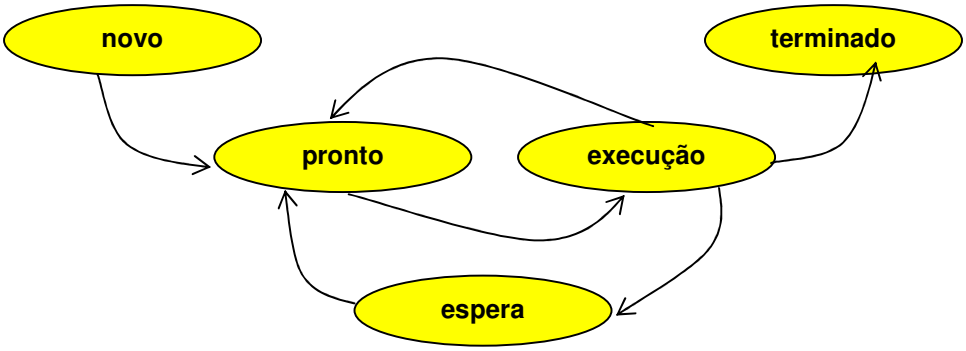
Laboratório de Sistemas Operacionais - Aula Prática 07 – 11.05.2016
Sistema Operacional Linux

Trabalhando com Processos

1. Esta sessão vai mostrar para você como trabalhar com processos no ambiente Linux. Inicialmente, vamos criar um novo diretório e chamá-lo de ***Lab07*** na sua pasta atual. Use o seguinte comando: ***mkdir Lab07***. Para ver se ele foi criado, use o comando ***ls –l*** . Ok?

2. Agora, vamos seguir para a nova pasta criada. Digite ***cd Lab07***. Para ter certeza que está no diretório correto, use o comando ***pwd*** e veja se o último arquivo é o ***Lab07***.

3. Vamos relembrar os ***status*** dos processos, que foram estudados na parte teórica de Sistemas Operacionais:



4. Agora vamos ver tudo isso na parte prática. Inicialmente, digite o seguinte comando: ***ps u*** . O comando os significa process status, ou estados dos processos, e ele mostra em que estado (conforme o fluxo do item 3 acima) um determinado processos está.

Entendendo os status de ***ps u***:

R – ***Running*** – O processo está realizando alguma atividade (execução).
S – ***Sleeping*** – O processo está "dormindo" ou aguardando algo para realizar alguma atividade (Pronto)
D – ***Espera*** – O processo está "dormindo" indefinidamente (espera p/ I/O)
T – ***Traced*** – O processo está parado.
X – ***Morto*** – O processo está morto, cancelado.
W – ***Pagging*** – O processo está na memória paginada (memória virtual).
Z – ***Zombie*** – Chamado processo Zumbi. Está perdido.

5. Para maiores informações sobre o comando, procure o manual on-line:
   Digite ***man ps*** e veja todas as informações sobre um processo.

6. Entendendo melhor o comando **ps**. Experimente as várias versões do comando ***ps***:

   - digite ***ps*** e observe as informações
   - digite ***ps a*** e observe as informações
   - digite ***ps u*** e observe as informações
   - digite ***ps x*** e observe as informações
   - digite ***ps aux*** e observe as informações

7. Execute o comando: ***ps aux***

   Obs.: O comando acima ***ps aux*** listou todos os processos que estão sendo executados no servidor Linux.
   Note que a tela girou muito rapidamente e você não pode ver todos os processo, então, digite ***ps aux | more***. Desta forma, você pode ver os processos tela a tela. Para ver a próxima tela é só dar um toque na tecla da barra de espaços.

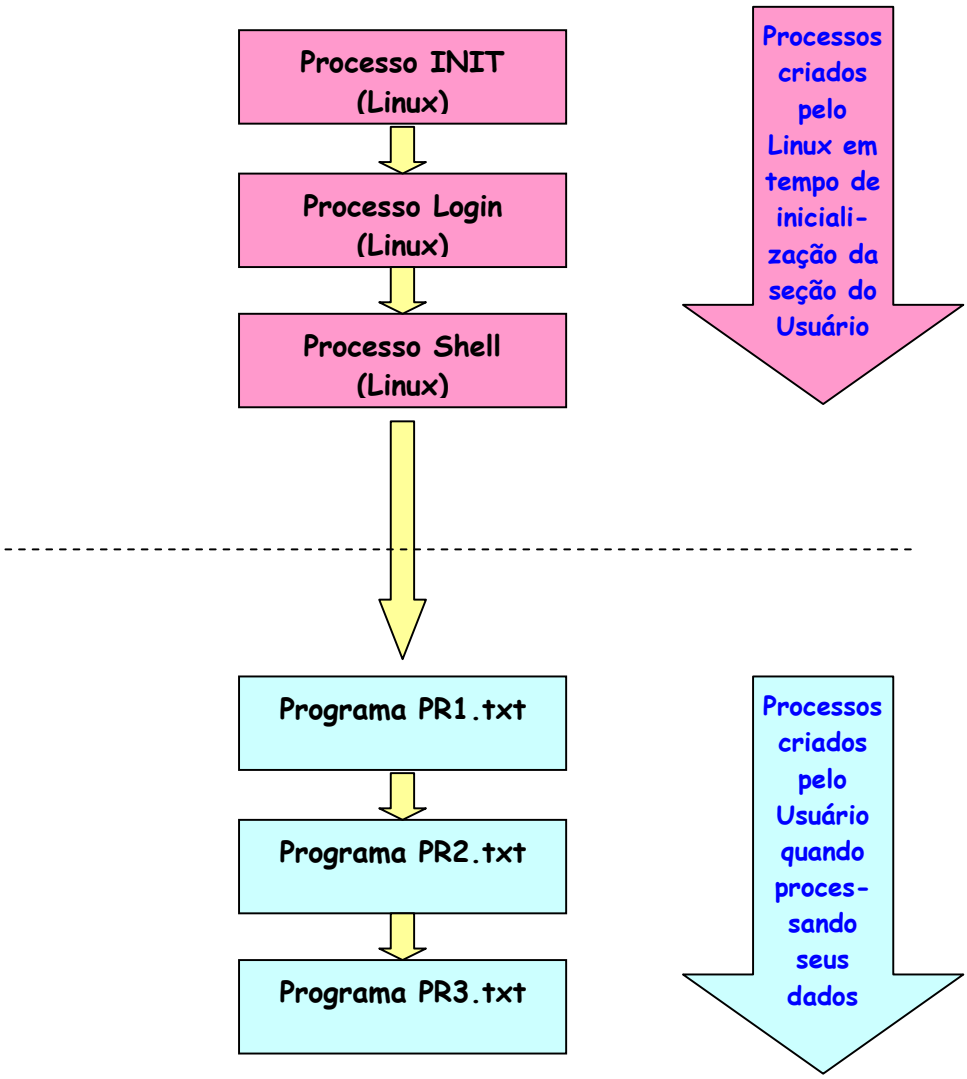
8. Na verdade, gostaríamos de saber quantos processos estão rodando no servidor em determinado momento, e contá-los um a um é complicado. Para obter esta informação, execute o comando: ***ps aux | wc –l***

   Obs.: O comando acima ***ps aux*** listou todos os processos que estão sendo rodados no seu usuário, e quando encontrou um pipe, passou todas essas linhas para o próximo comando. O próximo comando após o pipe é um ***wc*** (contador de caracteres) que conta quantas linhas tem e solta a informação na tela.

9. Vamos executar processos um dependente do outro. Teremos um programa ***PR1.TXT*** chamando um programa ***PR2.TXT***, e este chamando um programa ***PR3.TXT*** onde os três executam o comando ***ps u***.

   Obs.: O que pretendemos aqui é mostrar para vocês algumas características, como: (a) que um processo ***PR2.TXT*** é filho e dependente do processo ***PR1.TXT***, e que o processo ***PR3.TXT*** é neto de ***PR1.TXT***, filho do processo ***PR2.TXT*** e dependente desse; (b) que, cada vez que um processo chamado entra no ar, nós daremos um comando ***ps*** e pararemos a execução, para você ver como ficaram os estados dos processos; (c) que podemos criar variáveis globais e fazê-las circular por todos os processos e subprocessos que disparamos.

O objetivo desta seqüência é mostrar a você o grau de dependência que será criado entre os processos.



Obs.: Nos programas abaixo, o comando **read** realiza uma parada no sistema que está em execução, colocando-o em estado de espera, porque você espera que o usuário digite algo para continuar o processo.

10. Utilizando o ***vi***, digite o programa abaixo ou copie daqui para dentro do ***vi***.
    Faça ***vi PR1.txt.***

```
#!/bin/bash
#
# O programa primeiro dá uma mensagem
# Em seguida solta um comando mostrando o status deste programa
# O comando read é usado apenas para dar uma pausa no programa
# Em seguida o programa pr1.txt chama o programa pr2.txt
# Quando voltar de pr2.txt o programa dá uma mensagem
# Emite novamente o comando ps u
# Novamente usamos o read para dar uma pausa no programa
#
printf "Processo 1 começando. \n"
ps u
read x
./PR2.txt
printf "Retornando para Processo 1. \n"
ps u
read x
```

11. Faça uma cópia do programa ***PR1.txt*** e chame de ***PR2.txt***.
    Use o comando: ***cp PR1.txt PR2.txt.***
    Agora, utilizando o vi, abra o programa PR2.txt e deixe-o conforme o programa abaixo. Faça ***vi PR2.txt*** :

```
#!/bin/bash
printf "Processo 2 começando. \n"
ps u
read x
./PR3.txt
printf "Retornando para Processo 2. \n"
ps u
read x
```

12. Faça uma cópia do programa PR2.txt e chame de PR3.txt.
    Use o comando: ***cp PR2.txt PR3.txt.***
    Agora, utilizando o vi, abra o programa PR3.txt e deixe-o conforme o programa abaixo. Faça ***vi PR3.txt*** :
    #!/bin/bash
    printf "Processo 3 começando. \n"
    ps u
    read x

13. Para executar todos os programas será necessário dar permissão de execução a todos eles: Execute os seguintes comandos:

    chmod u+x PR1.txt
    chmod u+x PR2.txt
    chmod u+x PR3.txt

14. Certifique-se de que todos os programas estão com permissão de execução, fazendo. ***ls –l PR*.txt***

15. Agora execute o programa ***PR1.txt***, fazendo o seguinte:
    ./PR1.txt

    Obs.: Execute o programa e vá verificando como se comportam os status dos processos ***PR1.txt***, ***PR2.txt*** e ***PR3.txt***.

    Note que, cada vez que você executa os processos, eles ganham PID (Identificação de processos) diferente.

16. E se eu quiser ver todos os processos que estão rodando no servidor Linux, como faço? Digite o comando ***top***. O comando top mostra todos os

processos que estão rodando no servidor Linux e todas as informações relacionadas aos processos.

Obs.: Faça uma completa análise das informações que você está vendo na tela do top. Entenda uma a uma.

17. Para sair do comando ***top***, dê um toque na tecla ***q***.

18. O comando ***top*** dá um ***refresh*** a cada 3 segundos, e este é o seu padrão. Você pode mudar este padrão.
    Digite novamente ***top***. Agora dê um toque na tecla ***s***. Aparecerá um query perguntando para quantos segundos você quer que o ***top*** dê o refresh. Digite 7 ou qualquer outro número e assista a sua execução.

19. Agora vamos entender como funciona o comando ***kill***, porque em algumas situações, você deseja eliminar algum processo que esteja rodando ou que esteja parado desnecessariamente. Para isso, vamos colocar dois programas em estado de parado (***stopped***) e depois vamos eliminá-lo.

    Digite ***vi PR1.txt*** e quando o programa abrir dê um ***ctrl + Z***.
    Digite ***vi PR2.txt*** e quando o programa abrir dê um ***ctrl + Z***.

    Para ter certeza que os seus programas ***vi*** estão parados, digite ***ps u*** e veja se eles estão lá.

20. Agora vamos eliminar os programas que você interrompeu e que estão parados. Digite ***kill -9 nnnn*** onde nnnn é o número do PID do processo. Você acha este número quando deu o comando ***ps u***. O comando ***kill*** aceita vários PIDs, ou seja, você pode matar vários processos ao mesmo tempo.
    O ***kill*** pode ficar assim: ***kill -9 nnnn1 nnnn2 nnnn3 .... nnnnm***

Obs. Tome muito cuidado para não matar o processo errado, por exemplo, matar o seu ***Shell***.

Você só consegue matar processos que foram abertos e que estão sendo administrados por você. Somente o ***root*** (ou administrador) é quem pode matar processos de qualquer usuário.

21. Agora vamos entender o que significa executar um programa em ***Foreground*** e ***Background***.

Foreground – Toda vez que você digita algo no ***prompt*** do seu Shell, você está processando instruções e programas na prioridade mais alta. O problema é que, se o programa ou instrução que você quer processar for muito demorado, você perde o ***prompt*** e não consegue fazer mais nada.

Background – Qualquer comando ou programa pode ser executado nas partições de mais baixa prioridade. Podemos colocar quantos programas ou comandos quisermos. A grande vantagem é que o prompt do seu Shell fica liberado para você continuar trabalhando e fazendo qualquer outra atividade.

Veja no quadro abaixo, que a diferença entre ambos é que em ***background*** há um ***&*** no final do comando.

Foreground	Background
$ ./pg1.txt	**$ ./pg1.txt &**

22. Vamos praticar a execução de um comando pesado em ***Foreground***:

Digite o seguinte comando: ***ls –lR /***

Obs.: Este comando lista todos os arquivos e diretórios de forma recursiva a partir da raiz do servidor. Se você não tiver acesso a determinado arquivo, o linux lista isso na tela. Não se preocupe com isso, é normal. ***É um comando pesado***. Note que você perdeu o controle do seu ***prompt***.

Como está saindo muita informação na tela, vamos colocar o resultado do comando dentro de um arquivo, portanto, digite o comando: ***ls –lR / > aa***

Obs.: Idem ao anterior, porém o resultado vai para arquivo ***aa***, no entanto você continua perdendo o controle do prompt.

23. Vamos praticar a execução do mesmo comando, agora em ***Background***:

Digite o seguinte comando: ***ls –lR / > aa &***

Obs.: Note que a diferença entre este comando e aquele último que executamos em foreground é o fato de que colocamos um caractere **&** após a última informação do comando.
O comando agora está sendo executado em segundo plano, e você pode ter o controle do seu ***prompt***.

24. Ok, agora você já sabe colocar programas ou comando pesados em background, mas como é que você monitora o que está sendo feito?
Isto é feito pelo comando ***jobs***.

Para entender claramente como ele funciona, repita o comando do item 23 e em seguida digite ***jobs***. Note a informação que você vai receber. Você vai receber a informação em forma de PID.

25. Bem, você deve ter criado um arquivo ***aa*** muito grande, e então vamos eliminá-lo. Digite o comando ***ls –l aa*** e veja o tamanho deste arquivo. Em seguida, digite ***rm aa*** e o arquivo será eliminado.

26. Vamos refazer estes testes, mas deixando bem carregado o servidor. Então, vamos executar 2 processos em Background:

ls –lR / > aa & - Execução em segundo plano guardando em aa

ls –lR / > bb & - Execução em segundo plano guardando em bb

Monitore a execução dos processos digitando ***jobs***. Digite quantas vezes quiser a palavra ***jobs*** para ver como anda a execução.

Em seguida, digite: ***ps u***

depois, mate processos: ***kill***

no final, elemine os arquivos ***aa*** e ***bb***. Digite ***rm aa bb***

Obs. Se as execuções forem muito rápidas, optar pelo uso do backup (**tar**)

27. Vamos entender a interrupção de processos (Ctrl + Z):

Execute o comando seguinte: ***grep –iR arq*** em Foreground, e em seguida dar ***Ctrl +Z***. O processo será interrompido e a interrupção ganhará um número. Digamos que ele recebeu a seguinte informação [1]+

Para ver o seu processo parado, dê o comando ***ps u***

Libere o processo para ser executado, agora, em Background.
A sintaxe é ***bg [%n]***. Digite ***bg n*** sendo n o número que o Linux deu quando o processo foi interrompido. Conforme o exemplo acima, ficaria assim: ***bg 1***

Note que o processo que estava em Foreground agora está em Background, e para ver isso, digite ***jobs*** que ele mostrará o status do grep.

28. Entenda a interrupção de processos dois processos (Ctrl + Z):

Liberar ***grep –iR arq*** em Foreground, e em seguida dar Ctrl +Z. O processo será interrompido e a interrupção ganhará um número. [1]

Liberar ***ls –lR > aa &*** em Background, e em seguida dar Ctrl +Z. O processo será interrompido e a interrupção ganhará um número. [2]

Liberar o processo interrompido em Foreground para ser executado, agora, em Background.
Digite ***bg 1***

Liberar o processo interrompido em Background para ser executado, agora, em Foreground.
Digite ***fg 2***

29. Entendendo a alteração de prioridade de processos (**nice**):

nice –n 20 find / -name teste &
renice -1 26280 altera processo já iniciado.

30. **Feche** o Linux

"Um homem pode fracassar muitas vezes, mas só é um fracassado quando começa a culpar outra pessoa."

John Burroughs