Monitorando processos

- Definição de processos
- Os processos em Linux
- Arquivos e estruturas de processos em Linux
- Comandos e monitoramento
- Sinais

Definições

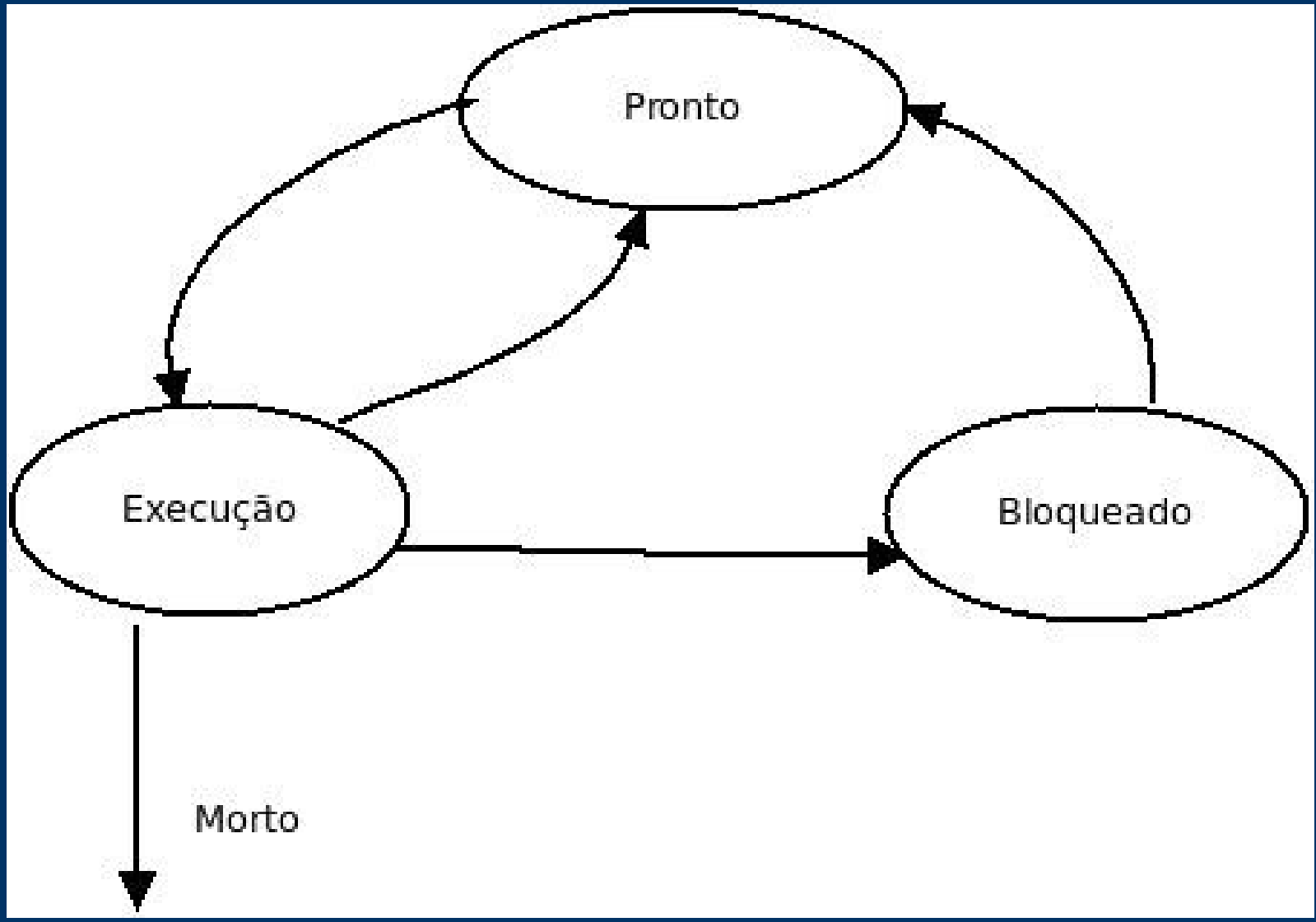
Processos: Todo programa em execução (em memória).

Threads: Conjunto de instruções de uma mesma rotina ou parte de um processo. São considerados subprocessos e compartilham área de memória.

Estados dos processos: rodando, espera e bloqueado.

Sistema operacional deve se preocupar com eventos diretamente ligados: escalonamento, recursos e memória

Estados de processos



Processos em Linux

Informações de um processo:
- Estado
- Mapa de endereços
- Prioridade
- Informações sobre os recursos de um processo
- Sinais para um processo
- Proprietário de um processo
- PID e PPID

Informações = contexto do processo

Processos em Linux

/proc – análogo ao BCP

/proc/[#pid] : subdiretório que armazena as informações do processo de PID indicado.

Contextos dos processos se encontram armazenados em forma de arquivos (pseudo-file system)

Diversos arquivos em formato texto que carregam os parâmetros que determinam funcionamento do sistema. (ver man (5) proc)

Informações de controle do /proc

- /proc/PID/cmdline : argumentos do processo
- /proc/PID/cwd : link para o diretório do processo
- /proc/PID/exe : link para o executável
- /proc/PID/environ :defs do ambiente usado
- /proc/PID/root : raiz vista pelo processo
- /proc/PID/status : informações (memória e estado)
- /proc/PID/task : atalhos para filhos

Processos em Linux

Escalonamento dos processos é administrado através de vetores de processos ativos e expirados.
Na versão 3 do kernel são 100 entradas de filas em um vetor e as 40 menos prioritárias (fair_sched) ficam em listas rubro-negras

Políticas de escalonamento podem ser de real time ou non real time.

Os processos podem utilizar recursos compartilhados que são controlados por semáforos. A comunicação pode ser realizada através de mensagens, sinais ou área de memória compartilhada

Na versão 3.0, o kernel pode ser tickless. Isto é, não tem uma frequência pré-definida para ser executado (antes era 10ms e 4ms)

Monitorando

- ps
- top
- vmstat
- pstree
- pidof

Alterando o foco

- &

- fg

- bg

Algoritmos de escalonamento

REAL TIME (priority 0 -99)

- SCHED_RR

- SCHED_FIFO

NON REAL TIME (nice 19 - -20)

-SCHED_OTHER

-SCHED_BATCH

Alterando prioridade

- nice

- renice

- chrt

Chamadas de sistema

- execv( )

-fork( )

Chamadas de sistema

```
int main()
{
pid_t pid_filho;
int retval;
pid_filho=fork();

if (pid_filho >=0)
{
if (pid_filho==0)
{
          printf("Filho criado com sucesso!\n");
          printf("Sou filho meu ID é: %d\n", getpid());
                  printf("Meu pai tem PID: %d\n", getppid());
         //sleep(20);
       for (int i=0;i<1000000;i++)
         { }
printf("Forneca um numero para matar o filho:\n");
              scanf(" %d", &retval);
}
else
{
 printf("Sou pai, meu PID é: %d\n",getpid());
 //sleep(19);
        for (int i=0;i<900000;i++)
 { }
        printf("Forneca um numero para matar o pai:\n");
               scanf(" %d", &retval);
}
}
else
{ printf("Erro! Nao foi possivel gerar filho!\n"); }
exit(0);
}
```

Sinais

- Mensagens ou operações enviadas aos processos

- Comunicação entre processos, entre usuário e processo e sistema com o processo

- Definição de sinais padrão. Atribuição de números a cada um dos sinais existentes.

Sinais

Tabela de sinais :

SIGHUP	1	Term	Hangup detected on controlling terminal or death of controlling process
SIGINT	2	Term	Interrupt from keyboard
SIGQUIT	3	Core	Quit from keyboard
SIGILL	4	Core	Illegal Instruction
SIGABRT	6	Core	Abort signal from abort(3)
SIGFPE	8	Core	Floating point exception
SIGKILL	9	Term	Kill signal
SIGSEGV	11	Core	Invalid memory reference
SIGPIPE	13	Term	Broken pipe: write to pipe with no readers
SIGALRM	14	Term	Timer signal from alarm(2)
SIGTERM	15	Term	Termination signal
SIGUSR1	30,10,16	Term	User-defined signal 1
SIGUSR2	31,12,17	Term	User-defined signal 2
SIGCHLD	20,17,18	Ign	Child stopped or terminated
SIGCONT	19,18,25	Cont	Continue if stopped
SIGSTOP	17,19,23	Stop	Stop process
SIGTSTP	18,20,24	Stop	Stop typed at tty
SIGTTIN	21,21,26	Stop	tty input for background process
SIGTTOU	22,22,27	Stop	tty output for background process

Sinais

Kill – envia um sinal a um processo através de seu PID

#> kill -9 2142

killall - envia o mesmo sinal a todos processos conforme o cmdline ou nome de um processo

#> killall -9 soffice

Praticando

Execute os comandos

#> tail -f /etc/passwd &
#> tail -f /etc/hosts &

Traga ao plano principal o processo que visualiza o passwd. Em seguida digite CTRL+Z

Qual é o estado do processo agora? Como fazê-lo executar e depois trazê-lo para o plano principal?

Conclusão

- Conhecer o escalonador
- Conhecer as ferramentas que identificam e monitoram os processos. Entender e identificar os estados dos processos
- Conhecer chamadas de sistemas (desenvolvedor)
- Conhecer a troca de sinais