

## Linux 系统与程序监控工具 atop 教程

### 引言

Linux 以其稳定性，越来越多地被用作服务器的操作系统(当然，有人会较真地说一句：Linux 只是操作系统内核:)。但使用了 Linux 作为底层的操作系统，是否我们就能保证我们的服务做到 7\*24 地稳定呢？非也，要知道业务功能是由系统上跑的程序实现的，要实现业务功能的稳定性，选择 Linux 只是迈出的第一步，我们更多地工作是不让业务程序成为稳定性的短板。

当我们的服务器出现问题的时候，外在的表现是业务功能不能正常提供，内在的原因，从程序的角度看，可能是业务程序的问题(程序自身的 bug)，也可能是服务器上人为的误操作(不当地执行脚本或命令)；从系统资源的角度看，可能是 CPU 抢占、内存泄漏、磁盘 IO 读写异常、网络异常等。出现问题后，面对各种各样可能的原因，我们应如何着手进行分析？我们有什么工具进行问题定位吗？

### atop 简介

本文要介绍的 atop 就是一款用于监控 Linux 系统资源与进程的工具，它以一定的频率记录系统的运行状态，所采集的数据包含系统资源(CPU、内存、磁盘和网络)使用情况和进程运行情况，并能以日志文件的方式保存在磁盘中，服务器出现问题后，我们可获取相应的 atop 日志文件进行分析。atop 是一款开源软件，我们可以从官网获得其源码和 rpm 安装包。

### atop 使用方法

在安装 atop 之后，我们在命令行下敲入"atop"命令即可看到系统当前的运行情况：

```

lx@LX: /var/log
ATOP - LX                2011/12/17  21:35:38                -----                10m0s elapsed
PRC | sys      5m13s | user   5m02s | #proc  165 | #zombie  1 | #exit  269 |
CPU | sys      51% | user   50% | irq    1% | idle    98% | wait    1% |
cpu | sys      25% | user   24% | irq    1% | idle    49% | cpu000 w 1% |
cpu | sys      25% | user   25% | irq    0% | idle    49% | cpu001 w 0% |
CPL | avg1     0.14 | avg5   0.14 | avg15  0.17 | csw 2295130 | intr 654283 |
MEM | tot      1.9G | free  246.9M | cache 717.1M | buff 169.8M | slab  80.0M |
SWP | tot      976.0M | free  975.9M |          | vmcom  3.5G | vmlim  1.9G |
PAG | scan    38953 | stall  0 |          | swin    0 | swout   16 |
DSK |          sda | busy   3% | read   369 | write  2607 | avio 5.34 ms |
NET | transport | tcpi   1016 | tcpo  1238 | udpi   165 | udpo   165 |
NET | network   | ipi    1181 | ipo   1439 | ipfrw  0 | deliv  1181 |
NET | wlan0    ---- | pcki   1184 | pcko  1442 | si     6 Kbps | so     4 Kbps |

  PID  SYSCPU  USRCPU  VGROW  RGROW  RDDSK  WRDSK  ST  EXC  S  CPU  CMD  1/49
  4600  2m25s  1m41s  -64K  -1964K 19660K   24K  --  -  R  41% chrome
   1143  2m11s  88.20s  8712K  8644K   0K     0K  --  -  S  36% Xorg
   1729  9.31s  25.15s  1412K  164K   0K     16K  --  -  S   6% compiz
   5523  2.35s  21.72s   0K     0K   -     -  NE  0  E   4% <firefox>
   4485  6.68s  16.90s   32K   524K  20K  10436K  --  -  S   4% chrome
   4607  4.26s  19.20s 15572K  4716K   0K     0K  --  -  S   4% chrome
   1736  6.78s  7.73s   0K     28K   0K     0K  --  -  S   2% pulseaudio
   4804  0.79s  8.73s   0K    316K   0K     0K  --  -  S   2% chrome
   2647  0.33s  1.69s  144K   392K   0K    212K  --  -  S   0% gnome-terminal
    
```

### 系统资源监控字段含义

上图中列出了不少字段以及数值，各字段的含义是什么？我们应该怎么看？以上每个字段的含义都是相对采样周期而言的，下面我们先来关注上图显示的上半部分。

**ATOP 列：** 该列显示了主机名、信息采样日期和时间点

**PRC 列：** 该列显示进程整体运行情况

1. **sys**、**usr** 字段分别指示进程在内核态和用户态的运行时间
2. **#proc** 字段指示进程总数
3. **#zombie** 字段指示僵死进程的数量
4. **#exit** 字段指示 atop 采样周期期间退出的进程数量

**CPU 列:** 该列显示 CPU 整体(即多核 CPU 作为一个整体 CPU 资源)的使用情况, 我们知道 CPU 可被用于执行进程、处理中断, 也可处于空闲状态(空闲状态分两种, 一种是活动进程等待磁盘 IO 导致 CPU 空闲, 另一种是完全空闲)

1. **sys**、**usr** 字段指示 CPU 被用于处理进程时, 进程在内核态、用户态所占 CPU 的时间比例
2. **irq** 字段指示 CPU 被用于处理中断的时间比例
3. **idle** 字段指示 CPU 处在完全空闲状态的时间比例
4. **wait** 字段指示 CPU 处在“进程等待磁盘 IO 导致 CPU 空闲”状态的时间比例

CPU 列各个字段指示值相加结果为 N00%, 其中 N 为 cpu 核数。

**cpu 列:** 该列显示某一核 cpu 的使用情况, 各字段含义可参照 CPU 列, 各字段值相加结果为 100%

**CPL 列:** 该列显示 CPU 负载情况

1. **avg1**、**avg5** 和 **avg15** 字段: 过去 1 分钟、5 分钟和 15 分钟内运行队列中的平均进程数量
2. **csw** 字段指示上下文交换次数
3. **intr** 字段指示中断发生次数

**MEM 列:** 该列指示内存的使用情况

1. **tot** 字段指示物理内存总量
2. **free** 字段指示空闲内存的大小
3. **cache** 字段指示用于页缓存的内存大小
4. **buff** 字段指示用于文件缓存的内存大小

5. **slab** 字段指示系统内核占用的内存大小

**SWP 列**: 该列指示交换空间的使用情况

1. **tot** 字段指示交换区总量
2. **free** 字段指示空闲交换空间大小

**PAG 列**: 该列指示虚拟内存分页情况

**swin、swout 字段**: 换入和换出内存页数

**DSK 列**: 该列指示磁盘使用情况，每一个磁盘设备对应一列，如果有 **sdb** 设备，那么增多一列 DSK 信息

1. **sda** 字段: 磁盘设备标识
2. **busy** 字段: 磁盘忙时比例
3. **read、write** 字段: 读、写请求数量

**NET 列**: 多列 NET 展示了网络状况，包括传输层(TCP 和 UDP)、IP 层以及各活动的网口信息

1. **XXXi** 字段指示各层或活动网口收包数目
2. **XXXo** 字段指示各层或活动网口发包数目

## 进程视图

为了更全面地展示进程信息，**atop** 提供了多种进程视图。

### 默认视图(Generic information)

进入 atop 信息界面，我们看到的就是进程信息的默认视图(上图下半部分)，按 g 键可以从其他视图跳到默认视图。

ATOP - LX												2011/12/22 21:17:25		-----		10m0s elapsed	
PRC		sys	98.88s		user	4m59s		#proc	154		#zombie	1		#exit	9		
CPU		sys	16%		user	49%		irq	1%		idle	130%		wait	4%		
cpu		sys	8%		user	25%		irq	1%		idle	63%		cpu000 w	3%		
cpu		sys	8%		user	24%		irq	0%		idle	67%		cpu001 w	2%		
CPL		avg1	0.08		avg5	0.17		avg15	0.16		csw	1050153		intr	555567		
MEM		tot	1.9G		free	331.1M		cache	740.8M		buff	293.3M		slab	80.7M		
SWP		tot	976.0M		free	976.0M					vmcom	1.9G		vmlim	1.9G		
DSK			sda		busy	7%		read	20718		write	2869		avio	1.74 ms		
NET		transport			tcpi	2399		tcpo	2266		udpi	219		udpo	257		
NET		network			ipi	2618		ipo	2537		ipfrw	0		deliv	2618		
NET		wlan0	----		pcki	2622		pcko	2541		si	31 Kbps		so	5 Kbps		

  

PID	SYS	USR	VGROW	RGROW	RDDSK	WRDSK	ST	EXC	S	CPU	CMD	1/17
2313	49.98s	3m55s	-0.1G	-49.7M	1560K	9148K	--	-	S	48%	firefox	
1205	30.96s	29.37s	-8036K	-7644K	0K	0K	--	-	R	10%	Xorg	
2210	3.43s	14.13s	904K	1936K	12K	16240K	--	-	S	3%	gnome-terminal	
1726	4.06s	11.10s	-196K	1000K	0K	12K	--	-	S	3%	compiz	
1728	2.55s	2.82s	-64.4M	-772K	0K	0K	--	-	S	1%	pulseaudio	
3061	3.61s	0.96s	0K	0K	-	-	NE	0	E	1%	<find>	
1698	0.10s	1.54s	0K	116K	0K	0K	--	-	S	0%	python	
1686	0.18s	0.86s	0K	0K	0K	0K	--	-	S	0%	ibus-daemon	
1872	0.07s	0.89s	68K	136K	0K	0K	--	-	S	0%	unity-panel-se	
2344	0.26s	0.58s	-384K	-52K	0K	0K	--	-	S	0%	ld-linux.so.2	

从上图中，我们可以看到 PID 为 3061 的 find 进程在退出前在内核模式下占用了 3.43 秒 CPU 时间，在用户模式下占用了 0.96 秒 CPU 时间，共使用 CPU 时间为 4.39 秒，相对 10 分钟采样周期，CPU 时间占用比例为 1%，ST 列表示进程状态，N 表示该进程是前一个采样周期新生成的进程，E 表示该进程已退出，EXC 列指示进程的退出码。从进程名在“<>”符号中，我们亦可知该进程已退出。

### 内存视图(Memory consumption)

内存视图展示了进程使用内存情况，按 m 键可进入内存视图。

```

lx@LX: /var/log
ATOP - LX          2011/12/18  11:35:45          -----          10m0s elapsed
PRC | sys      6m04s | user   3m38s | #proc   170 | #zombie   1 | #exit   37 |
CPU | sys      58%  | user   36%  | irq     1% | idle     96% | wait    8% |
cpu | sys      31%  | user   18%  | irq     0% | idle     48% | cpu001 w 2% |
cpu | sys      27%  | user   18%  | irq     1% | idle     48% | cpu000 w 6% |
CPL | avg1     0.47 | avg5    0.40 | avg15   0.23 | csw 2307564 | intr 1722069 |
MEM | tot      1.9G | free   73.2M | cache 479.5M | buff  49.0M | slab  51.2M |
SWP | tot      976.0M | free  971.9M |          | vmcom  4.1G | vmlim  1.9G |
PAG | scan  220407 | stall   0 |          | swin    0 | swout  1042 |
DSK |          sda | busy   13% | read   9583 | write  4059 | avio 5.78 ms |
NET | transport | tcpi  17321 | tcpo  14762 | udpi   467 | udpo   596 |
NET | network   | ipi  17839 | ipo   15928 | ipfrw  0 | deliv 17839 |
NET | wlan0    ---- | pcki  17803 | pcko  15931 | si  284 Kbps | so   22 Kbps |

  PID MINFLT MAJFLT VSTEXT VSIZE  RSIZE  VGROW  RGROW  MEM  CMD  1/23
 8459 1174e3   24   22K  819.0M 570.7M 819.0M 570.7M 29% VirtualBox
 4485  23392    2 52698K 507.7M 121.5M -3092K -1980K 6% chrome
 4600 436661    0 52698K 329.9M 98348K -43.5M -15.8M 5% chrome
 1729  22423    4   428K  234.4M 83536K 16612K -4492K 4% compiz
 1143  19865    6   1717K  256.0M 78268K 11668K 9016K 4% Xorg
 4520    825    0 52698K 232.9M 74700K 128K -5004K 4% chrome
 5286  95970    0 52698K 223.6M 66552K 128K -672K 3% chrome
 8324  42039    0 52698K 216.9M 65076K 6024K -12.2M 3% chrome
 4607    10    0 52698K 219.3M 62004K 0K -3712K 3% chrome
  
```

上图下半部分展示了每个进程占用的虚拟内存空间(VSIZE)、内存空间(RSIZE)大小，以及在上一个采样周期中虚拟内存和物理内存增长大小(VGROW、RGROW)，MEM 列指示进程所占物理内存大小。

从上图的 PAG 列的信息，我们可以知道此时系统内存负载较高，出现页换出情况，从进程视图中 VGROW 和 RGROW 列可看出 VirtualBox 进程占用内存量大量增长，部分进程占用的内存减少(VGROW 或 RGROW 字段为负值)，为 VirtualBox 进程腾出空间。

### 命令视图(Command line)

按 c 键我们可以进入命令视图，该视图展示了与每个进程相对应的命令。

```
lx@LX: /var/log
ATOP - LX 2011/12/23 00:27:25 ----- 10m0s elapsed
PRC | sys 70.75s | user 3m51s | #proc 157 | #zombie 1 | #exit 85 |
CPU | sys 11% | user 37% | irq 1% | idle 150% | wait 1% |
cpu | sys 5% | user 20% | irq 0% | idle 75% | cpu001 w 0% |
cpu | sys 6% | user 18% | irq 0% | idle 75% | cpu000 w 1% |
CPL | avg1 0.04 | avg5 0.10 | avg15 0.12 | csw 1151050 | intr 666227 |
MEM | tot 1.9G | free 531.2M | cache 388.7M | buff 286.9M | slab 163.7M |
SWP | tot 976.0M | free 976.0M | | vmcom 2.1G | vmlim 1.9G |
PAG | scan 111603 | stall 0 | | swin 0 | swout 0 |
DSK | sda | busy 2% | read 1706 | write 1119 | avio 4.38 ms |
NET | transport | tcpi 3720 | tcpo 2888 | udpi 69 | udpo 82 |
NET | network | ipi 3790 | ipo 3026 | ipfrw 0 | deliv 3790 |
NET | wlan0 ---- | pcki 3787 | pcko 3030 | si 66 Kbps | so 6 Kbps |

PID CPU COMMAND-LINE 1/27
2313 29% /usr/lib/firefox-8.0/firefox
1205 10% /usr/bin/X :0 -nr -verbose -auth /var/run/gdm/auth-for-gdm-H7V3if/dat
1728 3% /usr/bin/pulseaudio --start --log-target=syslog
2210 3% gnome-terminal
1726 2% compiz
3515 0% a.out
3586 0% find
3533 0% du
3514 0% a.out
```

有时我们某位“马大哈”同事执行了某个脚本或命令，使得系统资源占用率异常飙高，这时，我们可以很容易地通过 `atop` 的命令视图找到导致异常的命令。

## atop 日志

每个时间点采样页面组合起来就形成了一个 `atop` 日志文件，我们可以使用"`atop -r XXX`"命令对日志文件进行查看。那以什么形式保存 `atop` 日志文件呢？

对于 `atop` 日志文件的保存方式，我们可以这样：

1. 每天保存一个 `atop` 日志文件，该日志文件记录当天信息
2. 日志文件以"`atop_YYYYMMDD`"的方式命名
3. 设定日志失效期限，自动删除一段时间前的日志文件

其实 atop 开发者已经提供了以上日志保存方式，相应的 atop.daily 脚本可以在源码目录下找到。在 atop.daily 脚本中，我们可以通过修改 INTERVAL 变量改变 atop 信息采样周期(默认为 10 分钟)；通过修改以下命令中的数值改变日志保存天数(默认为 28 天)：

```
(sleep 3; find $LOGPATH -name 'atop_*' -mtime +28 -exec rm {} \; )&
```

最后，我们修改 cron 文件，每天凌晨执行 atop.daily 脚本：

```
0 0 * * * root /etc/cron.daily/atop.daily
```

## 小结

本文对 Linux 系统资源及进程监控工具 atop 进行了介绍，对 atop 所记录信息中的部分字段的含义以及进程视图进行了分析，最后讲述了 atop 日志文件的保存方式。

atop 工具会根据终端界面的大小调整所显示的字段，因此你使用 atop 时看到的部分字段可能与以上截图不相同。若想更深入地了解 atop 工具中字段含义和各种进程视图，请猛击 [Reference](#) 链接。