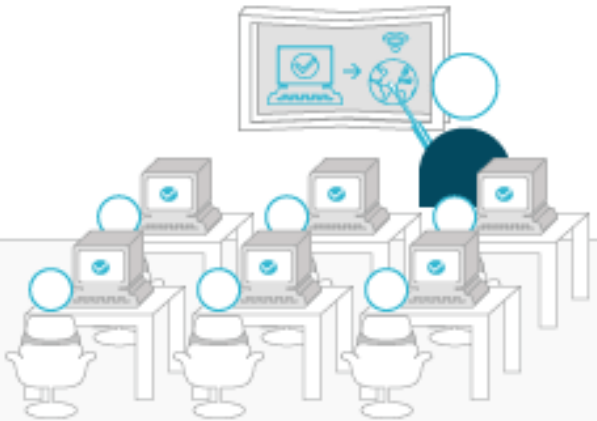

# Título do Curso:

## Administração de Sistemas GNU/Linux

# **Conteúdo** do curso

## ADMINISTRAÇÃO DE SISTEMAS GNU/LINUX

- Tópico 1: Introdução ao sistema operacional GNU/Linux.
- Tópico 2: Introdução ao Shell e comandos básicos.
- Tópico 3: Manipulação de conteúdos com comandos no Shell.
- Tópico 4: Comandos para gerenciamento do sistema e do Hardware.
- Tópico 5: Editor de Texto VI.
- Tópico 6: Administração de usuários e grupos.
- Tópico 7: Gerenciamento de permissões.
- **Tópico 8: Gerenciamento de processos.**
- Tópico 9: Sistemas de arquivos e particionamento.
- Tópico 10: Expressões regulares.
- Tópico 11: Introdução ao Shell Script.
- Tópico 12: Gerenciamento de Pacotes.
- Tópico 13: Agendamento de tarefas (cron) e Backup.



DGP

Tecnologia da Informação

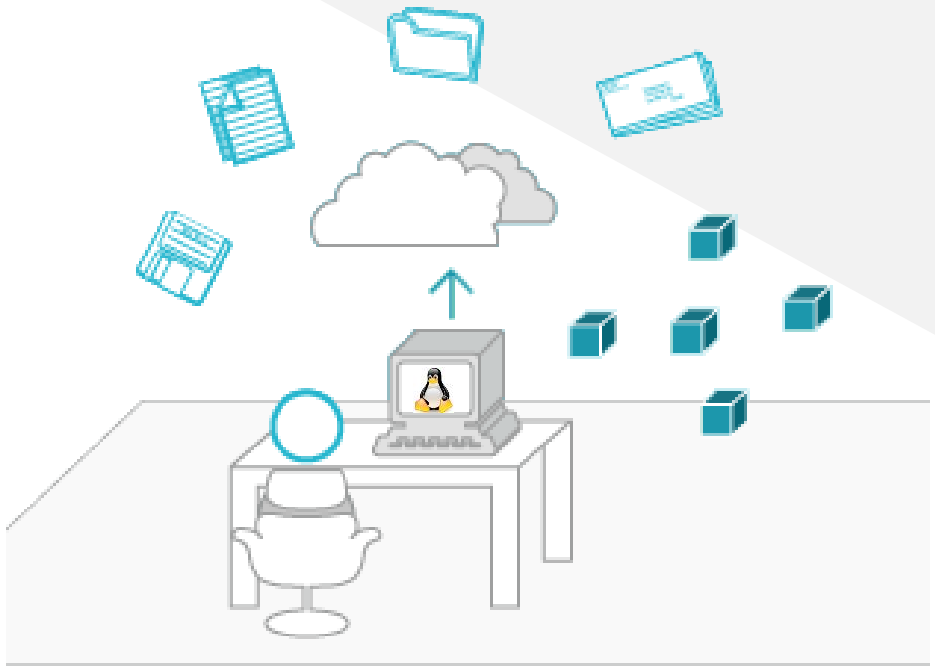
# **Tópico** 8

Gerenciamento de processos.

# **Gerenciamento** de

## Processos

- Neste slide serão abordados conceitos e comandos que nos auxiliam no gerenciamento de processos:
  - Conceitos sobre processos;
  - Comandos para gerenciamento de processos.



# Conceitos sobre processos

Tópico 8: Gerenciamento de processos.

# Conceitos
# Processos (Sistemas Operacionais)

- Em sistemas operacionais, todo software em execução gera pelo menos um processo;
- De acordo com a aplicação, um processo pode gerar outros processos, criando o que chamamos de processo “**pai**” e processos “**filhos**”;
- Cada processo possui diversas características;
- O estado de execução de um processo pode variar entre diversos “**estados de execução**”, porém, cada processo pode possuir apenas dois “**planos de execução**”;

# Conceitos
## Planos de Execução de um Processo

- Basicamente, um processo possui apenas dois planos de execução:
  - ***Foreground*** (Primeiro Plano): A execução do processo pode ser acompanhada na tela do terminal, onde devemos aguardar o término de sua execução para executar um novo comando;
  - ***Background*** (Segundo Plano): A execução do processo é realizada sem impedir que novos comandos sejam executados no terminal;
    - OBS.: Para executar um processo em segundo plano, basta adicionar o caractere “&” no final da linha de comando (para grande parte dos comandos, porém, nem todos).

# Conceitos
# Características de um Processo

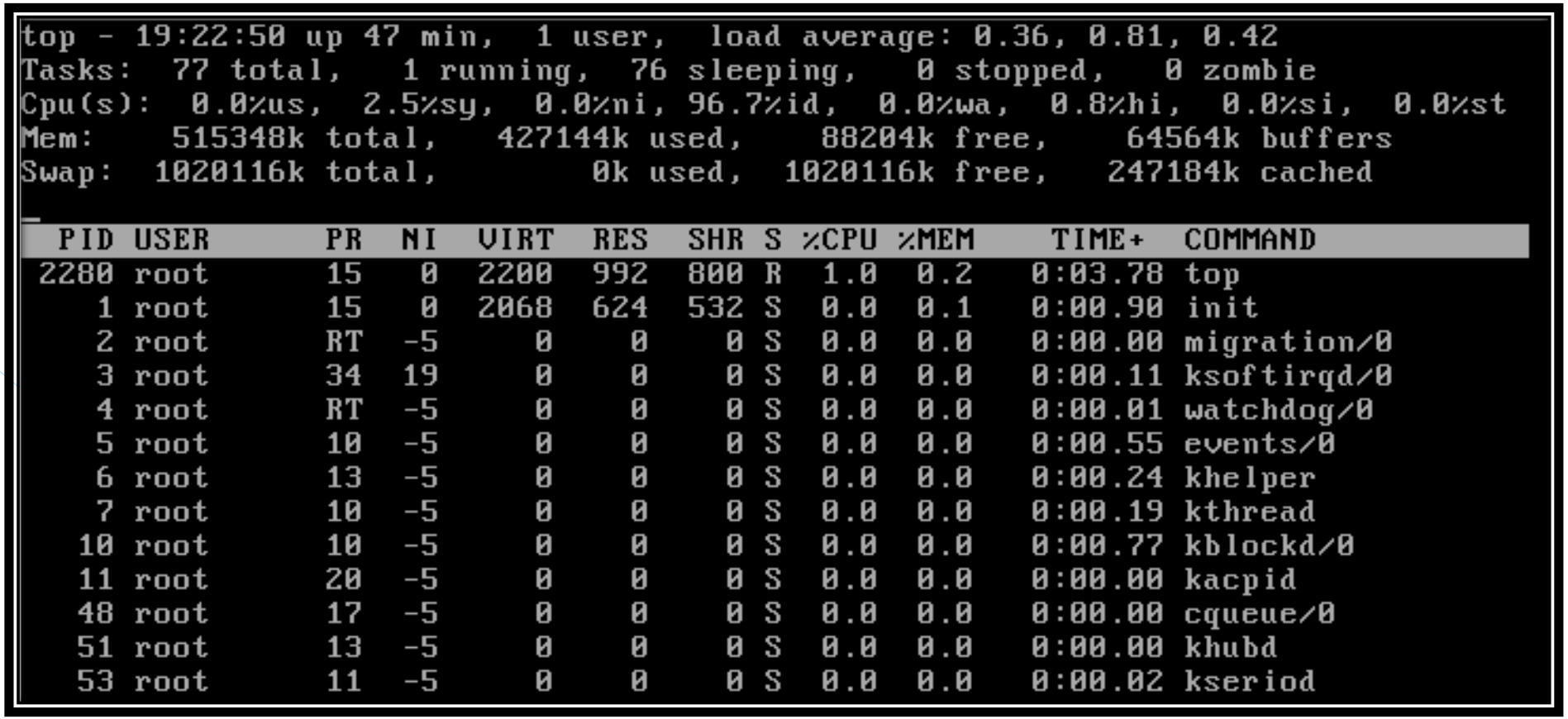
- Um processo possui diversas características, dentre elas:
  - **Tempo de Vida:** Tempo em que o processo utiliza os recursos do processador para ser executado. Varia significativamente de acordo com o processo;
    - OBS.: Podemos comparar o tempo de execução de um “ls” ou do próprio “top” com um serviço de rede (*daemon*), como um Servidor DNS, WEB, entre outros;
  - **PID (*Process Identifier*):** Número único e exclusivo que identifica um processo;
  - **UID (*User Identifier*) ou GID (*Group Identifier*):** Número que identifica o usuário ou o grupo “responsável” pela execução do processo;
  - ***Parent Process*:** Processo “pai”, ou seja, o processo responsável pelo processo analisado (caso seja um sub processo) ou o primeiro processo do sistema, que sempre será o “systemd” ou o “init” (com PID 1);
  - **PPID (*Parent Process Identifier*):** ID do processo “pai”;
- OBS.: Temos outras informações como “variáveis” do processo e diretório padrão, porém, não são relevantes no momento.

# Comandos para gerenciamento de processos

Tópico 8: Gerenciamento de processos.

# Processos – Comando “top”

- top → Exibe de forma dinâmica informações sobre processos:
  - Através do “top” podemos verificar o desempenho do Processador e quais processos estão consumindo os recursos computacionais:



# Processos – Comando “top”

- O comando “top” é um dos mais importantes no gerenciamento de processos no GNU/Linux, sendo importante identificar cada um dos parâmetros exibidos por ele, para facilitar o diagnóstico e a resolução de problemas referentes ao desempenho do sistema Linux.

- A primeira linha (que exibe a saída do comando “uptime”) exibe o horário atual, tempo que o sistema está “no ar” (ligado), quantos usuários estão “logados” e um dos parâmetros mais importantes, o “*load average*”, que mostra quantos processos em média estão aguardando (na fila) para serem executados, sendo que as separações por “vírgula” representam os intervalos de tempo de 1, 5 e 15 minutos.

- A segunda linha mostra a quantidade de processos bem como status de cada um deles.

# Processos – Comando “top”

- A terceira Linha nos mostra o desempenho do processador e o percentual utilizado em cada uma de suas classificações, conforme descrito abaixo:
  - %us = Percentual utilizado por processos em “modo usuário” (sem Nice).
  - %sy = Percentual utilizado por processos do Kernel.
  - %ni = Processos em “modo usuário” com priorização (Nice).
  - %id = Percentual disponível/ocioso do Processador (ou núcleo).
  - %wa = Aguardando operações de I/O (Disco / Rede).
  - %hi = Percentual utilizado para tratamento de interrupções de Hardware.
  - %si = Percentual utilizado para tratamento de interrupções de Software.
  - %st = Percentual utilizado por um Hypervisor (Execução de VM – *Steal time*).

# Processos – Comando “top”

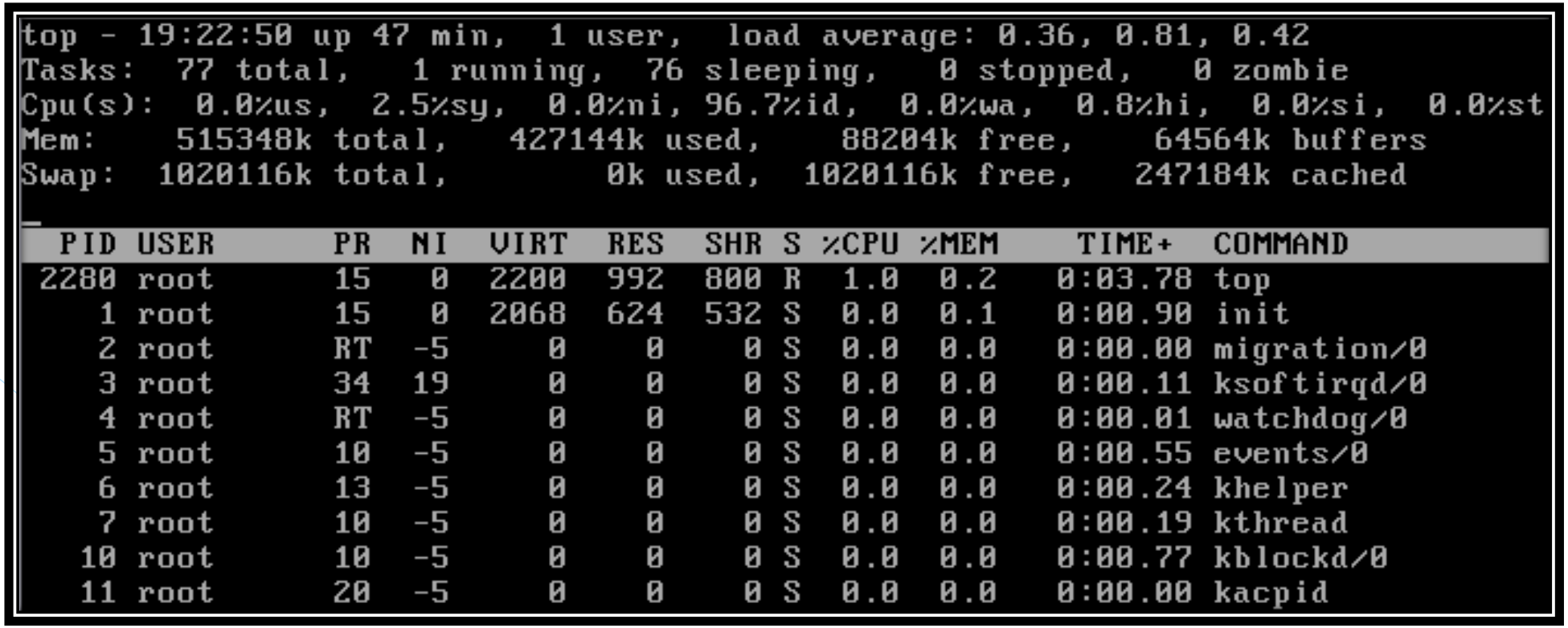
- *PERGUNTA – Através do Print abaixo é possível identificar se o sistema está com alto consumo de processamento no momento?*

```
top - 19:19:30 up 44 min,  5 users,  load average: 2.50, 0.84, 0.31
Tasks:  85 total,   7 running,  78 sleeping,   0 stopped,   0 zombie
Cpu(s):  8.3%us, 74.4%sy,  0.0%ni,  0.0%id,  0.0%wa,  9.9%hi,  7.4%si,  0.0%st
Mem:    515348k total,   508864k used,     6484k free,    62896k buffers
Swap:  1020116k total,        0k used,  1020116k free,   324212k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2345 root      18   0  4556 1112  880 R 11.9  0.2   0:05.35 tar
 2408 root      25   0  4044  944  496 R  9.7  0.2   0:01.87 du
 2409 root      20   0  3840  760  496 R  9.7  0.1   0:01.00 du
 2346 root      18   0  1896  580  224 R  2.8  0.1   0:05.01 gzip
 2280 root      15   0  2200  992  800 R  2.2  0.2   0:01.97 top
  311 root      10  -5     0    0    0 S  0.9  0.0   0:00.70 kjournald
   10 root      10  -5     0    0    0 S  0.3  0.0   0:00.70 kblockd/0
 1581 root      15   0  1724  572  480 S  0.3  0.1   0:00.14 syslogd
 1792 root      18   0  1972  632  556 S  0.3  0.1   0:04.00 hald-addon-stor
 2163 root      15   0 27496 3824 3272 S  0.3  0.7   0:00.32 gdm-rh-security
    1 root      15   0  2068  624  532 S  0.0  0.1   0:00.86 init
    2 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 migration/0
```

# Processos – Comando “top”

- *PERGUNTA – Através do Print abaixo é possível identificar se o sistema está com alto consumo de processamento no momento?*



# Processos – Comando “top”

- ***PERGUNTA 3 – Através do Print abaixo é possível identificar a causa dos valores exibidos para o “load average”?***

```
top - 17:26:13 up 4 days,  7:04,  2 users,  load average: 5.83, 6.00, 6.39
Tasks: 181 total,   2 running, 179 sleeping,   0 stopped,   0 zombie
Cpu0  : 11.6%us,  3.6%sy,  0.0%ni, 20.5%id, 62.3%wa,  0.7%hi,  1.3%si,  0.0%st
Cpu1  :  8.3%us,  3.7%sy,  0.0%ni,  0.0%id, 87.4%wa,  0.0%hi,  0.7%si,  0.0%st
Mem:   3353620k total,  3235808k used,   117812k free,   213216k buffers
Swap:  3004112k total,     2580k used,  3001532k free,  1688612k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 3150 root      18   0  798m 395m 6892 S    5 12.1  46:47.18 java
 2914 mysql     15   0  165m  45m 4144 S    1  1.4   6:31.04 mysqld
25263 vpopmail  18   0  4832 3028 1320 S    1  0.1   0:04.34 imapd
 3103 root      18   0  623m  11m 4112 S    1  0.4   4:50.08 slapd
 2674 vpopmail  16   0  3464 1504 1276 S    0  0.0   0:00.02 imapd
 3116 root      15   0  112m 110m  880 S    0  3.4   1:05.07 authdaemond
 3119 root      15   0  113m 110m  880 S    0  3.4   1:04.83 authdaemond
29959 vpopmail  18   0  4388 2552 1332 D    0  0.1   0:01.73 imapd
  214 root      15   0     0    0    0 S    0  0.0   2:11.80 pdflush
 1053 root      10  -5     0    0    0 D    0  0.0   1:12.40 md0_raid1
 2439 root      15   0  1704  612  520 S    0  0.0   2:27.48 syslogd
```

# Processos – Comando “ps”

- ps → Exibe por padrão apenas os processos executados em nosso terminal (tty):
  - Ex.: ps <opções>

```
[root@localhost ~]# ps
  PID TTY          TIME CMD
 2217 tty1     00:00:00 bash
 2528 tty1     00:00:00 ps
```

- A resposta deste comando é dividida em 4 ou 5 colunas (depende de parâmetros), sendo que cada uma delas retornam informações como:
  - 1ª → PID do processo;
  - 2ª → TTY em que o mesmo está sendo executado (sessão/terminal);
  - 3ª → STATus do processo;
  - 4ª → Tempo que este processo utilizou recursos do processador (em horas);
  - 5ª → O comando utilizado para a execução deste processo.

# Processos – Comando “ps”

- ps → Continuação...:
  - Geralmente ele é mais utilizado com os parâmetros “ax”, que possibilitam a visualização de todos os processos em execução no sistema e seu “STATus”.

```
[root@localhost ~]# ps ax
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:00 init [5]
    2 ?        S<     0:00 [migration/0]
    3 ?        SN     0:00 [ksoftirqd/0]
    4 ?        S<     0:00 [watchdog/0]
    5 ?        S<     0:00 [events/0]
    6 ?        S<     0:00 [khelper]
    7 ?        S<     0:00 [kthread]
   10 ?        S<     0:00 [kblockd/0]
   11 ?        S<     0:00 [kacpid]
   48 ?        S<     0:00 [cqueue/0]
   51 ?        S<     0:00 [khubd]
   53 ?        S<     0:00 [kseriod]
  115 ?        S      0:00 [pdflush]
```

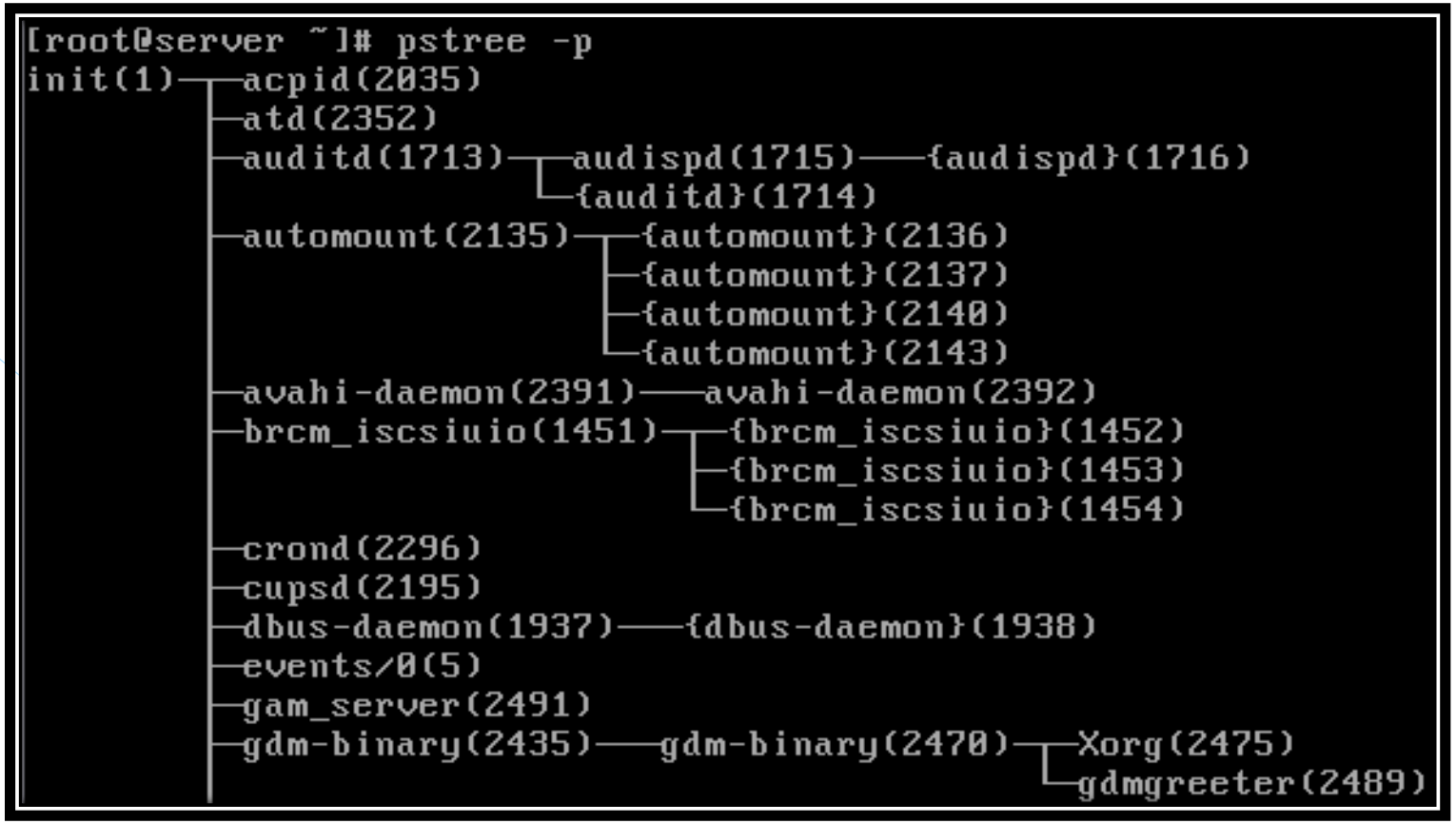
# Processos – Comando “ps”

- ps → Continuação... ***O que representa cada letra em STATus??***:

| Sigla | Descrição do STATUS |
|---|---|
| D | O processo está "dormindo" ininterruptivelmente |
| R | Processo em execução na CPU no momento |
| S | Parado ou interrompido ("dormindo" esperando um sinal para voltar a execução) |
| T | Parado por um sinal de controle de job (tarefa) ou sob análise |
| W | Paginação, não será utilizado |
| X | Processo foi morto e nunca mais visto |
| Z | Processo em estado "zombie" (defunto - Já foi executado porém, permite ao processo que o criou ler seu valor de saída, através do SIGCHLD) |
| < | Processo com alta prioridade |
| N | Processo com baixa prioridade |
| L | Processo com página bloqueada em memória |
| s | Processo "líder" de sessão (como o Shell Bash por exemplo) |
| l | Processo multi-thread |
| + | Processo está em foreground |

# Processos – Comando “pstree”

- pstree → Exibe de forma hierárquica a relação entre os processos (processo pai e processos filhos) - O parâmetro mais comum é:
  - “-p” → Mostra o PID entre parênteses;



# Conceitos sobre processos – Parte 2

Tópico 8: Gerenciamento de processos.

# Conceitos – Sinais de um processo

- Ao realizarmos determinadas interações com um processo, como encerramento do processo de forma padrão ou abrupta, bem como interromper sua execução com uma “pausa”, estamos enviando um sinal com a instrução desejada;

- O padrão POSIX atribui um número para cada tipo de sinal, que pode ser utilizado e enviado para um processo em execução;

- A tabela a seguir descreve o objetivo e a ação de cada um destes sinais:

# Conceitos – Sinais de um processo

| Sinal | Valor | Ação | Descrição do sinal |
|---|---|---|---|
| SIGHUP | 1 | A | Travamento no terminal controlador ou morte do processo controlador |
| SIGINT | 2 | A | Interrupção originária do teclado via [CTRL] + [c] |
| SIGILL | 4 | C | Instrução ilegal |
| SIGABRT | 5 | C | Sinal de Abortar, enviado pela função *abort* |
| SIGKILL | 9 | AEF | Destruir o processo - Interrupção de emergência |
| SIGTERM | 15 | A | Encerrar o processo de forma normal |
| SIGCHLD | 17 | B | Processo filho parado ou terminado |
| SIGCONT | 18 | | Continuar se interrompido - Podemos citar como exemplo a utilização dos comandos "bg" e "fg" após um [CTRL] + [z] |
| SIGTSTP | 20 | D | Interromper o processo (Parar/Pausar) - [CTRL] + [z] |

| Ação | Descrição da Ação em relação ao processo |
|---|---|
| A | Terminar o Processo |
| B | Ignorar o sinal recebido |
| C | Terminar o Processo e mostrar o "core" |
| D | Parar o processo |
| E | O sinal não pode ser pego/obtido pelo processo |
| F | O sinal não pode ser ignorado |

- ***OK, Porém, como utilizar??***
  - Próximo slide...

# Comandos para gerenciamento de processos – Parte 2

Tópico 8: Gerenciamento de processos.

# Encerrando Processos
# Comandos “kill” e “killall”

- kill → Permite enviar um sinal para um processo através do seu PID:
  - Ex.: kill -9 2130
    - O comando acima envia o sinal de encerramento abrupto SIGKILL (valor 9) para o processo que possuir o PID 2130;

```
[root@server ~]# sleep 30 &
[1] 10337
[root@server ~]# kill -9 10337
[root@server ~]#
[1]+  Killed                    sleep 30
```

- killall → Permite enviar um sinal para um ou mais processos através do nome:
  - Ex.: killall -15 top
    - O comando acima envia o sinal SIGTERM (valor 15) para todos os processos de nome “top” que estiverem em execução no sistema.

# Planos de Execução
# Comandos “bg”, “fg” e “jobs”

- Ao executar comandos no Shell, os comandos são executados em *Foreground* por padrão, impossibilitando realizar outras tarefas.

- Para evitar esta ociosidade, podemos executar um comando em *Background* e continuar trabalhando com o sistema na mesma sessão. Exemplo:

```
[root@server ~]# tar czf BKP.tar.gz /etc 2> /tmp/bkp-erro.log > /tmp/bkp.log &
[1] 10345
```

- No exemplo acima, temos o comando “**tar**” em execução, enviando possíveis saídas de erro ou de sucesso para outros arquivos, porém, com o “**&**” no final da linha de comando, fazendo com que o mesmo seja executado em ***Background***.

# Planos de Execução
# Comandos “bg”, “fg” e “jobs”

- Caso um comando seja executado em *Foreground*, podemos enviá-lo para *Background* através da seguinte sequência:
  - Pressionar **[CTRL] + [z]**, para interromper a execução do processo e obter o controle do Shell novamente;
  - Verificar o número do “job” interrompido e enviá-lo para *Background* com o comando “**bg**”, ou retorná-lo a *Foreground* com o comando “**fg**”;

- bg → Envia um processo (job) interrompido para *Background*:
  - Ex.: bg <nº>
- fg → Envia um processo (job) interrompido para *Foreground*:
  - Ex.: fg <nº>
- jobs → Lista os processos interrompidos:
  - Ex.: jobs -l

# Prioridade de Processos
# Comandos “nice” e “renice”

- nice → Possibilita executar um processo com uma prioridade específica:
  - Ex.: nice -n <valor> [comando]

```
[root@server ~]# nice -n 10 sleep 300 &
[4] 10401
```

    - O comando acima executa o processo “**sleep 300 &**” com a prioridade 10;
    - **OBS.: O valor da prioridade varia de “-20” a “19”, sendo que quanto menor o valor, mais prioridade terá o processo durante a execução;**

- renice → Permite alterar a prioridade de um processo já em execução:
  - Ex.: renice -15 10401

```
[root@server ~]# renice -15 10401
10401: old priority 10, new priority -15
```

    - O comando acima alterou a prioridade do processo com o PID “10401” do valor “**10**” para o novo valor de prioridade “**-15**”;

# Processos – Comando “nohup”

- nohup → Permite manter a execução de um processo caso o mesmo receba um sinal SIGHUP (valor 1), ou seja, o processo que estiver em *Foreground* ou *Background* continuará em execução mesmo após efetuar o *logoff* em uma sessão remota por exemplo:
  - Ex.: nohup [comando]

```
[root@server ~]# nohup /usr/scripts/backup.sh
```

    - O comando acima faz com que um script permaneça em execução mesmo após o fechamento da “tty” (sessão de acesso remoto).
    - Muito útil em intervenções remotas no dia-a-dia do administrador Linux.

# Curiosidades sobre Processos

- Na versão 2.4.x do Kernel Linux, os números de PID eram contabilizados em sequência até o número 32.000;

- Na versão 2.6.x, este número mudou para 1 bilhão;
  - Após esgotar os números disponíveis, o contador volta ao início, utilizando os números disponíveis;

# No próximo slide…

- Tópico 9: Sistemas de arquivos e particionamento.
  - Conceitos sobre sistemas de arquivos, dispositivos e partições.
  - Comandos para gerenciamento de disco e sistemas de arquivos.

# Referências


- BONAN, Adilson Rodrigues. **LINUX – Fundamentos, Prática & Certificação LPI**. Editora: Alta Books. RJ. 2010;

- PEREIRA, Guilherme Rodrigues. **Slides para aula expositiva**. Centro Universitário UNA.

- SILVA, Gleydson Mazioli. **Guia Foca GNU/Linux**. Disponível em: https://guiafoca.org/

- TANENBAUM, Andrew S. **Sistemas operacionais modernos**. 4. ed. São Paulo: Pearson, 2016.

- About.com Linux – top Linux Command. Disponível em: http://linux.about.com/od/commands/l/blcmdl1_top.htm

DGP

Tecnologia da Informação

# Obrigado!

Guilherme Rodrigues