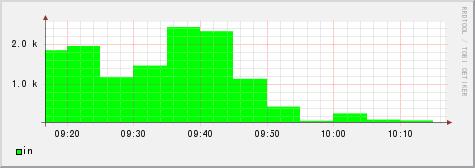
[root@dns1 bob]# rrdtool graph 5.png   
> -S 1200 \                                # -S 指定 resolution=1200  
> --start now-120000 \  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE:step=300 \        # :step 指定 resolution=300  
> AREA:value1#ff0000       
481x154  
[root@dns1 bob]#

  
  
  
可以看到 5.png 和 1.png 是一样的，也就是说 –S 1200 并没有起作用，而是 :step=300 起作用了  
  
**实例5 ：使用 :start 和 :end 只显示指定时间内的数据**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

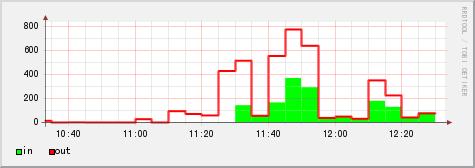
[root@dns1 bob]# rrdtool graph 1.png \  
> --start now-1h \                        # X 轴显示1个小时的长度  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE:start=now-600:end=now-300 \        # 但只取10分钟前到5分钟前的这部分  
> AREA:value1#00ff00:in    
475x168  
[root@dns1 bob]#

  
  
如果我们不加 :start 和 :end ，则效果如下 ：  
  
  
  
我们甚至可以让两个对象显示不同的时间，例如   
  
**实例6 ：让两个对象显示不同时间段的数据**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

[root@dns1 bob]# rrdtool graph 1.png \  
> --start now-2h \                # 规定时间为2小时内  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE:end=now:start=end-1h \        # 规定时间为1小时内                  
> DEF:value2=eth0.rrd:eth0\_out:AVERAGE \        # 没有指定 :start 和 :end，默认和 –-start 一样也是2小时  
> AREA:value1#00ff00:in \  
> LINE2:value2#ff0000:out:STACK   
475x168  
[root@dns1 bob]#

  
  
  
**实例7 ：把一段时间分为几段分别显示 ：**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864875###)

**CODE:**

[root@dns1 bob]# rrdtool graph 1.png \  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE:end=now:start=end-1h \        # 当前1小时内  
> DEF:value2=eth0.rrd:eth0\_in:AVERAGE:end=now-1h:start=now-2h \        # 2小时前  
> DEF:value3=eth0.rrd:eth0\_in:AVERAGE:end=now-2h:start=now-3h \        # 3小时前  
> LINE1:value1#00ff00:"1 hours ago" \  
> LINE2:value2#ff0000:"2 hours ago" \  
> LINE3:value3#000000:"3 hours ago"    
475x168  
[root@dns1 bob]

  
  
我们把3个小时内的数据用三种不同粗细、不同颜色的曲线画了出来。  
  
看到了吗，out部分（红色）显示了2个小时内的流量，而in部分（绿色）则只显示了1个小时内的部分  
  
在这里要提一点，虽然我们指定了 –-start 或者 –-end ,或者 :start , :end，但并不意味着曲线就一定会从指定的时间点开始和结束。  
  
例如我们上面指定了 :start=now-600:end=now-300 ，也就是只显示5分钟的数据。但图表出来的效果却是10(10:05-10:15)分钟  
  
的数据，这是因为我们挑选的时间当中“不慎”横垮了两个周期(10:05-10:10,10:10-10:15)，所以 RRDtool 会把它们全部画出来，而  
  
不是只画其中的5分钟。

**五）说明文字（Label）**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

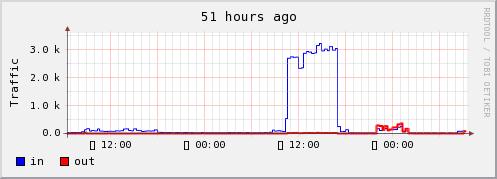
[-t|--title string] [-v|--vertical-label string]

-t 是用于图表上方的标题，-v 是用于 Y 轴的说明文字  
  
**实例1 ：给图表增加标题**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

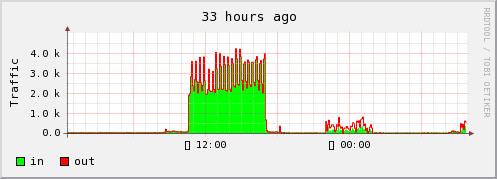
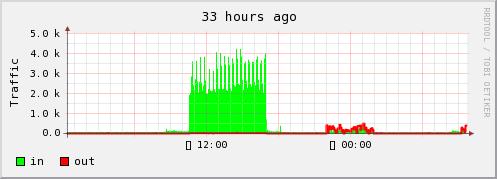
[root@dns1 bob]# rrdtool graph 1.png \  
> --start now-183600 \                # 从当前开始往前51个小时  
> -t "51 hours ago" -v "Traffic" \        # 标题是 “51 hours ago”，Y 轴的说明文字是 “Traffic”  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE \  
> DEF:value2=eth0.rrd:eth0\_out:AVERAGE \  
> LINE1:value1#0000ff:in \        # 注释 ：以1个像素宽的曲线画出 value1，颜色是蓝色，图例的说明文字是“in”  
> LINE2:value2#ff0000:out        # 注释 ：以2个像素宽的曲线画出 value2，颜色是红色，图例的说明文字是 “out”  
497x179  
[root@dns1 bob]#

  
  
现在我们用的是 LINE 的方式来绘图。LINE 可以有3种，分别是 LINE1|2|3,也就是线条的粗细。还有一种是 STACK 方式下面再介绍。  
  
可以看到流入的流量比流出的流量稍大，这样看的话，out 流量比较难看，是否可以有别的方式呢？RRDtool 还提供了  
  
另外一种格式，就是 STACK 。意思就是在前一个对象的基础（把前一个对象的值当成 X 轴）上绘图，而不是从 X 轴开始。  
  
**实例2 ：使用 STACK 方式绘图**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

[root@dns1 bob]# rrdtool graph 3.png \  
> --start now-120000 \  
> -t "33 hours ago" \  
> -v "Traffic" \  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE \  
> DEF:value2=eth0.rrd:eth0\_out:AVERAGE \  
> AREA:value1#00ff00:in \  
> LINE:value2#ff0000:out:STACK                # 注意最后的 “STACK” ，表示在 value1 的基础上绘图  
497x179  
[root@dns1 bob]#

  
  
  
这是没有采用 STACK 方式绘图的效果 ：  
  
  
  
可以看得出上面采用 STACK 方式的比较清晰，但要注意，采用 STACK 方式后，在读取 out 流量时，Y 轴的刻度不再是 out 的值，  
  
应该用刻度值减去 in 的值，才是 out 真正的值。这点比较麻烦。需要配合 GPRINT 语句才能达到一定的效果。  
  
**六）图表大小（Size）**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

[-w|--width pixels] [-h|--height pixels]

这里说图表大小而不是图片大小，是因为 –w ，-h 控制的是 X/Y 轴共同围起来的那部分大小，而不是整个图片的大小，这点在前面就可以看出了。  
  
默认的图表大小是 400 （长）x 100 （高），但一般会返回497x179 这样的数字，这个才是图片的大小。  
  
RRDtool 比 MRTG 好的一个地方就是 MRTG 一放大图片，就会变得模糊。RRDtool 则不会。  
  
在官方文档中，-w 似乎是一个比较敏感的参数，我们看下面的描述 ：

**QUOTE:**

First it makes sure that the RRA covers as much of the graphing time frame as possible. Second it looks at the resolution of the  
  
RRA compared to the resolution of the graph. It tries to find one which has the same or higher better resolution. With the ``-r''   
  
option you can force RRDtool to assume a different resolution than the one calculated from the pixel width of the graph.

**QUOTE:**

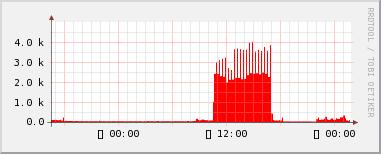
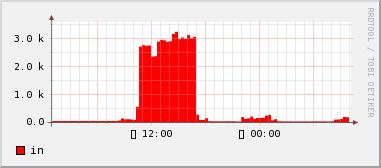
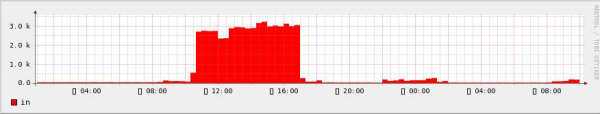
By default, rrdtool graph calculates the width of one pixel in the time domain and tries to get data from an RRA with that resolution.   
  
With the step option you can alter this behaviour. If you want rrdtool graph to get data at a one-hour resolution from the RRD, set   
  
step to 3'600. Note: a step smaller than one pixel will silently be ignored

这两段话分别是从rrd-beginners  和 rrd\_graph 文档中摘出来的。似乎看起来 –w 会影响到图表的 resolution ，进一步影响到 RRDtool 如何选择 RRA ，  
  
但经过实验却发现并非如此。  
  
我对这两段话中的图表的 resolution 一词不知如何理解和计算，希望好心的朋友能够指点一下 ^\_^ .  
  
  
**实例1 ：使用 –w 设定图表大小为 300 像素**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

[root@dns1 bob]# rrdtool graph 3.png   
> -w 300 \                                # 设定 size 为 300 pixel  
> --start now-120000 \  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE \  
> AREA:value1#ff0000             
381x154  
[root@dns1 bob]#

  
  
可以看到图表是不是变小了呢？而且整个图片的大小也变小了。  
  
如果用前面的话来推理，由于 120000/300（-w的值）= 400 > 300 （step）,由于没有 resolution=400 的 RRA，  
  
所以应该采用 resolution=7200 的第2个 RRA 的数据来绘图，但实际上不是。  
  
  
  
上面这个才是 300 pixel 宽，resolution=7200 的效果  
  
所以我觉得 –w 和 –h 并不能影响 RRDtool 如何选择 RRA ，只能起到缩小放大的作用。  
  
BTW：当你绘制的时间范围较大时，可以使用 –w 增大图表大小，这样看起来比较“舒服“  
  
  
  
  
**七） Y 轴上下限（Limits）**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

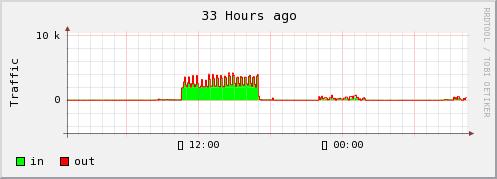
[-u|--upper-limit value] [-l|--lower-limit value] [-r|--rigid]

默认情况下，RRDtool 会和 MRTG 一样自动调整 Y 轴的数字，来配合当前的数值大小。如果想禁止该特性，可以通过 –upper-limit 和  
  
-–lower-limit 来做限制，表示Y轴显示的值从多少到多少。如果没有指定 –rigid ，则在图表的上下边界处还是会有一些延伸，但如果指定了  
    
-–rigid ，则严格按照 –-upper-limit 和 –-lower-limit 绘制。  
  
在使用 –lower-limit 时要注意，如果数据中有负数，如果 -–lower-limit 为 0，则负数部分是显示不出来的。  
  
**实例1 ：使用 –-upper-limit 和 –-lower-limit 限制 Y 轴的上下限**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

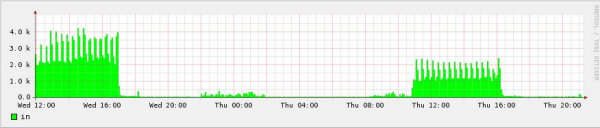
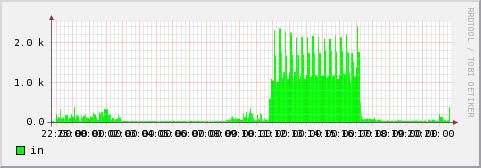
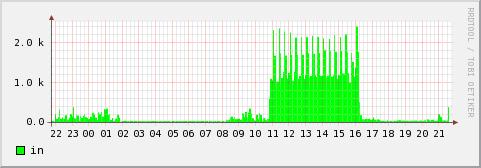
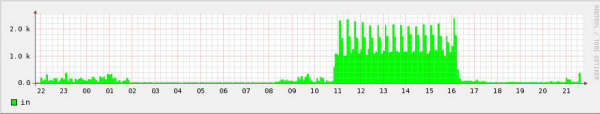
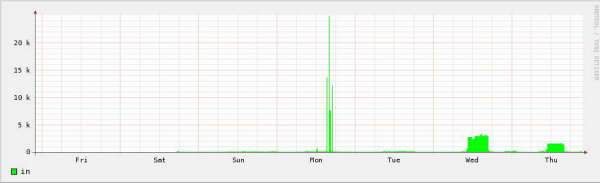
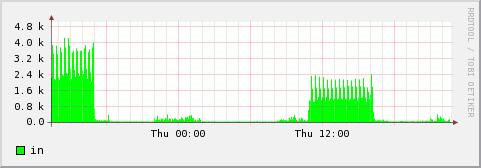
[root@dns1 bob]# rrdtool graph 1.png \  
> --start now-120000 \  
> -v "Traffic" -t "33 Hours ago" \  
> --lower-limit -5000 \                # 限制Y轴下限为 -5000   
> --upper-limit 10000 \                # 上限为 10000  
> --rigid \                                         # 严格按照上面的规定来画  
> DEF:value1=eth0.rrd:eth0\_in:AVERAGE \  
> DEF:value2=eth0.rrd:eth0\_out:AVERAGE \  
> AREA:value1#00ff00:in \  
> LINE1:value2#ff0000:out:STACK  
497x179  
[root@dns1 bob]#

  
  
  
**八） X/Y 轴刻度（Grid）**

[**[Copy to clipboard]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)[**[ - ]**](http://bbs.chinaunix.net/viewthread.php?tid=864876###)

**CODE:**

[-x|--x-grid GTM:GST:MTM:MST:LTM:LST:LPR:LFM]   
[-x|--x-grid none]  
[-y|--y-grid grid step:label factor]   
[-y|--y-grid none]  
[-Y|--alt-y-grid]  
[-X|--units-exponent value]

RRDtool 中设置 X 轴的刻度比较复杂，如果没有必要，可以交给 RRDtool 自动去处理。  
  
例如上面的图，33 小时的情况下，X 轴只有2个值，显得很不足。这时候有两种方法 ：  
  
A）一是使用 –w 增大图表的宽度，这样 RRDtool 会自动加多一些刻度上去。  
  
  
  
不过需要增加多大才会有上面的这种效果不得而知，所以这种方法不是很方便。  
  
B）二是通过上面的选项自己设置 X/Y 轴的刻度如何显示。首先看上图，在垂直的线中，红色的线称为 Major Grid（主要网格线），  
  
    灰色的线称为 Base Grid （次要网格线）（这里是借用 EXCEL 中的概念 ^\_^ ）。 X 轴下面的时间文字成为 Label 。了解这  
      
    三样东西后，就可以动手调整刻度了。  
      
C）有两种方法可以快速去掉 X/Y 轴的刻度，就是 –-x-grid none 和 –-y-grid none  
  
D）**GTM:GST ：**控制次要格网线的位置。GTM 是一个时间单位，可以是 SECOND、MINUTE、HOUR、DAY 、WEEK、MONTH、YEAR 。  
  
     GST 则是一个数字，控制每隔多长时间放置一根次要格线。例如我们要画一个1天的图表，决定每15分钟一根次要网格线，则格式为 MINUTE:15  
       
E）**MTM:MST ：**控制主要网格线的位置。MTM 同样是时间单位，MST 是一个数字。接上面的例子，决定一个小时1根主要网格线。则格式为 HOUR:1   
  
     LTM:LST ：控制每隔多长时间输出一个label 。决定为1小时1个 label 。则格式为 HOUR:1  
  
G）**LPR:LFM ：**LTM:LST 只是决定了 label 的显示位置了，没有指定要显示什么内容。LPR 指的是如何放置 label 。如果LPR 为0，则数字对齐格线  
  
    （适用于显示时间）。如果不为0，则会做一些偏移（适用于显示星期几、月份等）。至于LFM 则需要熟悉一下 date 命令的参数，常用的有 %a（星期几）、  
      
    %b（月份）、%d（天）、%H（小时）、%M（分）、%Y（年）。我们决定显示小时和分，所以用 %H%M  
  
H）综合起来，X 轴的刻度定义就是 –-x-grid MINUTE:15:HOUR:1:HOUR:1:0:’%H:%M’。最好把 %H:%M 括起来  
  
     建议 MST是 GST 的2-6倍，MST 和 LST 相同。这样画出来的图比较美观一些  
  
  
      
  
这明显就是图片太小了，不够显示。把上面的 :%M 去掉就可以了，只显示小时，不显示分钟  
  
      
  
如果把图片放大一点就更好了 (-w 800)  
  
      
  
所以在设置 X 轴的刻度时，要记得不要显示太多东西，否则需要增大图片的大小  
  
      
  
I）Y 轴刻度的设置又不一样了  
  
   **grid step ：**用于控制Y轴每隔多少显示一根水平线  
  
   **label factor ：**默认为1，也就是在每根水平线的高度那里显示一个值。  
  
   例如下面就是一个例子 （每隔800显示一根水平线）  
  
      
  
J）Y 轴还有一个很方便的选项就是 –Y ，它可以最大限度的优化 Y 轴的刻度，建议每次绘图都加上去。  
  
K）Y 轴另外一个有用的选项就是 –X （虽然选项名是 -X ，但确实是用来设置 Y 轴刻度值的）。在上面的图我们看到 RRDtool 自动对 Y 轴的值进行调整，  
  
     以 k 为单位显示。但如果你不想以 k 显示，而是想固定以某个单位来显示（M，b）该怎么办呢？这就要用到 –X 选项了。-X 后面跟一个参数，参数值  
       
     范围是 -18、-15、-12、-9、-6 、-3、0、3、6、9、12、15、18 。0 表示以原值显示，3 表示数值除以1000，也就是以 k 为单位显示，6 就是以  
       
     M 显示，9 就是以 G 显示。如果你给出1或者2，则 RRDtool 也可以接受，但会被“静悄悄”的改为0。下面就是一个以原值（-X 0）显示的例子  
  
