Monitoramento de desempenho no Linux

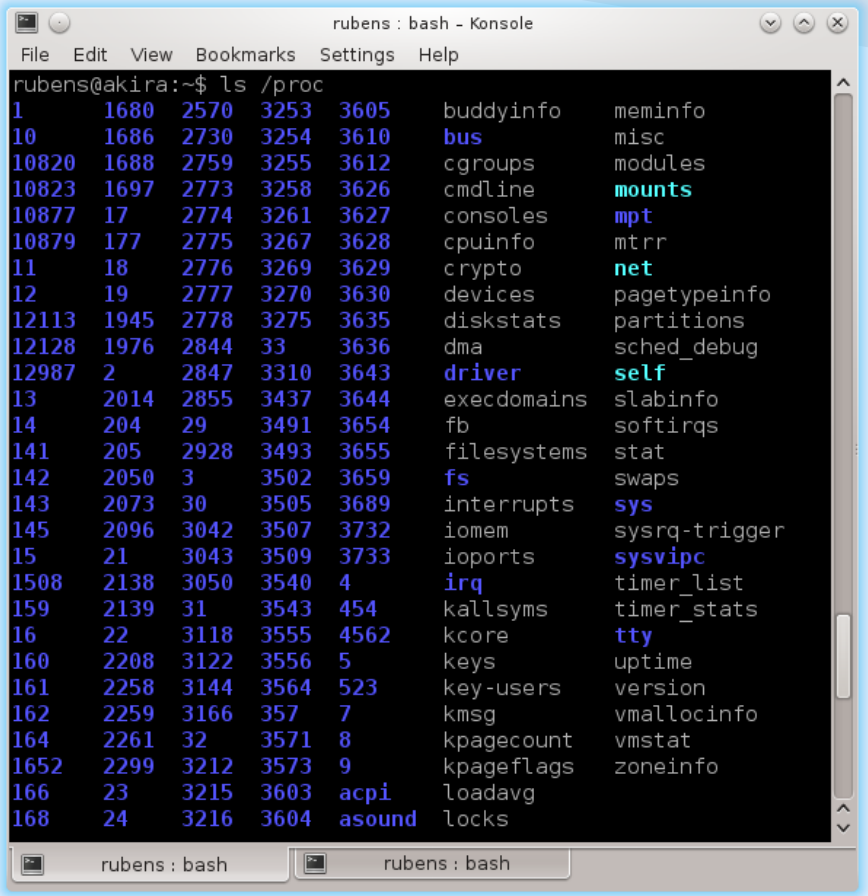
Rubens Matos <rsmj@cin.ufpe.br>

Agenda

- Proc filesystem
  - Monitorando memoria do sistema
  - Monitorando memoria de um processo
- Utilitários sysstat
  - iostat
  - pidstat

Utilizando o /proc

- /proc
  - Pseudo-sistema de arquivos, existente no GNU/Linux e em varios outros SOs baseados no Unix
  - Estruturado como uma hierarquia de diretorios e arquivos



Utilizando o /proc

- /proc
  - Interface para estruturas de dados internas do kernel (nucleo do sistema)
    - Acessar dados sobre processos e outros recursos do SO
    - Alterar parametros do kernel em tempo de execucao
  - Vários contadores de desempenho disponíveis
    - /proc/stat
    - /proc/meminfo
    - /proc/vmstat
    - /proc/diskstats
    - /proc/net/...
    - /proc/<pid>/...

Utilizando o /proc

Pode-se ver detalhes sobre cada opção do /proc na man page:

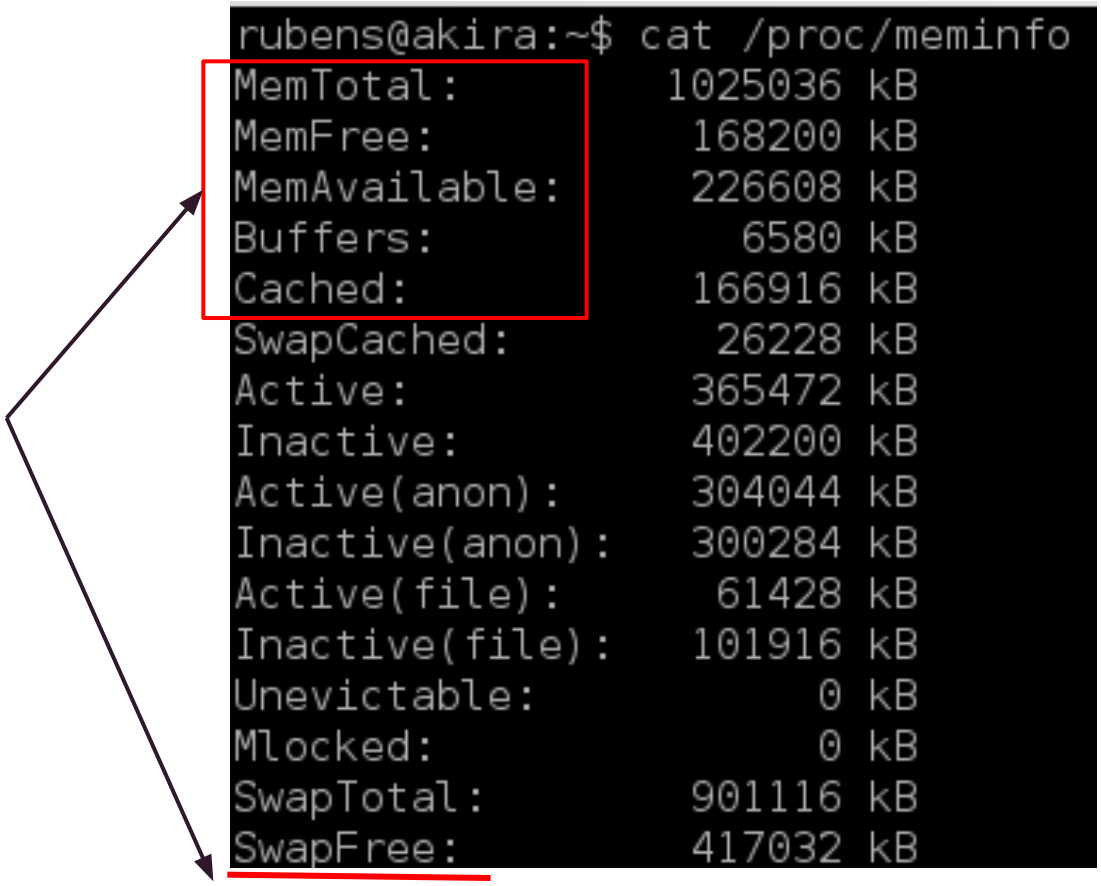
Digita-se no terminal: man proc

```
/proc/[pid]/io (since kernel 2.6.20)
       This file contains I/O statistics for the process, for example:

           # cat /proc/3828/io
           rchar: 323934931
           wchar: 323929600
           syscr: 632687
           syscw: 632675
           read_bytes: 0
           write_bytes: 323932160
           cancelled_write_bytes: 0

       The fields are as follows:

       rchar: characters read
              The  number of bytes which this task has caused to be read from storage.  This is simply
              passed to read(2) and similar system calls.  It includes things such as terminal I/O and
              actual physical disk I/O was required (the read might have been satisfied from pagecache

       wchar: characters written
              The  number  of  bytes which this task has caused, or shall cause to be written to disk.
              rchar.

       syscr: read syscalls
              Attempt to count the number of read I/O operations—that is, system calls such as read(2)

       syscw: write syscalls
              Attempt to count the number of write I/O operations—that is, system calls such as write(

       read_bytes: bytes read
              Attempt to count the number of bytes which this process really did cause to be fetched f
              accurate for block-backed filesystems.
```

/proc/meminfo

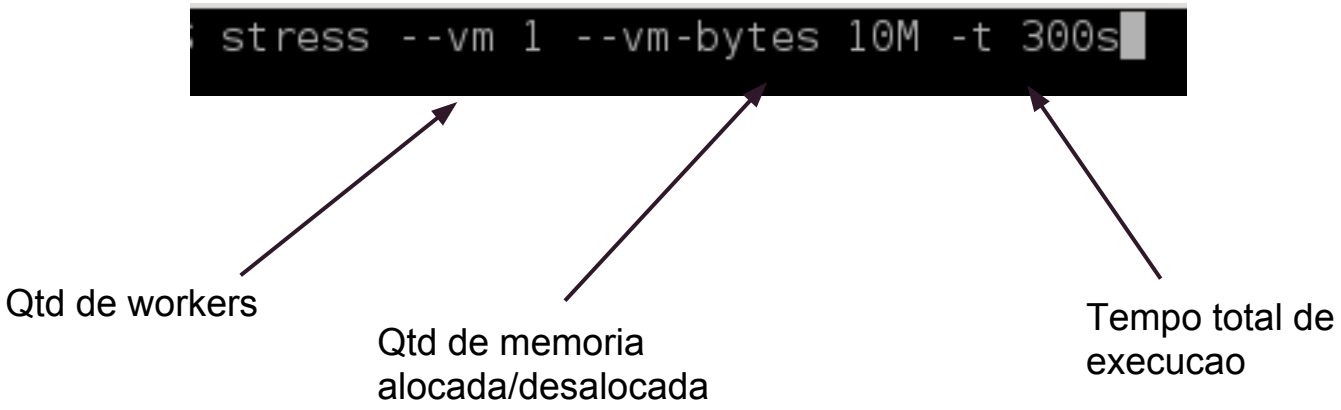
- /proc/meminfo

Informações bastante úteis para avaliar questões de desempenho



/proc/meminfo

- Vamos monitorar a memória do sistema enquanto executamos um teste de stress.



/proc/meminfo

- Nós podemos monitorar em tempo real
- Ou salvamos num arquivo (log) para visualizar depois

```
watch -n 5 cat /proc/meminfo
```

```
./monitora-memoria.sh
```

```
#!/bin/bash
tempo=0;
echo "MemFree Buffers Cached SwapFree">log-memoria.txt
while [ $tempo -lt 300 ]
do
   mfree=`cat /proc/meminfo | awk '/MemFree/{print $2}'`
   buff=`cat /proc/meminfo | awk '/Buffers/{print $2}'`
   cach=`cat /proc/meminfo | awk '/^Cached/{print $2}'`
   swapfree=`cat /proc/meminfo | awk '/SwapFree/{print $2}'`
   echo "$mfree $buff $cach $swapfree" >> log-memoria.txt
   sleep 5
   tempo=$((tempo + 5));
done
```

/proc/<pid>/status

- Em muitas situações, é essencial medir o uso de recursos para um processo em específico

```
rubens@akira:~$ cat /proc/16195/status
Name:   stress
State:  R (running)
Tgid:   16195
Ngid:   0
Pid:    16195
PPid:   16194
TracerPid:      0
Uid:    1000    1000    1000    1000
Gid:    1000    1000    1000    1000
FDSize: 64
Groups: 24 25 27 29 30 44 46 103 105 109 1000
VmPeak:   109572 kB
VmSize:   109572 kB
VmLck:         0 kB
VmPin:         0 kB
VmHWM:    102492 kB
VmRSS:     21940 kB
VmData:   102456 kB
VmStk:       136 kB
VmExe:        20 kB
VmLib:      2820 kB
VmPTE:        80 kB
VmSwap:        0 kB
Threads:        1
SigQ:   0/7884
SigPnd: 0000000000000000
ShdPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 0000000000000000
SigCgt: 0000000000000000
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
Seccomp:        0
Cpus_allowed:   1
```

VmSize: toda a memória virtual usada pelo processo.

VmHWM: teto atingido pelo RSS

Resident set size: memória física (RAM) usada pelo processo.

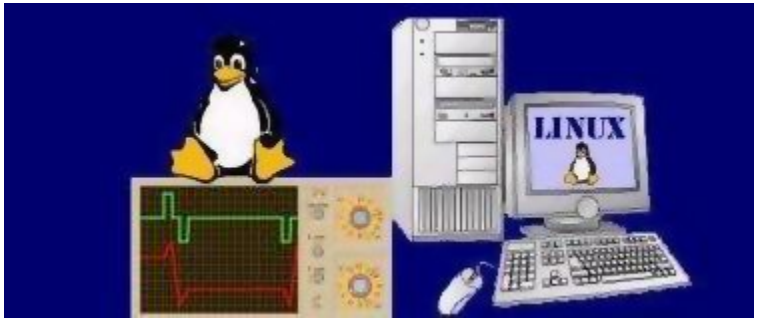
Como obter o PID:

```
rubens@akira:~$ pgrep -n stress
16195
```

Ferramentas auxiliares

- Algumas ferramentas facilitam o monitoramento:
  - Agrupam dados coletados do /proc
  - Convertem unidades
  - Gerenciam tempo/intervalo de monitoramento
  - Exemplos: **sysstat**, top, htop, netstat, etc
  - Algumas outras também geram relatórios, gráficos, alertas, etc.
  - Exemplos: Nagios, Ganglia, Cacti, ...

Utilizando o Sysstat



- O sysstat é um pacote de utilitários para coleta de dados de desempenho
  - iostat: Disco e I/O em geral
  - mpstat: Processador e memória
  - pidstat: Monitoramento por processo
  - sa2: Gera relatórios diários com os dados coletados

IOSTAT

- O iostat pode gerar dois tipos de relatorios
  - Utilização de CPU (para todo o sistema)
  - Utilização de dispositivo individual

```
rubens@akira:~$ iostat 5 4
Linux 3.14-1-amd64 (akira)      05/20/2014      _x86_64_        (1 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           8.44    0.71    5.41    2.16    0.00   83.29

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               9.71       183.18        90.76    3121339    1546480

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.02    0.00    0.00    0.61    0.00   98.37

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               5.08        30.08        14.63        148         72

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.20    0.00    0.00    0.00    0.00   99.80

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.40         0.00         2.41          0         12

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.00    0.00    0.00  100.00

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.20         0.00         0.81          0          4
```

IOSTAT

- Com iostat -d /dev/sda, eu monitoro apenas o dispositivo /dev/sda

```
rubens@akira:~$ iostat -d /dev/sda 5 4
Linux 3.14-1-amd64 (akira)      05/20/2014      _x86_64_        (1 CPU)

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               9.91       185.35        88.85    3237423    1551852

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.40         0.00         2.42          0         12

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.00         0.00         0.00          0          0

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               0.00         0.00         0.00          0          0
```

IOSTAT

- Vamos testar o monitoramento enquanto o **stress** gera carga pro disco

```
stress --hdd 1 --hdd-bytes 500M -t 300s
```

```
rubens@akira:~$ iostat -d /dev/sda 5 60
Linux 3.14-1-amd64 (akira)      05/20/2014      _x86_64_        (1 CPU)

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda              10.77       201.81       125.57    3624391    2255084

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda             220.71       155.29    105355.29        660     447760

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda             181.86       819.61     67734.31       3344     276356

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda              47.26        57.38     20835.44        272      98760

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda              42.33         5.18     20355.94         24      94248

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda             128.22      2111.96     17944.02       9356      79492

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda             224.65      3608.45     29738.97      15372     126688
```

PIDSTAT

- Com o pidstat podemos monitorar as informações que encontram-se no /proc/<pid>/..., porém com muitas facilidades:
  - Estatísticas do processo e de seus filhos conjuntamente
  - Estatísticas das threads associadas ao processo

PIDSTAT

```
stress --vm 1 --vm-bytes 10M -t 300s
```

```
rubens@akira:~$ pidstat -T ALL -C "stress" -r 2 10
Linux 3.14-1-amd64 (akira)      05/20/2014      _x86_64_        (1 CPU)

03:03:00 AM   UID       PID  minflt/s  majflt/s    VSZ    RSS   %MEM  Command
03:03:02 AM  1000     17599 682000.00      0.00  109572  64708   6.31  stress

03:03:00 AM   UID       PID minflt-nr majflt-nr  Command
03:03:02 AM  1000     17599   1377640         0  stress

03:03:02 AM   UID       PID  minflt/s  majflt/s    VSZ    RSS   %MEM  Command
03:03:04 AM  1000     17599 729235.68      0.00  109572  32764   3.20  stress

03:03:02 AM   UID       PID minflt-nr majflt-nr  Command
03:03:04 AM  1000     17599   1451179         0  stress

03:03:04 AM   UID       PID  minflt/s  majflt/s    VSZ    RSS   %MEM  Command
03:03:06 AM  1000     17599 725540.50      0.00  109572  61996   6.05  stress

03:03:04 AM   UID       PID minflt-nr majflt-nr  Command
03:03:06 AM  1000     17599   1451081         0  stress

03:03:06 AM   UID       PID  minflt/s  majflt/s    VSZ    RSS   %MEM  Command
03:03:08 AM  1000     17599 725940.80      0.00  109572 102196   9.97  stress
```

Referências

- Man-pages do Linux
- Site do iostat:

  http://sebastien.godard.pagesperso-orange.fr