

Aula 00

*Arquitetura e Sistemas Operacionais p/
Receita Federal (Analista de Informática)
- 2020*

Autor:

**Equipe Informática e TI, Evandro
Dalla Vecchia Pereira**

28 de Dezembro de 2019

Características e Configurações do Sistema Operacional Linux	2
<i>Sistema Multitarefa e Multiusuário</i>	<i>4</i>
<i>Kernel, Shell e Servidor de Janelas</i>	<i>4</i>
<i>Execução em Foreground ou Background.....</i>	<i>6</i>
<i>Agendamento de Tarefas</i>	<i>7</i>
<i>Sistemas de Arquivos.....</i>	<i>8</i>
<i>LVM (Logical Volume Manager).....</i>	<i>10</i>
<i>Estrutura dos Diretórios</i>	<i>11</i>
<i>Memória Virtual</i>	<i>11</i>
<i>Gerenciadores de Pacotes</i>	<i>12</i>
<i>Configuração de Serviços ou Funcionalidades</i>	<i>15</i>
<i>Interoperabilidade.....</i>	<i>17</i>
<i>Serviço de Diretório</i>	<i>19</i>
<i>Questões Comentadas</i>	<i>22</i>
Usuários, grupos, permissões, controles de acesso e manipulação de arquivos e diretórios	43
<i>Questões Comentadas</i>	<i>47</i>
Shell Script.....	51
<i>Questões comentadas</i>	<i>54</i>
Gerenciamento de Processos.....	59
<i>Questões comentadas</i>	<i>63</i>
Superusuário	65
<i>Questões comentadas</i>	<i>67</i>
Lista de Questões	69
GABARITO	92





PROF. EVANDRO DALLA VECCHIA

Autor do livro "Perícia Digital - Da investigação à análise forense", Mestre em Ciência da Computação (UFRGS), Bacharel em Ciência da Computação (PUCRS), Técnico em Redes de Computadores (Ecom/UFRGS) e em Processamento de Dados (Urcamp). Perito Criminal na área de Perícia Digital desde 2004 no Instituto-Geral de Perícias/RS. Professor de pós-graduação em diversas instituições, nas áreas de Perícia Digital, Perícia Criminal e Auditoria de Sistemas. Lecionou na graduação de 2006 a 2017, nas instituições PUCRS, Unisinos, entre outras. Professor em cursos de formação e aperfeiçoamento de Peritos Criminais, Delegados, Inspetores, Escrivães e Policiais Militares.

Áreas de cursos ministrados pelo professor no Estratégia: Computação Forense, Arquitetura de Computadores e Sistemas Operacionais.

Entre em contato:   profevandrodallavecchia

CARACTERÍSTICAS E CONFIGURAÇÕES DO SISTEMA OPERACIONAL LINUX

Buenas, tudo belezinha? Vamos começar nossa aula com as características básicas do Linux, aquelas que quem já estuda sistemas operacionais conhece as classificações, então vamos ver onde o Linux se “encaixa”.

Começamos dizendo que o **Linux é um sistema operacional interativo**, ou seja, diferentemente dos sistemas em lote ou os sistemas de tempo real, ele permite uma interação do usuário diretamente com o computador, mesmo durante a execução de um programa. Essa interação ocorre através de dispositivos de entrada (teclado, mouse, entre outros) e saída (monitor de vídeo, impressora, entre outros).

O **usuário “administrador”, que tem “poder total” sobre o sistema é o root e seu diretório “home” é o “/root”**. Os usuários “comuns” têm como seu diretório “home” o “/home/nome_usr”, ex.: “/home/evandro”.

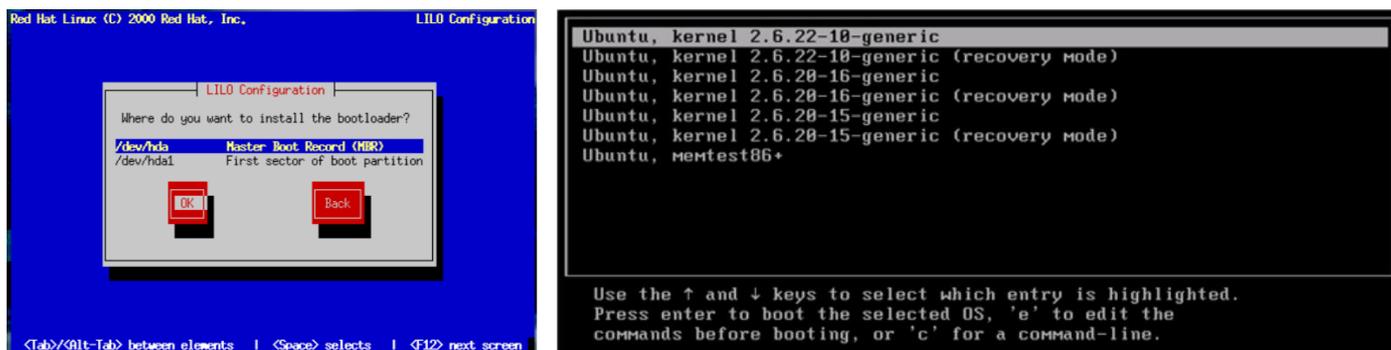
A instalação do Linux pode ocorrer de algumas formas, sendo as mais comuns através de uma mídia (CD, DVD, pen drive) ou através da rede. No caso da mídia, obviamente ela precisa ser “bootável” e a BIOS deve estar configurada para ler primeiro o CD/DVD ou a USB (se for pen drive). Abaixo uma tela de instalação a partir de um CD.





A instalação pela rede exige mais conhecimento e geralmente é feita por técnicos de TI nas empresas. É necessário configurar um servidor TFTP e um DHCP que irá fornecer a mídia de instalação para as máquinas da sua rede local. Se a BIOS da máquina cliente suportar, é possível então inicializar o sistema de instalação a partir da rede (usando PXE e TFTP) e prosseguir com a instalação a partir da rede.

Muitas pessoas utilizam o Linux e o Windows no mesmo computador, cada um em uma partição (com seus respectivos sistemas de arquivos). E como é possível escolher qual deles deve ser inicializado quando a máquina é ligada? Existem **gerenciadores de boot** que auxiliam nisso, sendo os mais conhecidos o **LILLO** (*L*inux *L*oader) e o **GRUB** (*G*rand *U*nifield *B*ootloader). Abaixo podemos ver os dois (à esquerda o LILLO e à direita o GRUB).



Ao escolher a partição, o *kernel* (veremos em breve o que é e para que serve) é carregado e inicializado. Na sequência há a execução dos scripts de inicialização do sistema.

Em relação à **portabilidade**, o Linux se sai muito bem! Ele não foi pensado para ser um sistema portátil, mas acabou indo nessa direção, tendo um dos núcleos de sistemas operacionais mais portáteis, sendo executado em sistemas desde o iPaq (um computador portátil) até o IBM S/390 (um mainframe). Na atualidade o Linux hoje funciona em dezenas de plataformas, desde mainframes até um relógio de pulso, passando por diversas arquiteturas: x86 (Intel, AMD), x86-64 (Intel EM64T,



AMD64), ARM, PowerPC, Alpha, SPARC, entre outras, além de sistemas embarcados (*handhelds*, consoles de videogames, PVR – *Personal Video Recorder*, telefones celulares, TVs etc.).

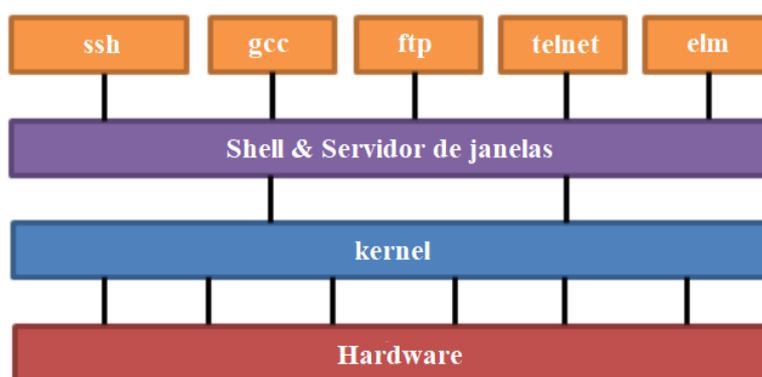
SISTEMA MULTITAREFA E MULTIUSUÁRIO

O Linux permite que vários processos sejam executados ao mesmo tempo (**multitarefa**), mesmo que haja apenas um processador na máquina. O sistema operacional faz o escalonamento dos processos no processador, dando a ilusão de um paralelismo (ocorre um pseudoparalelismo). O comando “ps”, por exemplo, pode ser executado e você verá diversos processos em execução. Quando há a criação de um processo novo, ele passa a ser filho de outro e a chamada de sistema utilizada é a `fork()`.

No Linux é possível que vários usuários utilizem ao mesmo tempo (**multiusuário**), então alguém pode deixar algumas aplicações em execução em sua sessão, bloquear o acesso e outra pessoa abrir uma outra sessão, executando novas aplicações. São permitidos vários usuários simultâneos, a não ser que seja definido um limite de sessões nos arquivos de configuração. O comando “who” pode ser executado para ver quem está logado no computador.

KERNEL, SHELL E SERVIDOR DE JANELAS

O **kernel** (núcleo) da maioria dos sistemas baseados em Unix (FreeBSD e derivados, Oracle Solaris, IBM AIX, HP-UX e todos os Linux e derivados) é **monolítico**. Mas o que é isso? Todo o conjunto de instruções de controle do hardware é executado no espaço de núcleo no modo de supervisão, ou seja, é um **único executável que possui todos os códigos de suporte necessários agregados**. Em teoria, um kernel monolítico pode travar o sistema inteiro se 1 módulo parar de funcionar adequadamente.



Na figura acima podemos ver cinco aplicativos que utilizam alguma interface (shell ou servidor de janelas), que por sua vez faz a “ligação” com o kernel. E somente o kernel faz a iteração com o hardware (disco, teclado, monitor de vídeo etc.).

O shell é um interpretador de comandos (aquele terminal preto, geralmente) e sua função é ler a linha de comando, interpretar seu significado, executar o comando e devolver o resultado pelas saídas (monitor de vídeo, por exemplo). Na verdade, a interface shell é um arquivo executável, encarregado de interpretar comandos, enviá-los ao kernel e devolver resultados. Existem vários



tipos de shell, sendo os mais comuns o sh (Bourne shell), o bash (Bourne again shell), o csh (C shell), o Tcsh (Tenex C shell), o ksh (Korn shell) e o zsh (Zero shell). Seus nomes normalmente correspondem ao nome do executável.

Alguns dos servidores de janelas (ambientes gráficos) mais populares do mundo Linux são Gnome, KDE, Cinnamon, MATE, XFCE, Pantheon, Deepin, entre outros. Abaixo podemos ver uma tela do Gnome (à esquerda) e uma tela do KDE (à direita)



Uma característica interessante, por ser um sistema operacional de código aberto, é a possibilidade de **compilar o kernel**. Mas qual o motivo para isso? Para facilitar vamos a um exemplo: suponha que você tenha uma placa-mãe X e sua placa de rede integrada é uma da fabricante Y. No Kernel que você está usando (uma versão mais antiga) não há suporte para essa placa Y, ou seja, você não pode navegar na rede. Lendo a documentação do Kernel de uma versão mais recente, você constata que o suporte para Y foi adicionado. Então, se o kernel for atualizado será possível navegar na rede com tal placa com esse novo módulo.

Os **módulos são partes do kernel** que são carregadas somente quando solicitadas por algum aplicativo ou dispositivo e descarregadas da memória quando não são mais utilizadas. Dessa forma, evita-se a construção de um kernel grande (estático) que ocupe grande parte da memória com todos os drivers compilados e permite que partes do kernel ocupem a memória somente quando forem necessários.

A localização padrão dos módulos do kernel é a seguinte: `"/lib/modules/versão_do_kernel/"` e um comando que permite obter informações acerca do servidor é o `"uname"`. Com o argumento `"-a"` é possível obter todas as informações, incluindo a versão do kernel. Abaixo um exemplo mostrando a versão do kernel (4.12):

```
[root@localhost lib]# uname -a  
Linux localhost 4.12.0-rc6-g48ec1f0-dirty #21 Fri Aug 4 21:02:28 CEST 2017 i586  
GNU/Linux  
[root@localhost lib]#
```



EXECUÇÃO EM FOREGROUND OU BACKGROUND

Os processos podem ser executados de duas formas: em primeiro plano (*foreground*) ou em segundo plano (*background*). Os processos que executam em *foreground* são os que necessitam de interação direta com o usuário, incluindo a troca de informações. Os processos em *background* não necessitam dessa interação com o usuário.

Há situações em que é necessário passar um processo que está sendo executado em *foreground* para *background* e vice-versa. Por exemplo, em uma sessão de transferência de arquivos entre computadores remotos, a baixa velocidade da linha de transmissão pode fazer com que o tempo de transferência leve muito tempo. Assim, seria interessante passar o processo para segundo plano, liberando o *shell* para outras atividades do usuário.

A passagem de um processo de *foreground* para *background* é realizada primeiro suspendendo o processo (CTRL + Z), seguido do comando "bg", que envia o processo para segundo plano (*background*). Observe que suspender a execução de um processo não significa finalizá-lo, apenas torná-lo temporariamente inativo!

A lista dos processos executados em *background* pode ser visualizada com o comando "jobs", que mostra o número da tarefa (*job*) associada. Se o usuário quiser interagir novamente com o processo, deve utilizar o comando "fg" seguido de % número_job. Exemplo:

```
estrategia:~$ jobs
[1] - Running
script.sh
[2] + Suspended (tty output)  teste.sh
estrategia:~$ fg %2
```

O mais utilizado para executar em *background* é utilizar "&" logo após o nome do binário ou script. Por exemplo, se você deseja executar um script "aprovacao.sh", que vai realizar vários cálculos e no fim escrever em um arquivo, não é necessário que você fique com o *shell* "travado", aguardando a conclusão da tarefa. Então é só digitar (note que o *shell* não fica esperando a conclusão do *script*, já aguarda o próximo comando):

```
estrategia:~$ aprovacao.sh &
estrategia:~$
```

Processos que tipicamente só são executados em *background*, pois não possuem interação com o usuário, são os *daemons*. Geralmente possuem um nome que terminam com a letra "d", como por exemplo, *syslogd* (*daemon* que gerencia o *log* do sistema). Em um ambiente Unix, o "processo pai" de um *daemon* é normalmente o processo *init* (PID=1). De forma geral, os sistemas operacionais iniciam *daemons* durante o processo de *boot*.



AGENDAMENTO DE TAREFAS

Para o agendamento de tarefas a aplicação mais conhecida é o **Cron**, que já existe há muito tempo e passou por muitos estágios de evolução. Porém, a maioria das suas implementações atuais (Vixie Cron, ISC Cron, BCron etc.) ainda se baseiam no pressuposto de que o sistema está em funcionamento de forma ininterrupta. Isso é um problema quando se utiliza a virtualização, quando os sistemas que operam sob demanda, sendo ligados e desligados conforme sua necessidade.

Além disso, os usuários de *notebooks* suspendem ou desligam seus equipamentos para transportá-los, e até mesmo *desktops* e estações de trabalho são desligados para poupar energia. O resultado disso é que os *cronjobs* falham regularmente, pois o sistema se encontra desligado no momento em que a tarefa deveria ser executada!

Para contornar essa situação, algumas distribuições contam com o Anacron como alternativa ao Cron. O Anacron permite a criação de listas de tarefas que serão realizadas em intervalos pré-definidos e, quando o Anacron é inicializado, ele verifica essas listas e executa as tarefas ainda não realizadas. No entanto, o Anacron possui algumas limitações. A primeira é que ele não é um *daemon*, então precisa ser executado sempre que for necessário.

Outra questão é que o Anacron não está preparado para lidar com períodos de tempo menores do que dias. A combinação desses problemas pode levar a situações nas quais o Cron e o Anacron funcionam ao mesmo tempo, e algumas tarefas podem ser executadas duas vezes ou nenhuma!

Uma solução é o Fcron, que faz o que o Vixie Cron e o Anacron fazem e ainda mais. É possível usar o Fcron para agendar *cronjobs* com data e hora fixas, com intervalos de tempo ou até mesmo de acordo com a disponibilidade do sistema.

Independente da implementação utilizada, a configuração é muito parecida, e vale destacar também que o Cron (ou uma de suas variantes) é um programa de nível multiusuário, ou seja, cada usuário pode agendar suas tarefas individualmente. Isso ocorre graças ao próprio arquivo onde são armazenados os scripts: o **Crontab**.

Como o próprio nome sugere, a estrutura do Crontab consiste em tabelas e nessas tabelas são preenchidas todas as informações referentes à tarefa: **minutos; horas; dias do mês; mês; dias da semana; comando**. Abaixo um exemplo do comando ("crontab -l" lista a crontab):

```
crontab -l  
55 * * * 5 /home/root/bin/backup.sh
```

Agora vamos analisar alguns exemplos comentados:

executa cinco minutos depois da meia-noite, todos os dias (* = tudo)

```
5 0 * * * $HOME/bin/script.sh
```



executa às 14h15min no dia 1º de cada mês

```
15 14 1 * * $HOME/bin/script.sh
```

executa às 22h nos dias de semana (1 a 5 = segunda-feira a sexta-feira)

```
0 22 * * 1-5 $HOME/bin/script.sh
```

SISTEMAS DE ARQUIVOS

Quando falamos em armazenamento de informações a longo prazo temos três requisitos:

1. Deve ser possível armazenar um grande volume de informações;
2. As informações devem “sobreviver” ao término do processo (programa em execução);
3. Vários processos devem ser capazes de acessar as informações ao mesmo tempo (ex.: vários programas acessando o mesmo banco de dados).

A solução mais comum é armazenar as informações em mídias de armazenamento (HDs, SSDs, entre outras), em unidades chamadas **arquivos**. Tais informações armazenadas em arquivos devem ser persistentes, ou seja, não devem ser afetadas pela criação ou término de um processo. O arquivo geralmente só pode “sumir” se for excluído por seu criador (ou pelo administrador do sistema).

O sistema operacional faz o gerenciamento dos arquivos (estrutura, nome, como acessar, usar etc.) através do **sistema de arquivos**. Enquanto o usuário final “enxerga” os arquivos através de seus nomes, localizações, tamanhos, tipos, proprietários etc., os projetistas dos sistemas de arquivos devem se preocupar também com aspectos mais técnicos, como por exemplo se vão ser utilizadas listas encadeadas ou mapa de bits para monitorar o espaço de armazenamento livre.

Mesmo lidando com sistemas de arquivos típicos do Windows (FAT32, NTFS etc.), os sistemas de arquivos nativos do Linux são, entre outros:

EXT (EXTended File System): durante algum tempo o **EXT2** foi o sistema de arquivos padrão do Linux, herdando características de outros sistemas de arquivos e sendo baseado no UFS (usado no FreeBSD e Solaris), além de possuir semelhanças com os sistemas de arquivos HFS e HFS+ (Apple).

A menor unidade de alocação do Ext é o bloco (o equivalente ao *cluster* em sistemas de arquivos FAT e NTFS), cujos tamanhos possíveis são 1024, 2048 e 4096 (escolhido no processo de formatação).



Quando falamos especificamente da família de sistemas de arquivos EXT, temos que ter em mente que a versão 2 não possuía *journaling*. Somente na versão 3 surgiu! Em seguida vamos ver mais detalhes.

A principal característica do **EXT3** é o uso do recurso de *journaling*, onde o sistema de arquivos mantém um *journal* (diário) das alterações realizadas. Esse "diário" armazena uma lista das alterações realizadas, permitindo que o sistema de arquivos seja reparado de forma muito rápida após o desligamento incorreto. O **Ext3** possui três modos de operação:

- **ordered** (default): o *journal* é atualizado no final de cada operação. Isso faz com que exista uma pequena perda de desempenho, já que a cabeça de leitura do HD precisa realizar duas operações de gravação, uma no arquivo que foi alterado e outra no *journal* (que é um arquivo especialmente formatado);
- **writeback**: o *journal* armazena apenas informações referentes à estrutura do sistema de arquivos (metadados) e não em relação aos arquivos propriamente ditos, é gravado de forma mais ocasional, aproveitando os momentos de inatividade.
- **journal**: é o mais seguro, todavia mais lento. Nesse modo, o *journal* armazena não apenas informações sobre as alterações, mas também uma cópia de segurança de todos os arquivos modificados, que ainda não foram gravados no disco. Por ser o mais lento, é o modo menos usado.

Para finalizar a família EXT, existe o **EXT4**, com melhorias que incluem, por exemplo, a ampliação do endereçamento para **48 bits**, o que permite endereçar um volume virtualmente ilimitado de **blocos e partições de até 1 exabyte (1 EB = 1024 petabytes)**! **O limite de 2 TB para os arquivos também foi removido**, abrindo espaço para o armazenamento de arquivos gigantescos (**no máximo 1 EB**, o mesmo tamanho do volume!).

Além do EXT2, EXT3, EXT4, existem outros sistemas de arquivos conhecidos (e cobrados em provas de concurso), a saber:

- **Reiser FS**: utilizado geralmente no Linux, foi o primeiro com suporte a journaling (incluído no núcleo Linux 2.4). Seu futuro é incerto depois que seu criador, Hans Reiser, foi condenado pelo assassinato de sua esposa. Há voluntários que continuam com o projeto;
- **XFS**: inicialmente desenvolvido pela Graphics para o seu sistema operacional IRIX, posteriormente teve seu código fonte liberado e foi adaptado para funcionar no Linux. É um sistema de arquivos desenvolvido em 64 bits, compatível com sistemas de 32 bits. Em plataformas de 64 bits, possui um limite de tamanho de 8 EB para um volume e para cada arquivo. É um sistema de arquivos com journaling;
- **Btrfs** (B-tree file system) é um sistema de arquivos baseado no princípio cópia em gravação (copy-on-write - COW), inicialmente desenvolvido pela Oracle para ser usado no Linux. Foi projetado para solucionar problemas como a falta de agrupamento de discos ou volumes, snapshots, checksums e uso de múltiplos volumes simultaneamente nos sistemas de arquivos do Linux. Possui a limitação de tamanho de volume igual à limitação de tamanho de arquivo (16 EB);
- Red Hat **GFS**: é um sistema de arquivos de cluster que permite que um cluster acesse simultaneamente um disparador de obstáculo do qual é dividido entre os nós. Trata-se de um sistema de arquivos nativo que se conecta diretamente por meio da interface com a camada VFS da interface de sistema de arquivo Kernel Linux. O GFS emprega os metadados distribuídos e diários múltiplos para uma operação mais eficiente em um cluster;



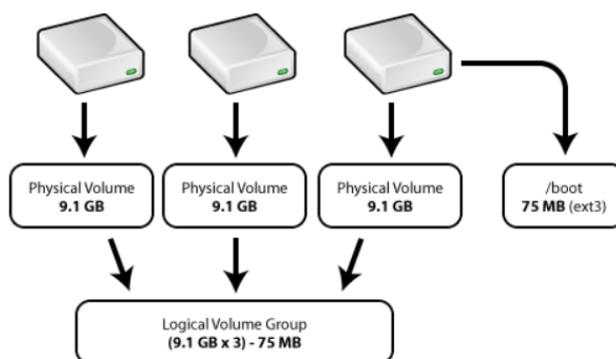
- **NFS** (Network File System): protocolo de sistema de arquivos distribuído, originalmente desenvolvido pela Sun Microsystems, que permite que um usuário em um computador cliente acesse arquivos através de uma rede como se estivesse acessando na máquina local. Utiliza o sistema Open Network Computing Remote Procedure Call (ONC RPC);
- **Swap**: memória virtual do Linux que tem uma partição específica (diferente do Windows que utiliza um arquivo de paginação). Possui uma organização própria, sem utilizar um sistema de arquivos (ou pode ser entendido como tendo um sistema de arquivos próprio para isso).

LVM (LOGICAL VOLUME MANAGER)

O gerenciador de volumes lógicos (LVM) é um recurso que visa facilitar a vida dos administradores de sistema. Embora os usuários domésticos geralmente prefiram criar apenas uma ou duas partições ("/" ou "/" + /home") durante a instalação do sistema, nos servidores são utilizadas muito mais partições por questões de segurança e desempenho.

O problema é que ao dividir o HD em várias partições diferentes surge o problema do aproveitamento do espaço, já que sempre algumas partições ficam cheias embora ainda exista muito espaço livre nas outras. Geralmente é possível redimensionar as partições, mas esse é sempre um processo demorado e que traz risco de perda de dados.

O LVM resolve o problema permitindo agrupar vários HDs em uma única unidade que pode ser dividida em vários volumes lógicos, vistos pelo sistema operacional como se fossem partições de disco. A grande vantagem é que novos volumes podem ser criados, excluídos ou redimensionados rapidamente e sem a necessidade de reiniciar o servidor.



Caso sejam usados HDs SCSI com suporte a hot-swap, é possível adicionar, remover ou ainda substituir HDs, fazendo as alterações necessárias nos volumes lógicos, tudo sem interrupções! O fato do sistema “visualizar” todos os HDs instalados como uma única unidade também facilita os backups.

Para utilizar o LVM é preciso compilar o Kernel ativando as opções “Multiple devices driver support (RAID and LVM)” e “Logical volume manager (LVM) Support”. Em geral as distribuições já trazem esses dois componentes compilados como módulos, bastando ativá-los usando o comando “modprobe”.



ESTRUTURA DOS DIRETÓRIOS

Uma característica forte do Linux é a sua estrutura de diretórios, que pode possuir pequenas modificações entre as distribuições, porém são semelhantes. Abaixo é mostrada tal estrutura e o que é armazenado, por padrão, em cada diretório/subdiretório:

- `"/etc"`: dados de configuração (scripts de inicialização, tabela de sistemas de arquivo, configurações de login, configuração da fila de impressão, entre outros);
- `"/etc/network/interfaces"`: configuração de rede (IP fixo ou DHCP, entre outros);
- `"/etc/resolv.conf"`: servidores DNS;
- `"/etc/ssh/sshd_config"`: configuração do SSH;
- `"/usr"`: maior parte dos aplicativos e bibliotecas do sistema;
- `"/usr/bin"`: executáveis dos programas;
- `"/usr/lib"`: bibliotecas e arquivos compartilhados;
- `"/usr/src"`: código-fonte de programas;
- `"/usr/doc"`: documentação em geral;
- `"/home"`: arquivos dos usuários (arquivos de trabalho, músicas, filmes, downloads, etc.);
- `"/var"`: sites hospedados (nem sempre), bases de dados do MySQL, etc.;
- `"/var/log"`: logs de diversos serviços;
- `"/var/www"`: sites hospedados (nem sempre);
- `"/boot"`: imagem do Kernel e o `initrd` (initial ram disk), carregados no início do boot;
- `"/bin"`: comandos básicos, como `"cd"`, `"ls"` e `"cat"`;
- `"/lib"`: módulos do Kernel e bibliotecas básicas do sistema;
- `"/opt"`: arquivos de aplicativos adicionais, que não são essenciais para o sistema. Cada aplicativo tem uma subpasta com seu respectivo nome;
- `"/proc"`: informações de depuração do Kernel, configurações que habilitam e desabilitam o suporte à algum elemento no Kernel;
- `"/tmp"`: arquivos temporários.

MEMÓRIA VIRTUAL

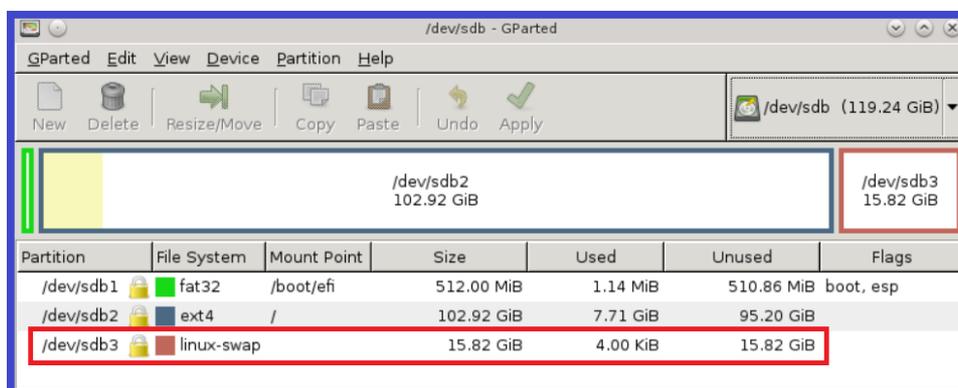
O conceito de memória virtual está presente nos sistemas operacionais modernos (incluindo o Linux, claro). Sabemos que um programa necessita passar pela memória principal para ser executado, mas e se não há memória RAM suficiente para executar todo o programa? Aí surgiu a ideia de executar o programa em partes e apenas as partes que necessitam ser executadas no momento ficam na memória RAM.



A **memória virtual** é uma área de troca de dados que serve como uma extensão da memória RAM. Por exemplo, se há 20 MB de memória RAM livre e um programa de 30 MB é executado, uma parte é carregada e o restante fica na memória virtual, para depois serem copiados para a memória principal a fim de serem executados.

Isso ocorre à medida que a execução do programa for acontecendo e o programa em questão (ou outros programas) liberarem a memória que ocupavam para o uso de outro programa. Podemos verificar que há um fluxo contínuo de dados entre o disco rígido (onde fica a memória virtual) e a memória RAM. Por isso concluímos que pouca memória RAM implica em usar muita memória virtual, que por sua vez, implica em usar muita leitura/escrita no disco rígido, deixando o funcionamento geral do sistema mais lento!

Geralmente ao instalar qualquer distribuição GNU/Linux, na opção automático da partição do disco onde será instalado o sistema, também encontramos a criação automática da **partição Swap**. Se optar por uma instalação personalizada, deve-se atentar para a criação de uma partição e marcá-la como Swap. Abaixo é mostrada uma partição Swap com cerca de 16 GB através do Gparted (utilitário que lista e permite alterar partições, bastante utilizado).



Quando não há partição livre ou não há espaço para a criação da memória virtual e o sistema operacional necessita de memória de troca para o melhor desempenho dos processos é possível criar o **arquivo de memória Swap**. Nada impede que ambas as soluções sejam utilizadas simultaneamente!

GERENCIADORES DE PACOTES

Segundo Morimoto, os **pacotes são as “peças” que formam todas as distribuições Linux** e podem conter programas, bibliotecas de sistema ou mesmo algumas outras “coisas” como papéis de parede e ícones. Alguns programas grandes (KDE, por exemplo) são divididos em vários pacotes para que o usuário possa instalar apenas as partes que interessam, ficando com um sistema mais enxuto.

Alguns pacotes dependem de outros (um programa pode precisar de uma biblioteca que faz parte de outro pacote, por exemplo), as chamadas **dependências**. Para evitar que algo não funcione ou com pacotes desnecessários, o instalador automaticamente verifica as dependências de cada pacote, adicionando ou removendo pacotes relacionados a ele. É por isso que às vezes ao marcar um determinado pacote alguns outros são marcados junto.



Instalar ou remover pacotes no Linux hoje em dia é muito fácil, pois existem vários gerenciadores de pacotes para facilitar essa tarefa. Vamos ver apenas quatro deles, pois são os geralmente cobrados em provas de concurso.

APT: resolve dependências para sistemas baseados em Debian (incluindo o Ubuntu). Possui uma sintaxe muito simples. Vamos a alguns exemplos para:

- atualizar os repositórios de software: `sudo apt-get update` ou `sudo apt update` (a lista com os repositórios devem estar no arquivo “/etc/apt/sources.list”);
- atualizar seus pacotes já instalados (softwares): `sudo apt-get upgrade` ou `sudo apt upgrade`;
- uma atualização mais completa (tenta atualizar pacotes para a versão mais recente e remover dependências mais antigas ou não utilizadas): `sudo apt-get dist-upgrade` ou `sudo apt full-upgrade`;
- atualização dos repositórios e atualizar também os pacotes já instalados: `sudo apt-get update && sudo apt-get upgrade` ou `sudo apt update && sudo apt upgrade`;
- instalar um software: `sudo apt-get install nome_do_pacote` ou `sudo apt install nome_do_pacote`;
- remover um pacote: `sudo apt-get remove nome_do_pacote` ou `sudo apt remove nome_do_pacote`.

Ao remover um software do sistema usando o comando `apt-get remove`, o APT realiza todo o trabalho de remoção de dependências não utilizadas. Algumas dependências podem permanecer no sistema, então é só utilizar a opção `autoremove`: `sudo apt-get autoremove` ou `sudo apt autoremove`.

Para procurar um pacote a ser instalado, basta usar: `sudo apt-cache search termo_pesquisa` ou `sudo apt search termo_pesquisa`.

O APT atualmente não oferece a possibilidade de instalar um pacote de um *site*, ou seja, o usuário deve encontrar e baixar o pacote a ser instalado por conta própria, utilizando nesse caso o DPKG.

DPKG: criado para instalar pacotes “.deb” na distribuição Debian. Também é utilizado por outras distribuições, como Ubuntu, KNOPPIX etc. Os principais comandos utilizados pelo gerenciador de pacotes DPKG são `dpkg` e `dpkg-reconfigure`. A sintaxe é: `dpkg [opções] nome_pacote`

Opções:

- i : instalação simples.
- r : desinstala o pacote - exceto arquivos de configuração do pacote.
- P : desinstala o pacote - todos os arquivos do pacote.
- l : exibe os pacotes que estão instalados.
- p : exibe informações sobre o pacote instalado.
- s : exibe o status do pacote.
- l : exibe informações sobre pacotes não instalados.
- S : exibe o pacote do qual o arquivo faz parte.



- L : exhibe os arquivos que fazem parte de um pacote instalado.
- c : exhibe os arquivos que fazem parte de um pacote não instalado.
- help : exhibe uma mensagem de ajuda.

Alguns exemplos:

- Instalação de um pacote: # dpkg -i pacote.deb
- Desinstalação de um pacote, mantendo os seus arquivos de configuração: # dpkg -r pacote
- Desinstalação de um pacote, inclusive os seus arquivos de configuração: # dpkg -P pacote
- Exibição de informações do pacote não instalado: # dpkg -l pacote.deb

Em relação ao comando dpkg-reconfigure: reconfigura pacotes “.deb” após terem sido instalados utilizando o debconf (sistema de configuração de pacotes “.deb”). Esse comando fará perguntas para reconfigurar o pacote.

Sintaxe: dpkg-reconfigure opções nome_pacote

Opções:

- a ou --all :: reconfigura todos os pacotes
- h ou --help :: exhibe ajuda

Exemplo: Reconfigura o pacote ssh → # dpkg-reconfigure ssh

RPM: criado pela Red Hat para instalar pacotes “.rpm”, sendo utilizado por várias distribuições, como Red Hat Enterprise/Fedora Core, Mandriva, SuSe, entre outras. O principal comando do gerenciador de pacotes RPM, é o rpm.

Sintaxe: rpm [opções] nome_pacote

Algumas opções:

- i: instalação simples.
- v: exhibe detalhes da instalação.
- h: mostra o caractere "#", enquanto o programa é instalado.
- U: atualização do programa de uma versão anterior para uma mais recente.
- nodeps: não procura dependências.
- force: força a instalação.
- root diretório: utiliza o sistema com raiz em diretório para todas as operações.
- e: desinstala o pacote.



--import: importa a chave pública GPG do distribuidor de um pacote.

--help: exibe uma mensagem de ajuda.

-q: consulta (*query*) se um pacote já está instalado.

Exemplos:

#rpm -q httpd → Consulta se o pacote está instalado.

rpm -ivh tree.i386.rpm → Instala o pacote, exibindo detalhes da instalação e mostra o caractere “#”, enquanto o programa é instalado.

rpm -e tree → Desinstala o pacote tree.

YUM: resolve dependências de pacotes para distribuições que utilizam o RPM. É um gerenciador de pacotes padrão incluído em algumas distribuições baseados no REDHAT, incluindo o Fedora e CentOS. Muito parecido com o APT, vejamos alguns comandos:

- Atualização dos repositórios do YUM: `sudo yum update`;
- Instalação de um pacote: `sudo yum install nome_do_pacote`;
- Remoção de um pacote: `sudo yum remove nome_do_pacote`;
- Busca de um pacote que será instalado: `sudo yum search nome_do_pacote`.

O YUM não inclui um comando autoremove para encontrar e remover dependências não utilizadas, mas incluí um recurso para a instalação de um pacote a partir de uma URL (função que não existe no APT):

```
sudo yum install $url
```

CONFIGURAÇÃO DE SERVIÇOS OU FUNCIONALIDADES

Diversos serviços ou funcionalidades no Linux são configurados através de arquivos texto e alguns deles costumam ser cobrados em provas de concurso. Vamos a eles...

Segurança da senha do usuário: os arquivos que armazenam os dados dos usuários do Linux são o “/etc/passwd” e o “/etc/shadow”. Cuidado! O passwd não armazena a senha, embora o nome sugira isso! Vamos ver o exemplo de uma linha de cada arquivo, relativo ao usuário “estrategia”. Primeiro o arquivo passwd:

```
estrategia:x:1010:1010:Estrategia Concursos:/home/estrategia:/bin/bash
```

Separado por dois pontos, podemos identificar os seguintes campos:

- nome do usuário = “estrategia”;
- “senha” do usuário (o “x” indica que a senha está armazenada no arquivo /etc/shadow);
- identificação do usuário (uid = 1010);
- identificação do grupo (gid = 1010);
- comentário sobre o usuário (geralmente o nome completo) = “Estrategia Concursos”;



- diretório principal (home) do usuário = “/home/estrategia”;
- *shell* utilizado, ao logar, pelo usuário = “/bin/bash”.

Agora vamos ao arquivo shadow:

```
estrategia:$6$AkC...CdbF:17355:0:99999:7:::
```

Separado por dois pontos, podemos identificar os seguintes campos:

- nome do usuário = “estrategia”;
- *hash* da senha do usuário = “\$6\$AkC...CdbF” (obs.: inserir os três pontos no meio porque esse campo é muito grande);
- última vez (em dias) que a senha foi modificada, desde 1 de janeiro de 1970 = 17355;
- depois de alterada a senha o usuário só poderá alterar novamente depois de x dias = 0 (0 significa que está desabilitado);
- a partir do momento que o usuário altera a senha, é obrigado a alterá-la novamente depois de x dias = 99999;
- quantos dias antes do fim do prazo para alterar a senha, deve ser mostrado um alerta = 7.

Os outros dois campos estão vazios, mas significam: inativo (quanto tempo sua conta ficou inativa/desabilitada), expirada (quantos dias depois de 01/01/1970 que sua conta está desabilitada ou expirada).

Configuração de interfaces de rede: os programas *ifup* (habilita interfaces de rede) e *ifdown* (desabilita) se baseiam na configuração do arquivo “/etc/network/interfaces”, o qual vamos ver um exemplo abaixo, com comentários.

```
auto lo                # configuração automática para lo
iface lo inet loopback # lo é uma interface de loopback
auto eth0              # configuração automática para eth0
allow-hotplug eth0    # ativa a interface eth0 (com o ifup)
iface eth0 inet static address 192.168.100.1 netmask 255.255.255.0
network 192.168.100.0 broadcast 192.168.100.255
# a última linha faz com que ifup ative a eth0 com o endereço fixo
192.168.100.1/24
```

Para os próximos dois serviços, é muito difícil cobrar algo muito específico, basicamente as bancas cobram os nomes de arquivos, então vamos direto ao ponto.

DNS (Domain Name System): o serviço de resolução de nomes tem um arquivo de configuração, o “/etc/resolv.conf”. Lembre que “*resolv*” está ligado a “resolver um nome”, fica mais fácil de lembrar.

NFS (Network File System): o sistema de arquivos em rede utilizado pelo Linux utiliza um arquivo que controla quais diretórios o servidor NFS exporta, o “/etc/exports”.



INTEROPERABILIDADE

Quando falamos em sistemas de arquivos FAT, NTFS, EXT, XFS, entre outros, estamos falando em sistemas de arquivos “locais”, os que permitem o acesso a uma mídia interna ou conectada a um computador, por exemplo. Existem também os sistemas de arquivos em rede, que dão a ilusão de um acesso local, porém o acesso é remoto. Os mais conhecidos são o NFS e o SMB/CIFS, os quais veremos a seguir.

O **NFS (Network File System)** é um protocolo de sistema de arquivos distribuído, **originado no Unix**, que permite que um usuário em um computador cliente acesse arquivos através de uma rede como se estivesse acessando na máquina local. O cliente NFS tem a finalidade de tornar o acesso remoto transparente para o usuário do computador, e esta interface cliente e servidor, executada pelo NFS através dos protocolos Cliente-Servidor, fica bem definida quando o usuário, ao chamar um arquivo/diretório no servidor, lhe parece estar acessando localmente, sendo que está trabalhando com arquivos remotamente.

Uma máquina Linux pode ser um servidor NFS e um cliente NFS, o que significa que pode exportar sistemas de arquivo para outros sistemas e montar sistemas de arquivo exportados de outras máquinas. Por exemplo, um grupo de usuários pode ter acesso aos arquivos de um determinado projeto usando um diretório compartilhado do sistema de arquivo NFS montado no diretório “/projeto”. Para acessar os arquivos compartilhados, o usuário vai até o diretório “/projeto”, sem senhas ou comandos especiais para lembrar. Dessa forma, os usuários trabalham como se o diretório estivesse em suas máquinas locais.

Algumas **características do NFS** são:

- Arquitetura cliente-servidor: o servidor recebe as requisições vindas do cliente, verifica a validade, executa e retorna ao cliente;
- *Stateless* (sem estado): considera cada requisição como uma transação independente que não está relacionada a qualquer requisição anterior, de forma que a comunicação consista de pares requisição/resposta independentes. Diante de uma falha no sistema, quando o servidor NFS volta a funcionar corretamente, o estado anterior é restaurado e o funcionamento continua como se não houvesse ocorrido nenhum problema;
- *Caching*: permite que as informações utilizadas mais recentemente sejam alocadas para posterior uso, ou pode carregar os dados em antecipação a operações futuras;
- *File Locking* (bloqueio de arquivo): permite que um processo tenha acesso exclusivo a um arquivo ou parte deste, e força outros processos que estão solicitando acesso ao mesmo arquivo, aguardarem a liberação;
- Utiliza o protocolo RPC (*Remote Procedure Call*): define um modo independente do sistema para processos se comunicarem através de uma rede de computadores, sendo que o cliente faz a chamada do procedimento remoto no servidor;

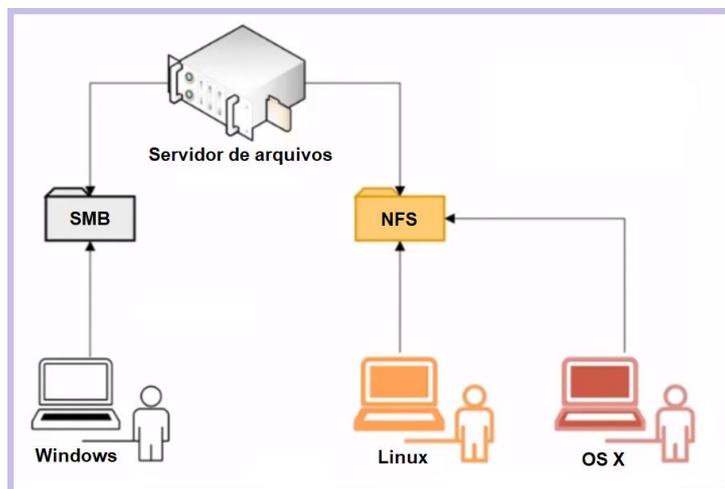


- Protocolo de transporte padrão é o UDP, no entanto, é possível utilizar o TCP;
- Compartilhar arquivos de um servidor NFS é conhecido como exportar os diretórios.

Para que os clientes possam acessar o servidor NFS é necessário que os seguintes *daemons* estejam em execução:

- *nfsd*: *daemon* NFS, atende requisições dos clientes NFS;
- *mountd*: *daemon* de montagem NFS, executa as solicitações passadas pelo *nfsd*;
- *portmap*: *daemon portmapper*, permite que clientes NFS descubram qual porta o servidor NFS está utilizando.

Na figura abaixo podemos ver o acesso a um servidor de arquivos através da rede. Note que máquinas Unix-like (ex.: Linux e OS X) utilizam o NFS como protocolo e as máquinas Windows utilizam o SMB/CIFS, que veremos o funcionamento na sequência.



Antes de começar a falar do protocolo em si, deixo bem claro que as questões de concurso já cobraram alguns detalhes do NFS, mas em relação ao SMB/CIFS a cobrança é superficial. Vamos lá...

O **SMB/CIFS (Server Message Block/Common Internet File System)** é um protocolo de redes cujo uso mais comum como é o compartilhamento de arquivos em uma rede local. Ele permite que o cliente manipule arquivos como se estes estivessem em sua máquina local (mesma coisa que o NFS, heim?). Algumas das operações suportadas são: leitura, escrita, criação, exclusão e renomeação, sendo que os arquivos/diretórios manipulados estão em um servidor remoto.

O protocolo SMB/CIFS funciona através do envio de pacotes do cliente para o servidor. O servidor recebe o pacote, verifica se a requisição é válida (se o cliente possui as permissões apropriadas), executa a requisição e retorna um pacote de resposta ao cliente. O cliente verifica o pacote de resposta para determinar se a requisição inicial obteve êxito. O protocolo mais utilizado para transporte confiável é o NetBIOS sobre TCP (NBT). Outros protocolos foram utilizados na camada de transporte, mas o NBT se tornou o mais utilizado.

O SMB/CIFS é muito utilizado pelos sistemas operacionais Windows, podendo funcionar como um servidor ou cliente. Isso não quer dizer que sistemas Unix-like não possuam também esse protocolo! Na verdade, a maioria dos sistemas Unix-Like (**Linux**, por exemplo) possuem uma implementação de

cliente/servidor do SMB/CIFS via **Samba**, que é um “software servidor” para Linux que permite o gerenciamento e compartilhamento de recursos em redes formadas por computadores com o Windows. Dessa forma é possível utilizar o Linux como servidor de arquivos, servidor de impressão, entre outros, como se a rede utilizasse servidores Windows (Server 200x, 201x, entre outros).



Resumindo: o Samba é um componente que realiza a comunicação entre servidores Linux e Windows, permitindo que eles compartilhem recursos de disco e de impressão. O *daemon* `smbd` possui dois modos de compartilhamento. O primeiro modo (*sharemode*) é mais simples, gerando uma senha para cada recurso que é compartilhado. O segundo modo (*user*) permite o compartilhamento de vários recursos com um único *login* e senha. Os privilégios de acesso aos recursos devem ser sempre mediados pelos administradores do sistema.

Um outro componente interessante do pacote Samba é o *nmbd*, um servidor de nomes NetBios (mais conhecido por **WINS - Windows Internet Name Server**) responsável por entender e responder solicitações de resoluções de nomes NetBios sobre IP. É ele o responsável pelo “aparecimento” do ícone do servidor Samba no ambiente de rede do Windows! Obs.: Se você não sabe o que é essa resolução de nomes, pense que é algo como o DNS, mas não de forma hierárquica, é uma resolução feita pelo Windows (sabendo isso já está bom para a prova 😊).

SERVIÇO DE DIRETÓRIO

Um serviço de diretório é um sistema de software que armazena, organiza e fornece acesso a informações em um diretório do sistema operacional. Diretórios podem ser muito limitados em escopo, suportando apenas um pequeno conjunto de tipos de nós e tipos de dados, ou eles podem ser muito amplos, suportando um conjunto de tipos arbitrário ou extensível. No DNS os nós são nomes de domínio e os itens de dados são endereços IP (e apelidos, nomes de servidor de e-mail etc.).

Em um diretório utilizado por um sistema operacional de rede, os nós representam **recursos** que são gerenciados pelo sistema operacional, incluindo usuários, computadores, impressoras, entre outros recursos compartilhados. Muitos serviços de diretório têm sido utilizados desde o aparecimento da Internet, mas o mais utilizado é um descendente do serviço de diretório X.500.

Antes de vermos uma implementação de serviço de diretório para o Linux (OpenLDAP), é importante entendermos o protocolo que fornece mecanismos de acesso aos objetos, o **LDAP (Lightweight**



Directory Access Protocol). Entidades internacionais (ITU, ISO, IETF, entre outras) trabalham na definição de padrões diversos, incluindo a padronização que dá suporte a serviços de diretórios. Um padrão de uso genérico é o **X.500** (da ISO) que possui uma grande abrangência, mas é muito complexo e não foi adotado em sua íntegra como um padrão de mercado. Um padrão mais “light” que de fato se tornou um padrão de mercado foi o LDAP.

O padrão LDAP define um sistema de nomeação hierárquico, no qual é possível referenciar qualquer objeto que esteja no AD. Um nome LDAP é composto pelo caminho completo do objeto (ex.: uma impressora, um computador etc.), partindo do domínio raiz até chegar ao objeto em si. Algumas abreviaturas (**atributos**) são utilizadas nessa nomenclatura hierárquica:

- cn: *common name* (nome da conta de um usuário, grupo etc.);
- sn: sobrenome (*surname*);
- ou: faz referência a uma unidade organizacional;
- dc: componente de domínio (normalmente o nome do domínio);
- o: nome da organização (geralmente o domínio raiz);
- c: *country* - país (normalmente não utilizado).

Vamos a um exemplo de um nome LDAP:

CN=evandrodv, OU=professores, DC=ti, DC=estrategiaconcursos.com.br → esse nome representa o usuário “evandrodv”, cuja conta está contida na unidade organizacional “professores”, no domínio “ti.estrategiaconcursos.com.br”. Obs.: os dois componentes de domínio foram concatenados.

Por padrão um servidor LDAP “**escuta**” na porta 389 (TCP) e as principais características do protocolo são:

- Baseado em padrão aberto: qualquer desenvolvedor pode acessar sua especificação e realizar a implementação;
- Possui APIs bem definidas: facilita a vida dos programadores;
- Maior velocidade de consulta que um BD relacional;
- Replicável e distribuível;
- Facilita a localização de informações e recursos: pesquisa feita nome.

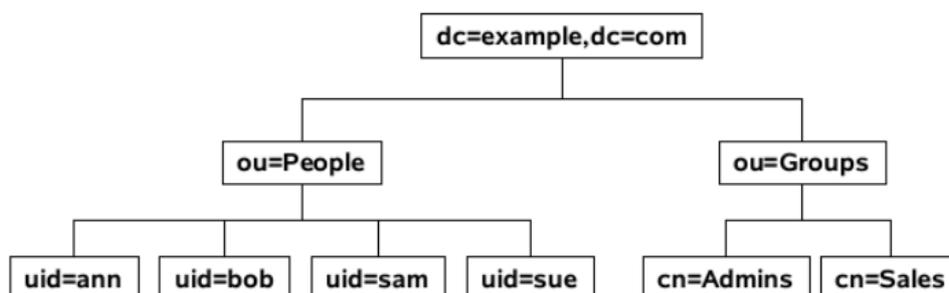
Algumas **operações (comandos)** que podem ser utilizados através do LDAP são:

- Bind: autentica e especifica a versão do protocolo LDAP;
- Search: procura/recupera entradas dos diretórios;
- Compare: compara uma entrada com determinado valor;
- Add: adiciona uma nova entrada;
- Delete: exclui uma entrada;
- Modify: modifica uma entrada;
- Modify DN: move ou renomeia uma entrada;
- Abandon: aborta uma requisição prévia;



- Unbind: fecha a conexão;
- Extended Operation: operação genérica para definir outras operações;
- StartTLS: protege a conexão com o TLS - implementada a partir da versão 3 do LDAP.

A representação dos dados é realizada através de uma estrutura hierárquica na forma de árvore (*Directory Information Tree - DIT*), a qual consiste em entradas de DN's (*Distinguished Names*). O LDAP utiliza a DIT como estrutura de dados fundamental:



Como podemos ver, a árvore de diretório possui uma forma hierárquica:

- Primeiro o diretório raiz;
- Após a rede corporativa, os departamentos e por fim os computadores dos usuários e os recursos de rede.

Alguns conceitos que também já foram cobrados em concursos são mostrados na sequência.

Schema: conjunto de objetos e atributos para o armazenamento. É modelado de acordo com o padrão X.500 da ISO.

Cada entrada (objeto) possui um identificador único (*dn - distinguished name*), o qual consiste de seu Relative Distinguished Name (RDN), construído de algum(ns) atributo(s) na entrada, seguido pelo DN da entrada pai.

Escalabilidade: é possível replicar servidores LDAP e incluir novos servidores à medida que aumenta a estrutura da organização. Ou seja, não é uma estrutura “engessada”.

Como já mencionado anteriormente, uma solução bastante conhecida para Linux é o **OpenLDAP** (que também possui versões para Windows, Solaris, entre outros sistemas operacionais). Vamos ver como instalar o OpenLDAP:

```
sudo apt install slapd ldap-utils
```

O principal arquivo de configuração (pelo menos para concursos) é o “*/etc/ldap.conf*”, que é usado para configurar os padrões a serem aplicados quando da execução dos clientes LDAP.



QUESTÕES COMENTADAS

1. (2016 - AOCP - EBSERH)

O NFS (Network File System) permite compartilhar sistemas de arquivos entre computadores conectados em rede e pode ser parte fundamental da infraestrutura da tecnologia da informação. Sobre NFS, analise as assertivas a seguir e assinale a alternativa que aponta a(s) correta(s).

I. O NFS é considerado sem estado (stateless) e, portanto, quando um servidor NFS volta a funcionar, o estado anterior é restaurado e a informação não é perdida.

II. O NFS pode ser implementado do lado servidor e do lado cliente.

III. O NFS roda sobre o protocolo RPC (Remote Procedure Call), que define um modo independente do sistema para processos se comunicarem através de uma rede de computadores.

IV. O NFS suporta apenas UDP como protocolo de transporte, pois ele apresenta desempenho significativamente melhor que o TCP em redes locais.

A) Apenas I e II.

B) Apenas II e IV.

C) Apenas III.

D) Apenas III e I.

E) Apenas I, II e III.

Comentários:

I. Exato! O NFS é *stateless*, cada requisição é uma transação independente que não está relacionada a qualquer requisição anterior. Então, quando um servidor NFS volta a funcionar, o estado anterior é restaurado e a informação não é perdida. (II) Existe quem faz requisições de acesso e manipulação de arquivos e diretórios (cliente) e quem recebe as requisições, avalia as permissões, executa e retorna (servidor). (III) O NFS utiliza o RPC (*Remote Procedure Call*), o cliente faz a chamada ao servidor remoto (o servidor até pode estar na mesma máquina, mas o cenário mais comum é remoto). (IV) O protocolo padrão de transporte é o UDP, mas é possível optar pelo TCP.

Gabarito: E

2. (2017 - FGV - IBGE)

Em ambiente Linux, a chamada ao sistema (system call) que implementa a criação de um novo processo é:

A) `create_process`;

B) `new_process`;



- C) fork;
- D) spawn;
- E) duplicate.

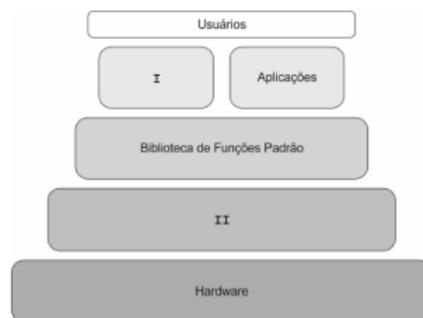
Comentários:

No Linux a chamada de sistema fork() faz uma “bifurcação” de um processo, sendo que o novo vira o filho do outro. Ex.: em um shell você executa algum comando, então o fork() será chamado, o shell será o processo pai do processo novo (o comando digitado).

Gabarito: C

3. (2017 - FCC - TRF-5ª Região)

Considere a figura abaixo que mostra a arquitetura do sistema operacional Linux



A caixa

- A) I representa a camada responsável pela interface entre o hardware e as aplicações. Dentre suas funções encontram-se gerenciamento de I/O, manutenção do sistema de arquivos, gerenciamento de memória e swapping, controle da fila de processos, etc.
- B) II representa a camada que permite o acesso a recursos através da execução de chamadas feitas por processos. Tais chamadas são geradas por funções padrão suportadas pelo kernel. Dentre suas funções estão habilitar funções padrão como open, read, write e close e manter a comunicação entre as aplicações e o kernel.
- C) I é um processo que executa funções de leitura de comandos de entrada de um terminal, interpreta-os e gera novos processos, sempre que requisitados. É conhecido também como interpretador de comandos.
- D) II é um processo que realiza modificações no shell, permitindo que funcionalidades do Linux sejam habilitadas ou desabilitadas, conforme a necessidade. Tal processo gera ganho de performance, pois à medida que customiza o shell, o usuário torna o Linux enxuto e adaptável.
- E) I é um processo que realiza modificações no kernel, permitindo que funcionalidades do Linux sejam habilitadas ou desabilitadas, conforme a necessidade. Tal processo gera ganho de performance, pois à medida que customiza o kernel, o usuário torna o Linux enxuto e adaptável.



Comentários:

A caixa I pode ser um *shell* que recebe comandos do teclado, interpreta e executa, gerando novos processos (filhos) através da chamada de sistema `fork()`. Ex.: execução do comando “ls” no *shell*.

Gabarito: C

4. (2017 - CS-UFG - DEMA E)

Na maior parte das distribuições Linux, a memória virtual é uma partição denominada

- A) EXT
- B) SWAP
- C) SETUP
- D) BIOS

Comentários:

Vimos que pode ser partição ou arquivo (ou até ambos), mas geralmente se utiliza uma partição de SWAP.

Gabarito: B

5. (2017 - Quadrix - COFECI)

Com relação à administração de sistemas Windows e Unix/Linux, julgue o item subsequente.

Não é possível realizar uma instalação automática, por meio de uma rede, no sistema operacional Linux.

Comentários:

Pode-se realizar uma instalação através de uma mídia (CD, DVD, pen drive) ou através da rede, desde que a placa de rede permita e a configuração esteja adequada para um *boot* pela rede.

Gabarito: Errado

6. (2017 - Colégio Pedro II - Colégio Pedro II)

Sistema operacional é um conjunto de programas básicos e utilitários que fazem seu computador funcionar. O Debian é um sistema operacional, em cujo núcleo está o Kernel, o programa mais fundamental no computador e que faz todas as operações mais básicas, permitindo que você execute os outros programas. O Debian atualmente usa o Kernel Linux. Mas um sistema operacional não funciona somente com o Kernel, são necessários utilitários e aplicativos. O Debian utiliza estas ferramentas do projeto GNU. Por esse motivo, muitos utilizadores defendem que devemos chamar o sistema de “Debian GNU/Linux”. Reconheça (1) o gerenciador de pacotes (programas) usado no Debian e em distribuições derivadas do Debian (Ubuntu, Knoppix, Big Linux etc.) que utiliza uma lista



de dependências para instalar tudo o mais automaticamente possível, e (2) o arquivo de configuração utilizado por este gerenciador de pacote.

A) Gerenciador (1) - apt-get

Arquivo (2) - source.list

B) Gerenciador (1) - apt-get

Arquivo (2) - package.cfg

C) Gerenciador (1) - make install

Arquivo (2) - package.cfg

D) Gerenciador (1) - dpkg

Arquivo (2) - source.list

Comentários:

Das alternativas mostradas, apenas “apt-get” é um gerenciador que utiliza uma lista de dependências e o nome dela é “sources.list”. Nas alternativas aparece com um “s” a menos, mas tudo bem, relaxe...

Gabarito: A

7. (2018 - CESPE - EMAP)

O kernel do sistema operacional Linux tem a função de interpretar os comandos executados em um terminal.

Comentários:

Quem tem essa função é o *shell*! O *kernel* é o núcleo do sistema operacional!

Gabarito: Errado

8. (2018 - FGV - MPE-AL)

Instalar, atualizar e remover pacotes no sistema operacional CentOS 7 é uma tarefa frequente para desenvolvedores de sistemas. Por isso, eventualmente podem ocorrer dúvidas sobre se determinado pacote está instalado ou qual é a versão que está sendo utilizada. Para dirimir essas dúvidas, sobre o pacote `httpd` devemos utilizar o comando

A) `rpm -q httpd`

B) `yum check httpd`

C) `apt-get find httpd`

D) `find -iname httpd`

E) `crontab -l httpd`



Comentários:

Para quem não lembra da parte da aula sobre o gerenciador utilizado no CentOS, pelo menos deve lembrar que a resposta só pode ser uma das 3 primeiras, certo? Senão, a coisa tá feia...

Se também lembrar que “-q” é em relação a “query” (pesquisa), aí fica mais tranquilo para marcar a alternativa A.

Gabarito: A

9. (2018 - FGV - COMPESA)

Com relação às características e tarefas de administração do sistema operacional Linux, analise as afirmativas a seguir.

I. O esquema de partição GPT (Guid Partition Table) oferece a possibilidade de até 128 partições primárias, com tamanhos maiores do que 2TB cada.

II. Ext4, XFS e exFAT são exemplos de sistemas de arquivo com jornal, utilizados pelo Linux.

III. A área de swap no Linux pode ser uma partição dedicada de swap, um arquivo de swap ou uma combinação de arquivos e partições de swap.

Está correto o que se afirma em

- A) I, somente.
- B) II, somente.
- C) III, somente.
- D) I e III, somente.
- E) I, II e III.

Comentários:

(I) Ao contrário da MBR, que oferece 4, a GPT realmente permite até 128, está correta! (II) exFAT é um sistema de arquivos do Windows, não do Linux! (III) Embora muitos pensem que tem que ser uma partição, pode ser um arquivo de swap ou ainda uma combinação.

Gabarito: D

10.(2018 - CS-UFG - SANEAGO-GO)

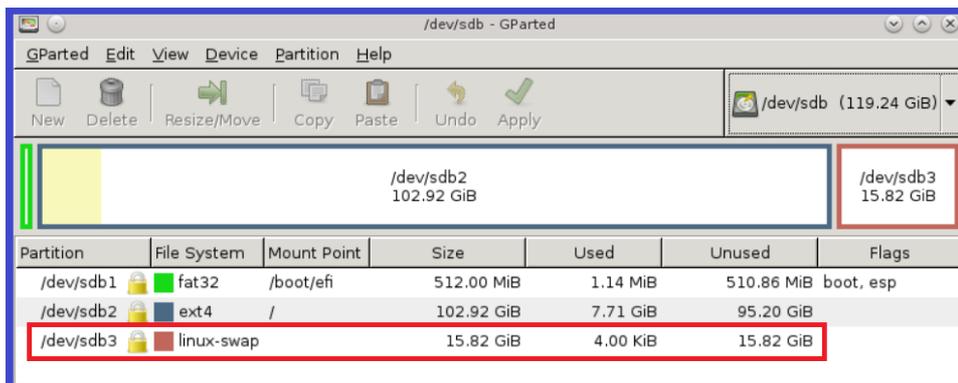
Um utilitário GNU, presente em diversos sistemas operacionais linux, que lista as partições dos discos rígidos, é:

- A) fdisk.
- B) mcdisk.
- C) parted.
- D) gedit.



Comentários:

Na verdade é GParted, mas tudo bem...releve uma letrinha 😊



Gabarito: C

11.(2018 - IF-RS - IF-RS)

São tipos de sistemas de arquivos utilizados no Linux:

- A) FAT 16, FAT 32, NTFS
- B) FAT 32, HFS, ExFAT
- C) Ext2, Ext3, Btrfs
- D) Ext2, Ext3, HFS
- E) Ext2, Ext3, NFS, SMB

Comentários:

Família FAT e NTFS= Windows.

HFS = Apple.

NFS e SMB = Sistemas de arquivos de rede.

Só sobrou a letra C! Lembrando: família EXT, Btrfs, XFS, JFS, ReiserFS, entre outros = Linux!

Gabarito: C

12.(2018 - COPESE-UFT - UFT)

Embora seja possível realizar boot de um sistema Linux a partir de um pendrive, a maioria das instalações do Linux o realiza a partir do disco rígido do computador. Esse processo consiste em duas fases básicas:

1. Executar o carregador de boot a partir do dispositivo de boot;
2. Iniciar o kernel do Linux e iniciar os processos.

Assinale a alternativa que contém um gerenciador de boot para sistemas Linux:



- A) CISC (Complex Instruction Set Computer).
- B) GRUB (Grand Unified Bootloader).
- C) ORM (Object Relational Mapping).
- D) CMS (Content Management System).

Comentários:

O LILO já foi bastante utilizado (mas nem aparece como opção) e o GRUB é mais utilizado ultimamente.

Gabarito: B

13.(2018 - COPESE-UFT - UFT)

Sistemas operacionais Linux permitem logins simultâneos de diferentes usuários. Assinale a alternativa que contém o comando a ser utilizado para visualizar os usuários logados em um determinado instante.

- A) listusers
- B) Idconfig
- C) who
- D) uname

Comentários:

O Linux é um sistema operacional multiusuário, ou seja, pode ter N usuários logados ao mesmo tempo. Para saber QUEM está logado pode ser utilizado o comando "who".

Gabarito: C

14.(2018 - IF-RS - IF-RS)

Considerando que se está logado como "root" no terminal de uma estação de trabalho com um sistema operacional Linux debian ou derivado, qual comando configura o DNS para 8.8.8.8?

- A) dns 8.8.8.8
- B) echo 8.8.8.8 > /etc/hosts.conf
- C) echo nameserver 8.8.8.8 > /etc/resolv.conf
- D) nslookup 8.8.8.8
- E) nameserver 8.8.8.8

Comentários:

Lembrando...o DNS resolve nomes, então o arquivo de configuração é o "resolv.conf" e sabemos que os arquivos de configuração por padrão ficam em "/etc". Só tem uma opção...a letra C.



Mas, continuando...o comando echo “escreve”, então o comando completo “escreveu” “nameserver 8.8.8.8” no arquivo “/etc/resolv.conf”, ou seja, configurou o serviço de DNS.

Gabarito: C

15.(2018 - COPEVE-UFAL - UFAL)

O assistente de tecnologia da informação precisa atualizar o sistema operacional GNU/Linux do servidor WEB de uma empresa. Qual comando, usando o modo root, deve ser utilizado?

- A) upgrade distro-all.
- B) upgrade apt-distro.
- C) apt-get dist-upgrade.
- D) apt-get purge “nome da distro”.
- E) apt-cache showpkg “nome da distro”.

Comentários:

Sabemos que o gerenciador de pacotes é o APT, já descartamos as duas primeiras. Mesmo sem lembrar da aula, dá para ver que pela lógica a letra C indica um upgrade da distribuição Linux.

Gabarito: C

16.(2018 - COPEVE-UFAL - UFAL)

Dadas as afirmativas quanto aos conceitos de Kernel no sistema operacional GNU/Linux,

- I. Compilar o Kernel permite ao usuário remover drivers inúteis diminuindo o tempo de arranque do sistema operacional.
- II. Os módulos do Kernel do sistema estão armazenados no diretório /boot/.
- III. O comando uname -a exibe as informações do kernel do sistema.
- IV. Módulos são as partes do kernel que são carregadas somente quando são requisitadas por um aplicativo ou dispositivo.

Verifica-se que está(ão) correta(s)

- A) II, apenas.
- B) I e II, apenas.
- C) III e IV, apenas.
- D) I, III e IV, apenas.
- E) I, II, III e IV.

Comentários:



(I) Compilar o *kernel* permite eliminar algo inútil ou acrescentar algo útil (ex.: um driver de uma placa de rede que você possui e não tinha antes). (II) Não! Ficam em “/lib/modules/versão_do_kernel”. (III) Exato! Conforme tela abaixo, com destaque na versão do kernel. (IV) Perfeito! Assim não carrega o que não precisa para a memória!

```
[root@localhost lib]# uname -a  
Linux localhost 4.12.0-rc6-g48ec1f0-dirty #21 Fri Aug 4 21:02:28 CEST 2017 i586  
GNU/Linux  
[root@localhost lib]#
```

Gabarito: D

17.(2018 - FGV - AL-RO)

Wallace é administrador de um servidor com sistema operacional CentOS e deseja compartilhar um diretório com os integrantes da empresa por meio do protocolo NFS.

Para que a configuração desse compartilhamento possa ser efetuada, Wallace deve editar o arquivo /etc/

- A) services
- B) modules
- C) hosts
- D) export
- E) fstab

Comentários:

O Linux geralmente é intuitivo, então digamos que você tenha esquecido da aula...mas lembra que o NFS é um sistema de arquivos de rede e os diretórios a serem “exportados” devem estar listados no arquivo “export”.

Gabarito: D

18.(2018 - COPEVE-UFAL - UFAL)

Analise a configuração de rede no GNU/Linux.

```
auto lo  
iface lo inet loopback  
auto eth0  
allow-hotplug eth0  
iface eth0 inet static address 192.168.100.1 netmask 255.255.255.0  
network 192.168.100.0 broadcast 192.168.100.255
```

Dadas as afirmativas quanto à configuração de rede,



- I. O “iface eth0 inet static” informa ao sistema que a placa de rede terá o endereço IP estático.
- II. O “iface lo” informa ao sistema que a placa de rede receberá um endereço IP via servidor DHCP.
- III. Uma segunda placa de rede com fio instalada nesse computador irá receber a denominação eth1.
- IV. O “allow-hotplug” permite conectar dispositivos hotplug no computador.

verifica-se que está(ão) correta(s)

- A) II, apenas.
- B) I e III, apenas.
- C) II e IV, apenas.
- D) I, III e IV, apenas.
- E) I, II, III e IV.

Comentários:

(I) O “iface eth0 inet static” informa ao sistema que a placa de rede terá o endereço IP estático, ou seja, não receberá um endereço através do protocolo DHCP. (II) O “iface lo” indica a interface de loopback (aquela para testes, geralmente o endereço IP é 127.0.0.1). (III) Uma segunda placa de rede com fio instalada nesse computador irá receber a denominação eth1, pois eth indica “ethernet” e já existe a eth0. (IV) O “allow-hotplug” permite conectar dispositivos hotplug no computador (“allow” = permite, então ficou tranquila essa).

Gabarito: D

19.(2018 - FGV - AL-RO)

Roberto trabalha como administrador de redes em uma empresa de cosméticos e sua principal função é a configuração de estações de trabalho dos seus colaboradores. Rotineiramente, Roberto instala e remove pacotes nas estações de trabalho às quais ele dá suporte.

Assinale a opção que indica o comando utilizado por Roberto para desinstalar pacotes RPM nas estações de trabalho com sistema operacional Linux CentOS.

- A) rpm -Uvh pacote
- B) rpm -e pacote
- C) rpm -ivh pacote
- D) rpm -qi pacote
- E) rpm -qa | grep pacote

Comentários:

Algumas opções do rpm:

-i: instalação simples.



- v: exibe detalhes da instalação.
- h: mostra o caractere "#", enquanto o programa é instalado.
- U: atualização do programa de uma versão anterior para uma mais recente.
- nodeps: não procura dependências.
- force: força a instalação.
- root diretório: utiliza o sistema com raiz em diretório para todas as operações.
- e: desinstala o pacote.
- import: importa a chave pública GPG do distribuidor de um pacote.
- help: exibe uma mensagem de ajuda.
- q: consulta (*query*) se um pacote já está instalado.

Gabarito: B

20.(2018 - Quadrix - CFBio)

Carregamento e inicialização do kernel e execução dos scripts de inicialização do sistema são algumas das etapas do processo de inicialização de um Linux típico.

Comentários:

Questão redonda, pois o núcleo do S.O. (Linux) deve ser carregado, depois os scripts de inicialização definidos pelo usuário (ou om a configuração padrão mesmo).

Gabarito: Certo

21.(2018 - Quadrix - CRM-PR)

Um computador pessoal com sistema operacional Linux, versão desktop, pode ter no máximo cinco usuários simultâneos.

Comentários:

Só teria o limite de 5 se fosse configurado previamente, mas por padrão não tem limite - até estourar a memória! 😊

Gabarito: Errado

22.(2018 - FADESP - IF-PA)

Sobre os pacotes do Sistema Operacional (SO) Debian 9.5, é correto afirmar que

A) o apt é um programa que pode substituir o dpkg para o gerenciamento de pacotes do SO.



- B) os pacotes devem ser distribuídos no formato .tar.gz.
- C) o dpkg-source compila e executa código fonte de um pacote.
- D) um pacote marcado como Essential não pode ser desinstalado do Sistema Operacional.
- E) cada pacote deve especificar as dependências sobre outros pacotes para seu funcionamento.

Comentários:

É comum que para uma aplicação funcione ela precise de alguns pacotes, porque alguns dependem de outros. Os gerenciadores de pacotes ajudam a verificar essas dependências.

Gabarito: E

23.(2018 - CONSULPAM - Câmara de Juiz de Fora-MG)

Em relação ao Sistema Operacional Linux, marque o item INCORRETO:

- A) Sua arquitetura é composta por um núcleo monolítico cujas funções são: gerenciar a memória, operar as entradas e saídas e o acesso aos arquivos.
- B) Outra característica do Linux é com relação aos drivers de dispositivos e suporte a rede, os quais podem ser compactadas e utilizadas como se fossem módulos ou bibliotecas (LKM em inglês Loadable Kernel Modules), separados pela parte principal, cujo carregamento pode ser ativado após a execução do núcleo.
- C) No quesito portabilidade, o Linux funciona com eficiência em plataformas como x64 da Intel (EM64T e AMD64) PowerPC, Alpha, SPARC, porém é de difícil instalação nos sistemas embarcados como PVR, celulares, Tv's e Handhelds.
- D) A partir da década de 90, ao passo que a distribuição do Linux se popularizou, foi também limitada, pois se torna uma alternativa no uso de software livre, contra os sistemas operacionais da Apple (Mac OS) e Microsoft (Windows).

Comentários:

Em relação à **portabilidade**, o Linux se sai muito bem! Ele não foi pensado para ser um sistema portátil, mas acabou indo nessa direção, tendo um dos núcleos de sistemas operacionais mais portáteis, sendo executado em sistemas desde o iPaq (um computador portátil) até o IBM S/390 (um mainframe). Na atualidade o Linux hoje funciona em dezenas de plataformas, desde mainframes até um relógio de pulso, passando por diversas arquiteturas: x86 (Intel, AMD), x86-64 (Intel EM64T, AMD64), ARM, PowerPC, Alpha, SPARC, entre outras, além de sistemas embarcados (*handhelds*, consoles de videogames, PVR – *Personal Video Recorder*, telefones celulares, TVs etc.).

Gabarito: C

24.(2018 - CESPE - FUB)

O Linux é um sistema operacional em que cada usuário consegue ter apenas um processo ativo por vez, processo esse que é iniciado automaticamente quando o sistema é carregado.



Comentários:

O Linux permite que vários processos sejam executados ao mesmo tempo (**multitarefa**), mesmo que haja apenas um processador na máquina. O sistema operacional faz o escalonamento dos processos no processador, dando a ilusão de um paralelismo (ocorre um pseudoparalelismo). O comando "ps", por exemplo, pode ser executado e você verá diversos processos em execução. Quando há a criação de um processo novo, ele passa a ser filho de outro e a chamada de sistema utilizada é a fork().

Gabarito: Errado

25.(2018 - COPESE - UFT)

O sistema operacional Linux é reconhecido por permitir diversos níveis de personalização, inclusive de suportar o uso de vários ambientes gráficos. Assinale a alternativa que NÃO constituiu uma interface gráfica usada no Linux

- A) Pantheon.
- B) XFCM.
- C) MATE Desktop.
- D) Cinnamon Desktop.

Comentários:

O servidor de janelas é um ambiente gráfico (que se parece bastante com o sistema operacional Windows), tornando possível que tarefas sejam executadas sem a digitação de comandos (mas o usuário pode abrir um shell dentro do ambiente gráfico, se quiser). Abaixo podemos ver dois exemplos, o Gnome (à esquerda) e o KDE (à direita). Além desses dois, outros exemplos são Pantheon, MATE, Cinnamon, XFCE, Deepin, entre outros.

Sacanagem foi colocar "XFCM" como alternativa. O correto seria "XFCE"!

Gabarito: B

26.(2018 - CCV-UFC - UFC)

De acordo com a linha do arquivo '/ etc / passwd' exibida abaixo, qual das seguintes afirmações é verdadeira?

```
jferreira:x:502:1000:Joao Ferreira:/home/jferreira:/bin/bash
```

- A) O usuário Joao Ferreira possui a senha 'x'.
- B) Shadow passwords (senhas ocultas) são utilizados no sistema.
- C) O usuário Joao Ferreira pertence ao grupo com groupID 502.
- D) Membros do groupID 502 podem ler o diretório /home/jferreira.
- E) O nome de usuário (username) 'jferreira' pertence ao grupo 'jferreira'.



Comentários:

O segundo campo seria a senha (antigamente até era), mas há um bom tempo o *hash* da senha é colocado no arquivo `/etc/shadow` e no lugar dela é colocado um "x". Ou seja, esse "x" indica que *shadow passwords* (senhas ocultas) são utilizados no sistema.

Gabarito: B

27.(2018 - Quadrix - CRM-PR)

É possível instalar, em um mesmo computador desktop, os sistemas operacionais Windows 7 e Linux, de forma independente.

Comentários:

É possível, sim! Cada um em uma partição e um gerenciador de partição é utilizado para criar um menu e mostrá-lo quando você liga a máquina. Um exemplo clássico é o GRUB.

Gabarito: Certo

28.(2018 - Quadrix - CRM-PR)

O usuário root do sistema Linux tem autonomia para acessar arquivos de outros usuários, com exceção dos usuários administradores.

Comentários:

Root pode tudo! Até mesmo acessar os arquivos de outros usuários administradores!

Gabarito: Errado

29.(2018 - IDECAN - CRF-SP)

"O arquivo _____ só é legível pelo superusuário e serve para manter senhas criptografadas protegidas contra o acesso não autorizado. Ele também fornece informações sobre contas que não são disponíveis em _____." Assinale a alternativa que completa correta e sequencialmente a afirmativa sobre o Sistema Operacional Linux.

- A) `/etc/passwd; /etc/group`
- B) `/etc/group; /etc/passwd`
- C) `/etc/passwd; etc/shadow`
- D) `/etc/shadow; /etc/passwd`

Comentários:

Lembre-se que no `/etc/passwd` é possível verificar alguns dados do usuário (menos a senha, tem um "x" em seu lugar). No `/etc/shadow` tem o *hash* da senha, além de outras informações (expiração da senha, entre outras).



Gabarito: D

30.(2018 - UERR - IPERON)

Um administrador de uma rede baseada em Linux deseja guardar no servidor os scripts especiais para iniciar ou interromper módulos e programas diversos. Para isso, ele precisa guardar essas informações dentro do diretório:

- A) /srv
- B) /opl
- C) /etc
- D) /lib
- E) home

Comentários:

A gente pensa que “etc” significa “etcetera”, mas o diretório quer dizer “*environment tables and controls*”, então os **arquivos de configuração** geralmente se encontram em **/etc**.

Gabarito: C

31.(2018 - UERR - IPERON)

Em um computador com Linux, deseja-se instalar o sistema de arquivos ext3, mas com o nível de Journaling no qual se grava mudanças em arquivos de metadados, que força que a escrita do conteúdo dos arquivos seja feita após a marcação de seus metadados. Esse nível de Journaling é denominado:

- A) ordered
- B) cshell.
- C) iso9660
- D) journal.
- E) fsreiser.

Comentários:

O Ext3 possui três modos de operação:

- **ordered** (default): o *journal* é atualizado no final de cada operação. Isso faz com que exista uma pequena perda de desempenho, já que a cabeça de leitura do HD precisa realizar duas operações de gravação, uma no arquivo que foi alterado e outra no *journal* (que é um arquivo especialmente formatado);
- **writeback**: o *journal* armazena apenas informações referentes à estrutura do sistema de arquivos (metadados) e não em relação aos arquivos propriamente ditos, é gravado de forma mais ocasional, aproveitando os momentos de inatividade.



- **journal**: é o mais seguro, todavia mais lento. Nesse modo, o *journal* armazena não apenas informações sobre as alterações, mas também uma cópia de segurança de todos os arquivos modificados, que ainda não foram gravados no disco. Por ser o mais lento, é o modo menos usado.

Gabarito: A

32.(2019 - CS-UFG - IF Goiano)

Os sistemas Linux buscam padronizar diretórios para a localização de arquivos. Comandos essenciais do sistema operacional, tais como `cat`, `tar`, `su`, `rm` e `pwd`, em geral, estão localizados nos seguintes diretórios:

- A) `/lib` e `/var`
- B) `/usr/lib` e `/lib`
- C) `/bin` e `/usr/bin`
- D) `/usr/bin` e `/var`

Comentários:

Comandos são executáveis (binários), então, como sabemos que a estrutura de diretórios no Linux é organizada pelo tipo de informação, só nos resta a alternativa que possui “bin” duas vezes. Vejamos:

- “`/usr/bin`”: executáveis dos programas;
- “`/bin`”: comandos básicos, como “`cd`”, “`ls`” e “`cat`”.

Gabarito: C

33.(2019 - FCC - SEFAZ-BA)

Um Auditor Fiscal da área de Tecnologia da Informação deseja desinstalar um pacote chamado `java-1.6.0-openjdk.x86_64` em linha de comando, como usuário `root`, no Red Hat Enterprise Linux 6. Para isso, terá que utilizar o comando

- A) `apt-get uninstall java-1.6.0-openjdk.x86_64`
- B) `apt-get remove -rf java-1.6.0-openjdk.x86_64`
- C) `rm -rf java-1.6.0-openjdk.x86_64`
- D) `yum remove java-1.6.0-openjdk.x86_64`
- E) `apt-get -rf java-1.6.0-openjdk.x86_64`

Comentários:



YUM: resolve dependências de pacotes para distribuições que utilizam o RPM. É um gerenciador de pacotes padrão incluído em algumas distribuições baseados no **REDHAT**, incluindo o Fedora e CentOS. Muito parecido com o APT, vejamos alguns comandos:

- Atualização dos repositórios do YUM: `sudo yum update`;
- Instalação de um pacote: `sudo yum install nome_do_pacote`;
- Remoção de um pacote: `sudo yum remove nome_do_pacote`;
- Busca de um pacote que será instalado: `sudo yum search nome_do_pacote`.

Gabarito: D

34.(2019 - IF-MS - IF-MS)

O sistema operacional Linux possui várias partições (áreas) em sua estrutura, cada uma com uma função definida. Assinale a alternativa que apresenta a partição que abriga a pasta raiz do sistema que contém arquivos essenciais ao seu pleno funcionamento:

- A) /boot
- B) /bin
- C) /
- D) /etc
- E) /root

Comentários:

No Linux o diretório raiz é representado por “/”, enquanto no Windows é representado por “\”.

Gabarito: C

35.(2019 - VUNESP - Câmara de Sertãozinho-SP)

Considere que o usuário de um computador com sistema operacional Linux deseja executar um comando em segundo plano (background) em um terminal Shell. Para efetivar essa ação, o usuário deve executar o comando

- A) e teclar a combinação Ctrl+c
- B) e teclar a combinação Ctrl+x
- C) seguido de bg
- D) seguido de &
- E) seguido de !

Comentários:

O mais utilizado para executar em background é utilizar “&” logo após o nome do binário ou script. Por exemplo, se você deseja executar um script “aprovacao.sh”, que vai realizar vários cálculos e no



fim escrever em um arquivo, não é necessário que você fique com o *shell* “travado”, aguardando a conclusão da tarefa. Então é só digitar (note que o *shell* não fica esperando a conclusão do *script*, já aguarda o próximo comando):

```
estrategia:~$ aprovacao.sh &  
estrategia:~$
```

Gabarito: D

36.(2019 - COSEAC - UFF)

No Sistema Operacional Linux o diretório local padrão do superusuário é:

- A) /home.
- B) /usr.
- C) /root.
- D) /lib.
- E) /dev.

Comentários:

Os “usuários comuns” possuem como diretório local padrão “/home/nome_usr” (ex.: /home/evandro), enquanto o *root* possui como diretório local padrão o “/root”.

Gabarito: C

37.(2019 - CESPE - SLU-DF)

Windows e Linux são classificados como sistemas operacionais de tempo real crítico, porque fornecem garantias absolutas de que todas as suas ações ocorrerão dentro de intervalos de tempo determinados.

Comentários:

Windows e Linux são sistemas operacionais interativos, pois recebem instruções por teclado, mouse, caneta ótica etc., realiza as tarefas, mostra os resultados via impressora, monitor de vídeo etc. Logo, há muita interação, o que é diferente de sistemas de tempo real ou ainda os sistemas de lote (são as 3 classificações relacionadas a esse assunto).

Gabarito: Errado

38.(2019 - INAZ do Pará - CORE-SP)

“Todos os arquivos e diretórios do sistema Linux instalado no computador partem de uma única origem: o diretório raiz. Mesmo que estejam armazenados em outros dispositivos físicos.”



Disponível em: <https://canaltech.com.br/linux/entendendo-a-estrutura-de-diretorios-dolinux/>.
Acesso em: 13.12.2018

Qual pasta no sistema operacional LINUX tem a função de armazenar arquivos de dispositivos do sistema?

- A) /dev.
- B) /etc.
- C) /lib.
- D) /var.
- E) /proc.

Comentários:

Dispositivos em inglês são *devices*. Como Linux geralmente é intuitivo...a resposta é “/dev”.

Gabarito: A

39.(2019 - INAZ do Pará - CORE-SP)

“O agendamento de tarefas é um recurso muito interessante para a administração de sistemas operacionais. É possível programar a execução de scripts de manutenção do sistema, disparar envio de newsletters, gerar relatórios de análises de logs, entre outros”

Disponível em: [https://www.vivaolinux.com.br/dica/Agendamento-de-tarefas-no-Linux-\(cron-e-at\)](https://www.vivaolinux.com.br/dica/Agendamento-de-tarefas-no-Linux-(cron-e-at)). Acesso em: 13.12.2018

Qual programa do sistema operacional LINUX executa tarefas de modo automático, em horários pré-determinados, e executa tarefas que não foram executadas, enquanto o sistema esteve desligado?

- A) Fcron.
- B) Cron.
- C) Anacron.
- D) Schedule.
- E) Crontab.

Comentários:

Para o agendamento de tarefas a aplicação mais conhecida é o **Cron**, que já existe há muito tempo e passou por muitos estágios de evolução. Porém, a maioria das suas implementações atuais (Vixie Cron, ISC Cron, BCron etc.) ainda se baseiam no pressuposto de que o sistema está em funcionamento de forma ininterrupta. Isso é um problema quando se utiliza a virtualização, quando os sistemas que operam sob demanda, sendo ligados e desligados conforme sua necessidade.

Para contornar essa situação, algumas distribuições contam com o Anacron como alternativa ao Cron. O Anacron permite a criação de listas de tarefas que serão realizadas em intervalos pré-definidos e, quando o Anacron é inicializado, ele verifica essas listas e executa as tarefas ainda não



realizadas. No entanto, o Anacron possui algumas limitações. A primeira é que ele não é um daemon, então precisa ser executado sempre que for necessário.

Outra questão é que o Anacron não está preparado para lidar com períodos de tempo menores do que dias. A combinação desses problemas pode levar a situações nas quais o Cron e o Anacron funcionem ao mesmo tempo, e algumas tarefas podem ser executadas duas vezes ou nenhuma!

Uma solução é o **Fcron**, que faz o que o Vixie Cron e o Anacron fazem e ainda mais. É possível usar o Fcron para agendar cronjobs com data e hora fixas, com intervalos de tempo ou até mesmo de acordo com a disponibilidade do sistema.

Gabarito: A

40.(2019 - IDECAN - IF-PB)

O Debian, uma das distribuições do sistema operacional Linux, possui um conjunto de pastas com nome pré-estabelecidos. Essas pastas possuem significado específico para o sistema operacional, visando atender necessidades do mesmo. Assinale a alternativa que indica corretamente o nome da pasta que armazena arquivos referentes às instalações de programas não oficiais da distribuição do sistema operacional.

- A) /bin
- B) /lib
- C) /opt
- D) /var
- E) /proc

Comentários:

Programas não oficiais poderiam ser chamados de “opcionais” (*optionals*), então vamos usar a lógica de nomes do Linux... “/opt”.

Gabarito: C

41.(2019 - UFRR - UFRR)

A respeito de diretórios no sistema operacional Linux, a estrutura é conhecida como árvore invertida, isto é, a raiz da árvore de diretórios é o topo. Qual é a representação do diretório raiz, e nome do usuário que possui privilégio para escrever neste diretório.

- A) (raiz) e usuário root
- B) \\ (raiz) e usuário administrador
- C) / (raiz) e usuário root
- D) [(raiz) e usuário administrador



E)]/ (raiz) e usuário root

Comentários:

Quem está acostumado com Windows sabe que o diretório raiz é “\”, mas no Linux é invertida: “/”. No Windows o usuário que “pode tudo” é o “Administrador”, enquanto no Linux é o “root”.

Gabarito: C

42.(2019 - IF-ES - IF-ES)

O Network File System (NFS) – Sistemas de Arquivos em Rede – tem como um dos principais propósitos dar suporte a um sistema heterogêneo, no qual clientes e servidores estejam possivelmente executando sistemas operacionais e hardwares diferentes. Sobre o NFS, é CORRETO afirmar:

- A) O NFS utiliza dois protocolos cliente-servidor, em que o primeiro é responsável pela montagem e o segundo é para acesso de diretório e arquivos.
- B) O servidor tem total gerência sobre o ponto de montagem nos clientes.
- C) Os serviços NFS são implementados apenas nos servidores Linux.
- D) Como critério de segurança, os clientes não podem ter acesso aos atributos dos arquivos.
- E) O NFS faz uso de máquinas distintas para servidores e clientes, impossibilitando que a mesma máquina seja tanto cliente quanto servidor

Comentários:

Para que os clientes possam acessar o servidor NFS é necessário que os seguintes *daemons* estejam em execução:

- **nfsd**: *daemon* NFS, atende requisições dos clientes NFS;
- **mountd**: *daemon* de montagem NFS, executa as solicitações passadas pelo nfsd;
- **portmap**: *daemon portmapper*, permite que clientes NFS descubram qual porta o servidor NFS está utilizando.

Para essa questão, esqueça o *portmap*, pois só serve para descobrir a porta do servidor. Os outros dois *daemons* fazem o papel dos protocolos citados na alternativa A: um atende requisições dos clientes (acessos a diretórios e arquivos) e o outro faz a montagem.

Gabarito: A

43.(2019 - FCC - Prefeitura de Manaus-AM)

Em computador com sistema operacional Linux e o OpenLDAP instalado, o programador editou o arquivo `ldap.conf` para configurar

- A) a autenticação do servidor.



- B) a execução do serviço do LDAP.
- C) o banco de dados a ser utilizado.
- D) a geração de índices do banco de dados.
- E) o acesso dos clientes ao diretório.

Comentários:

O principal arquivo de configuração (pelo menos para concursos) é o “`/etc/ldap.conf`”, que é usado para configurar os **padrões a serem aplicados quando da execução dos clientes LDAP**.

Gabarito: E

USUÁRIOS, GRUPOS, PERMISSÕES, CONTROLES DE ACESSO E MANIPULAÇÃO DE ARQUIVOS E DIRETÓRIOS

Um novo usuário pode ser adicionado utilizando-se ferramentas, como `adduser` ou `useradd`. A partir desse momento o usuário criado é adicionado nos arquivos `/etc/passwd` e `/etc/shadow`. Os arquivos recebem um número **UID** referente à identificação do usuário (dono), **GID** referente ao seu grupo, entre outros dados. Porém, o usuário `root`, por questões de segurança, não dispõe de UID (ou melhor, recebe um `id=0`).

Os números UID e GID variam de 0 a 65535 (dependendo do sistema, o valor limite pode ser maior). No caso do **usuário root**, **esses valores são sempre 0 (zero)**. Assim, para fazer com que um usuário tenha os mesmos privilégios que o `root`, é necessário que seu **GID seja 0**. Isso informa ao sistema que o usuário em questão é um superusuário.

Cada arquivo/diretório no Linux possui um conjunto de **permissões de acesso** e propriedades em três **níveis**: usuário dono do arquivo (**U**), grupo proprietário (**G**) e qualquer outro usuário que não se encaixe nos níveis anteriores (**O**).

Os tipos de **permissão** são:

- Read (**r**) - Leitura: permissão para visualizar o conteúdo do arquivo;
- Write (**w**) - Escrita: permissão para alterar o conteúdo do arquivo;
- eXecute (**x**) - Execução: permissão para executar o arquivo.

Dessa forma definimos a seguinte ordem:

U			G			O		
r	w	x	r	w	x	r	w	x

Existem também algumas **permissões especiais**:



- **setuid**: privilégio do dono e não de quem executou, ex.: passwd executado pelo evandro → privilégio de root, mas quem executou foi o usuário evandro!
- **setgid**: tem efeito tanto em arquivos como em diretórios. Para um arquivo, executa com privilégios do grupo do usuário dono. Para um diretório, o grupo dos arquivos criados dentro do diretório será o mesmo do diretório pai. Abaixo podemos notar um “s” no lugar do “x” do grupo. Isso demonstra que o setgid está com valor 1.

```
$ ls -ld teste
drwxrwsr-x. 2 egdoc egdoc 4096 Nov  1 17:25 teste
```

- **sticky bit**: quando usado em um diretório, todos os arquivos dentro dele poderão ser modificados somente por seus donos. Exemplo típico: “/tmp”, um diretório temporário onde todos podem colocar arquivos, mas cada um pode modificar o seu. Abaixo podemos notar um “t” no lugar do “x” dos outros. Isso demonstra que o sticky bit está com valor 1.

```
$ ls -ld /tmp
drwxrwxrwt. 14 root root 300 Nov  1 16:48 /tmp
```

Vamos rever nossa figura do “UGO” com as permissões especiais:

			U			G			O		
<u>uid</u>	<u>gid</u>	<u>sticky</u>	r	w	x	r	w	x	r	w	x

Analisando apenas as permissões especiais, se o setuid estiver ativo temos 100 = 4 (em decimal). Se tivermos o setgid ativo, temos 010 = 2 e se o sticky bit estiver ativo, temos 001 = 1.

Para **visualizar as permissões** é comum utilizarmos o argumento “-l” no comando “ls”, pois esse argumento lista arquivos com informações adicionais, incluindo colunas com as permissões, nome do usuário e grupo do dono:

```
[root@localhost etc]# ls -l teste
-rw-r--r-- 1 root root 6 Jul 22 18:43 teste
[root@localhost etc]#
```

Na figura acima podemos ver que se trata de um arquivo (o primeiro caractere é “-” = arquivo, se fosse diretório seria “d”). Depois vemos as permissões do **Usuário** (rw-), **Grupo** (r--) e **Outros** (r--). Não vemos permissões especiais.

Ok, e se eu quiser **modificar as permissões** depois de ter criado um arquivo? Aí temos o comando **chmod**. Vamos ver algumas formas de utilizá-lo:

chmod u=rwx, g=rx, o=x nome_arquivo → modifica as permissões do “nome_arquivo” para:



- usuário = completo (*read, write, execute*);
- grupo = leitura e execução (*read, execute*);
- outros = apenas execução (*execute*).

`chmod 0751 /home/evandro/arquivo.txt` → aqui temos um exemplo utilizando números em octal e temos a presença das permissões especiais (o primeiro número). Como o primeiro número é 0, fica fácil, pois em binário ficaria 000 (setuid = 0, setgid = 0, sticky bit = 0). Agora vamos ao UGO:

- usuário = 7 = 111 = completo (*read, write, execute*);
- grupo = 5 = 101 = leitura e execução (*read, execute*);
- outros = 1 = 001 = apenas execução (*execute*).

Acabamos de ver que os comandos acima são equivalentes em termos das permissões concedidas!

Um último exemplo:

`chmod +x teste` → dá a permissão de execução ao arquivo “teste”, mas para quem? U? G? O? Como não está explícito para quem é, todos recebem o “+x”. Veja abaixo antes e depois de aplicar o “`chmod +x teste`”. Tem um “`ls -l teste`” antes e depois para podermos verificar que realmente tanto o usuário, como o grupo e os outros receberam a permissão de execução (x).

```
[root@localhost etc]# ls -l teste
-rw-r--r-- 1 root root 6 Jul 22 18:43 teste
[root@localhost etc]# chmod +x teste
[root@localhost etc]# ls -l teste
-rwxr-xr-x 1 root root 6 Jul 22 18:43 teste
[root@localhost etc]#
```

Agora vamos ver como modificar o dono ou o grupo.

Embora o comando **chown** signifique *change owner*, ele permite a alteração do dono ou do grupo do arquivo/diretório. A opção “-R” aplica a arquivos/diretórios recursivamente e “-c” mostra o resultado. Vamos ver uns exemplos a seguir (note que o prompt mudou para “#”, o que indica que está executando como *root*, afinal tem que ser *root* para mudar um arquivo de dono ou de grupo).

```
# chown -R evandro /teste
```

→ “evandro” virou dono da pasta /teste, de forma recursiva.

```
# chown evandro:inf /teste
```

→ “/teste” passou a ter o dono “evandro” e o grupo “inf” (mudou os dois de uma vez!).

Por fim, um comando que altera apenas o grupo de um arquivo/diretório, o **chgrp**:

```
#chgrp -R inf /teste
```

→ Recursivamente, mudou o dono de grupo do diretório “/teste” para “inf”.



Alguns comandos para **criar/excluir arquivos e diretórios, trocar o diretório corrente e verificar o diretório corrente** são:

- mkdir: “make directory”, cria um diretório, ex.: mkdir /home/evandro/teste cria o diretório “teste” dentro de /home/evandro (se não hover esse caminho, ocorre um erro);
- cd: “change directory”, muda o diretório corrente, exemplos:
 - cd .. (muda para o diretório pai);
 - cd / (muda para o diretório rai);
 - cd ../teste (sobe um nível e “entra” no diretório teste);
 - cd teste (“entra” no diretório teste, que deve estar abaixo do diretório atual);
- pwd: mostra o diretório atual;
- rmdir: remove o diretório, ex.: rmdir /root/teste;
- touch: cria um arquivo vazio (entre outras funções), ex.: touch nome_arquivo;
- rm: remove um arquivo, ex.: rm nome_arquivo.
- echo: pode criar um arquivo de texto, se utilizado o redirecionamento de saída “>” ou “>>”, exemplos:
 - echo oi > arquivo.txt → cria um arquivo com o conteúdo “oi”;
 - echo oi >> arquivo.txt → adiciona no fim do arquivo.txt o conteúdo “oi”(se não existir, cria o arquivo).

Vamos ver uma sequência desses comandos a seguir, para melhorar o entendimento.

```
[root@localhost ~]# mkdir teste
[root@localhost ~]# cd teste
[root@localhost teste]# pwd
/root/teste
[root@localhost teste]# mkdir opa
[root@localhost teste]# ls
opa
[root@localhost teste]# rmdir opa
[root@localhost teste]# ls
[root@localhost teste]# touch arquivo_novo
[root@localhost teste]# ls -las
total 8
 4 drwxr-xr-x  2 root  root    66 Nov 19  2019 .
 4 drwx----- 4 root  root   176 Oct 24  2017 ..
 0 -rw-r--r--  1 root  root     0 Nov 19  2019 arquivo_novo
[root@localhost teste]# rm arquivo_novo
[root@localhost teste]# ls -las
total 8
 4 drwxr-xr-x  2 root  root    37 Nov 19  2019 .
 4 drwx----- 4 root  root   176 Oct 24  2017 ..
[root@localhost teste]# cd /
[root@localhost ~]# pwd
/
[root@localhost ~]#
```



QUESTÕES COMENTADAS

44.(2014 - FUNCAB - MDA)

No sistema operacional Linux, a criação de um grupo permite que um conjunto de usuários diferentes possua acesso a um mesmo arquivo. O comando do Linux que tem como objetivo alterar o grupo de um arquivo/diretório é o

- A) chown.
- B) chgroup.
- C) chset.
- D) chfile.
- E) chgrp.

Comentários:

Sabemos que o chown permite a alteração de usuário/grupo, mas a questão fala somente em grupo. Tem uma pegadinha na letra B, mas seguindo o padrão de 5 letras (chown), a resposta é “chgrp”.

Gabarito: E

45.(2018 - FGV - Prefeitura de Niterói-RJ)

Pedro é o proprietário do arquivo header.txt em um sistema Linux e gostaria de assegurar que somente ele tivesse permissão de leitura, gravação e execução a este arquivo, enquanto que todos os demais usuários com acesso ao sistema tivessem somente a permissão de leitura.

Assinale a opção que indica o comando que pode ser usado para conseguir esse objetivo.

- A) chmod ug+r header.txt
- B) chmod 766 header.txt
- C) chmod 722 header.txt
- D) chmod +r header.txt
- E) chmod 744 header.txt

Comentários:

Vamos fazer na sequência o UGO (usuário, grupo, outros):

U = r w x = 111 → 7

G = r - - = 100 → 4

O = r - - = 100 → 4

Gabarito: E



46.(2018 - FGV - MPE-AL)

Para definir o script `/usr/local/bin/meuscript` como executável no sistema operacional Linux, devemos usar o comando

- A) `chmod og /usr/local/bin/meuscript`
- B) `chmod 000 /usr/local/bin/meuscript`
- C) `chmod +x /usr/local/bin/meuscript`
- D) `chmod ugo-x /usr/local/bin/meuscript`
- E) `chmod 666 /usr/local/bin/meuscript`

Comentários:

A única que adiciona a execução “x” é a letra C. Como não especifica para “quem”, todo o UGO recebe a permissão para a execução.

Na letra E, se transformarmos 666 em binário, temos: 110 110 110 = rw- rw- rw- (não tem “x” para ninguém!).

Gabarito: C

47.(2019 - IF-ES - IF-ES)

Os administradores de redes precisam entender com clareza sobre o gerenciamento de contas de usuários nos Sistemas Operacionais de Servidores Linux. Em relação ao sistema de contas Linux, analise as afirmativas abaixo:

- I – O arquivo `/etc/shadow` serve para manter senhas criptografadas seguras quanto a acesso não autorizado.
- II – Um novo usuário pode ser adicionado utilizando-se ferramentas, como `adduser` ou `useradd`.
- III – Os usuários dos sistemas Linux recebem um número UID referente a sua identificação, porém o usuário `root`, por questões de segurança, não dispõe de UID.

É CORRETO o que se afirma em:

- A) I, II e III.
- B) I e II, apenas
- C) I e III, apenas.
- D) II e III, apenas.
- E) Nenhuma das afirmativas está correta.

Comentários:

(I) O arquivo `/etc/shadow` serve para manter o hash das senhas, entre outras informações relacionadas. (II) Um novo usuário pode ser adicionado utilizando-se ferramentas, como `adduser` ou `useradd` e automaticamente os dados serão acrescentados nos arquivos `/etc/passwd` e `/etc/shadow`. (III)



Os usuários dos sistemas Linux recebem um número UID referente a sua identificação. O root recebe o UID 0.

Gabarito: B

48.(2019 - CS-UFG - IF Goiano)

Na configuração do Sistema Linux, o comando chmod modifica as permissões de arquivos com respeito à execução, leitura e escrita. A expressão chmod 751 arq2 é equivalente a:

- A) chmod u=rx,g=rwx,o=x arq2
- B) chmod u=rx,g=rwx,o=r arq2
- C) chmod u=rwx,g=rx,o=x arq2
- D) chmod u=rwx,g=rx,o=r arq2

Comentários:

U = 7 = 111 = rwx

G = 5 = 101 = r-x

O = 1 = 001 = --x

Tirando os "tracinhos": u=rwx,g=rx,o=x

Gabarito: C

49.(2019 - CCV-UFC - UFC)

Qual dos itens abaixo contém o comando do sistema operacional Linux capaz de mudar o grupo de um arquivo ou diretório do sistema?

- A) df
- B) sed
- C) chown
- D) chmod
- E) passwd

Comentários:

Questão interessante porque não mostra o comando "chgrp". Mas sabemos que o chown permite a alteração de usuário/grupo, ou seja, é possível trocar o grupo também!

Gabarito: C



50.(2019 - FCC - TJ-MA)

Um Analista digitou o comando `chmod u=rwx,g=rx,o=r processo` para definir as permissões de acesso ao arquivo `processo`. O comando equivalente usando a notação octal é:

- A) `chmod 713 processo`
- B) `chmod 777 processo`
- C) `chmod 134 processo`
- D) `chmod 754 processo`
- E) `chmod 671 processo`

Comentários:

$U = rwx = 111 = 7$

$G = r-x = 101 = 5$

$O = 1-- = 100 = 4$

Então, temos “`chmod 754 processo`”.

Gabarito: D

51.(2019 - VUNESP - Câmara de Tatuí-SP)

Considere a execução da sequência de comandos a seguir, em um terminal shell do Linux:

```
# cd /root
```

```
# mkdir -p /root/foo/bar
```

```
# pwd
```

O resultado impresso na tela após a execução do último comando será:

- A) `/root`
- B) `/root/foo/bar`
- C) `bar`
- D) `/root/foo`
- E) `/foo/bar`

Comentários:

O comando `cd` troca de diretório, “indo” para `/root`.

O comando `mkdir` cria o diretório `/root/foo/bar`. O argumento “-p” faz com que, se houver erro no comando, que tente o diretório pai (“parent”). Vamos considerar que não houve erro.

O comando `pwd` mostra o diretório corrente. Aí tem uma leve pegadinha, pois foi criado o diretório `/root/foo/bar`, mas o diretório corrente ainda é o `/root`, pois o comando `cd` fez o “deslocamento” até esse diretório e não saiu mais! Logo, `pwd` deve retornar `/root`.



Gabarito: A

52.(2019 - CS-UFG - IF Goiano)

No sistema operacional GNU/Linux, usando a linha de comando, deseja-se executar o seguinte: a) criar um diretório chamado IFG; b) criar um arquivo de texto chamado Concurso.txt; c) apagar o arquivo de texto Concurso.txt; d) apagar o diretório IFG. Qual é a sequência de comandos a ser empregada?

Obs.: o sinal “,” é um mero separador entre os comandos

- A) mkdir IFG , touch Concurso.txt , rm Concurso.txt , rmdir IFG
- B) mkdir IFG , create Concurso.txt , kill Concurso.txt , killdir IFG
- C) create IFG , touch Concurso.txt , rm Concurso.txt , rmdir IFG
- D) create IFG , create Concurso.txt , kill Concurso.txt , killdir IFG

Comentários:

Questão bem objetiva. O mkdir serve para criar um diretório. Uma forma de criar um arquivo texto é através do comando touch. Para remover um arquivo utilizamos o comando rm. E para excluir um diretório, temos o comando rmdir.

Gabarito: A

SHELL SCRIPT

Linguagens de script são linguagens de programação executadas do interior de programas, a partir de outras linguagens de programação, entre outras situações. Elas servem para estender a funcionalidade de um programa e/ou controlá-lo, acessando sua API e, são frequentemente usadas como **ferramentas de configuração e instalação em sistemas operacionais**, como por exemplo, em alguns sistemas operacionais da família Linux.

O Unix Shell é ao mesmo tempo um interpretador de comandos e uma linguagem de programação. Como interpretador de comandos, ele dá acesso ao rico conjunto de utilidades do GNU e como linguagem de programação ele permite que tais utilidades sejam combinadas. Arquivos contendo comandos podem ser criados e se tornar comandos (assim como os arquivos em lote no Windows - .BAT). Esses novos comandos tem o mesmo status de comandos de sistema como os do diretório /bin. Importante destacar que Shell é uma linguagem interpretada, ou seja os comandos são interpretados na sequência, um a um, sem haver uma compilação.

Na sequência vamos ver um passo a passo de como criar um shell script.



Criando um arquivo:

Há dois modos de realizar essa ação: via modo gráfico ou via terminal. No segundo caso, pode ser utilizado o comando `vi`, conforme mostrado a seguir (note a “extensão” “.sh”).

`vi script.sh` → Será criado e aberto um arquivo de leitura e escrita.

Outra opção é digitar o comando `touch`:

`touch script.sh` → cria um arquivo sem abri-lo.

Permissão ao arquivo:

Para começar a editar o arquivo, é necessário conceder a permissão de escrita a ele. Uma opção é utilizar o comando:

`chmod 777 script.sh` → o comando `chmod` é utilizado para conceder permissões em diretórios e arquivos, enquanto o valor `777` permite que o usuário tenha total liberdade para editar o arquivo (na verdade dá liberdade total a qualquer usuário do sistema! Mas nosso foco agora não é estudar o `chmod`).

Pensando mais em segurança, pode-se, por exemplo, liberar apenas a execução (x):

`chmod +x script.sh` → como não foi especificado para quem foi liberada a execução (usuário, grupo ou todos), todos devem ter a permissão de execução.

Editar e executar o arquivo:

Depois da permissão do arquivo, vamos para a edição do arquivo com o comando `vi`:

`vi script.sh` → é necessário digitar “i” para colocar o `vi` no modo “inserção”.

Agora é necessário **definir qual Shell será utilizado na primeira linha do arquivo** (vamos escolher o `bash`, mas poderia ser outro interpretador de comandos, como `sh`, `ksh` ou `csh`):

```
#!/bin/bash
```

Inserir comentários:

Basta utilizar o caractere “#” para inserir o comentário, como mostrado abaixo:

```
# A linha abaixo serve para bla bla bla
```

Uso das variáveis:

Para declarar a variável, a sintaxe é a seguinte:

`nome_variavel = valor` → o valor pode ser string ou número. Para utilizar o valor, devemos colocar o símbolo “\$” na frente do nome que será inserido.



Entrada de dados do usuário:

Se o script pedir ao usuário o fornecimento de dados para a entrada do processamento, o programador precisará digitar um comando ler esses dados:

```
read nome_variavel
```

Comando de seleção:

Um comando simples que permite a execução de uma tarefa baseada na ação de um usuário que esteja usando o seu sistema é o condicional. A sintaxe é a seguinte:

```
if [ CONDIÇÃO ] ;  
then  
AÇÕES  
fi
```

É importante lembrar que para cada “if” aberto, deve haver um “fi” que feche a sequência. A mesma regra serve para os colchetes.

Funções:

Para organizar, separar e estruturar um algoritmo, é necessário o uso de funções:

```
nome_funcao()  
{  
AÇÕES  
}
```

Inserção de argumentos:

O shell script recebe dados fornecidos por outro programa ou por um usuário com a finalidade de produzir saídas, são os **argumentos**. Alguns nomes que qualificam os argumentos são:

```
$# - total de argumentos que foram passados;  
$* - retorno aos argumentos;  
$0 - nome do script executado.
```

Um **exemplo** simples é mostrado abaixo (script.sh), sendo o script à esquerda e a saída de sua execução à direita. Note que na execução do script são passados 3 parâmetros (a, b, c) e na chamada da função “imprime()” são passados 4 parâmetros (um, dois, tres, quatro).



<pre>#!/bin/sh imprime () { echo "Sou o programa \$0" echo "Recebi \$# parametros" echo "Param 1: \$1" echo "Param 2: \$2" echo "Lista de parâmetros: \$*" } imprime um dois tres quatro echo "Sou o programa \$0" echo "Recebi \$# parametros" echo \$1 \$2 \$3</pre>	<pre>\$./script.sh a b c Sou o programa teste.sh Recebi 4 parametros Param 1: um Param 2: dois Lista de parâmetros: um dois tres quatro Sou o programa script.sh Recebi 3 parametros a b c</pre>
--	---

Vale lembrar que para um bom entendimento de um shell script, é importante conhecer o funcionamento dos comandos, e não são poucos! Mas, por enquanto, as questões sobre esse assunto não pegaram pesado.

QUESTÕES COMENTADAS

53.(2011 - CESGRANRIO - Petrobras)

Em um sistema Unix, um arquivo de script chamado teste.sh foi copiado para o diretório /tmp. No shell do sistema, o usuário submeteu dois comandos: cd /tmp e teste.sh. Após a execução do segundo comando, o shell informou uma mensagem de erro, indicando comando não encontrado. O que deve ser feito para corrigir o problema que gerou essa mensagem?

- A) Certificar que o usuário não entrou com letras maiúsculas.
- B) Certificar que o arquivo tem permissão para ser executado.
- C) Omitir a extensão .sh ao entrar com o nome do script.
- D) Incluir ./ antes do nome do script.
- E) Mover o arquivo para o diretório home e executá-lo.

Comentário:



No Linux, quando um executável (binário ou um script com permissão de execução) não estiver no *path* (lista de diretórios que permitem a execução), é possível executar através de “./” antes do nome do executável.

Gabarito: D

54.(2012 - CESGRANRIO - Petrobras)

No ambiente UNIX, existem vários interpretadores de linha de comando conhecidos como shell. É importante, para cada script, informar em que shell ele deve ser executado.

Para isso, o usuário pode especificar o shell desejado

- A) na primeira linha do script.
- B) na última linha do script.
- C) em qualquer linha do script.
- D) em um arquivo à parte.
- E) na linha de comando, após o nome do arquivo que contém o script.

Comentário:

É necessário **definir qual Shell será utilizado na primeira linha do arquivo** (vamos escolher o bash, mas poderia ser outro interpretador de comandos, como sh, ksh ou csh):

```
#!/bin/bash
```

Gabarito: A

55.(2014 - CEPERJ - Rio previdência)

Shell script é uma linguagem de script para Linux, nada mais do que comandos do próprio Linux que são executados em uma determinada sequência para uma determinada finalidade. Nesse contexto, duas situações são listadas a seguir.

I- No terminal ou modo gráfico, deseja-se criar um arquivo que possa ser editado para que se torne o primeiro shell script a ser criado, sendo necessário utilizar um comando CMD1.

II- Para que seja possível executar o shell script criado, é preciso atribuir a este o direito de execução; para isso é necessário usar um comando CMD2.

Exemplos de CMD1 e de CMD2 são, respectivamente:

- A) touch shell1.sh e exec +x shell1.sh
- B) touch shell1.sh e chmod +x shell1.sh
- C) create shell1.sh e chmod +x shell1.sh
- D) new shell1.sh e chmod +x shell1.sh



E) `new shell1.sh` e `exec +x shell1.sh`

Comentário:

- Criando um arquivo:

Há dois modos de realizar essa ação: via modo gráfico ou via terminal. No segundo caso, pode ser utilizado o comando `vi`, conforme mostrado a seguir (note a “extensão” “.sh”).

`vi script.sh` → Será criado e aberto um arquivo de leitura e escrita.

Outra opção é digitar o comando `touch`:

`touch script.sh` → cria um arquivo sem abri-lo.

- Permissão ao arquivo:

Para começar a editar o arquivo, é necessário conceder a permissão de escrita a ele. Uma opção é utilizar o comando:

`chmod 777 script.sh` → o comando `chmod` é utilizado para conceder permissões em diretórios e arquivos, enquanto o valor `777` permite que o usuário tenha total liberdade para editar o arquivo (na verdade dá liberdade total a qualquer usuário do sistema! Mas nosso foco agora não é estudar o `chmod`).

Pensando mais em segurança, pode-se, por exemplo, liberar apenas a execução (x):

`chmod +x script.sh` → como não foi especificado para quem foi liberada a execução (usuário, grupo ou todos), todos devem ter a permissão de execução.

Gabarito: B

56.(2014 - FUMARC - AL-MG)

No Linux, considere a existência de um arquivo “arquivo.txt” no diretório corrente, com o seguinte conteúdo:

linha 1 valor 10 linha 2 valor 20 linha 3 valor 30

Deseja-se produzir, a partir do arquivo, a seguinte saída:

1 - 10 2 - 20 3 - 30

O script shell usado com comandos AWK que produz a saída desejada é:

A) `awk arquivo.txt '{ print $2" - "$4 }'`

B) `cat arquivo.txt | awk '{ print $2" - "$4 }'`

C) `cat arquivo.txt | awk '{ print $1" - "$3"\n" }'`

D) `cat arquivo.txt | awk '{ printf($2" - "$4) }'`

Comentário:



O comando `cat` é usado para unir, criar e exibir arquivos. No caso dessa questão, serve para exibir.

O `awk` é um utilitário especializado em manipulação de texto.

O pipe ("`|`") serve para uma comunicação entre os processos, ou seja, a saída do que está na esquerda dele é passada para o que está à direita.

Então:

`cat arquivo.txt` → exibe o conteúdo do arquivo e esse conteúdo é passado para:

`awk '{ print $2" - "$4 }'` → imprime o 2º argumento, um traço ("`-`"), o 4º argumento e continua processando o texto até o fim da linha. Logo, imprime:

linha 1 valor 10 linha 2 valor 20 linha 3 valor 30

Lembrando os traços entre o 2º e o 4º argumentos, fica assim:

1 - 10 2 - 20 3 - 30

Gabarito: B

57.(2015 - VUNESP - CRO-SP)

No sistema operacional UNIX, uma aplicação pode ser desenvolvida por meio de um script que é um conjunto de comandos que podem ser executados pelo interpretador de comandos (shell). Considerando a existência do interpretador `/bin/sh`, a primeira linha de um arquivo de script deve conter:

A) `*/bin/sh`

B) `$/bin/sh`

C) `&/bin/sh`

D) `#!/bin/sh`

E) `#*/bin/sh`

Comentário:

Para definir qual Shell será utilizado, deve-se especificar na primeira linha do arquivo (vamos escolher o `sh`, conforme a questão, mas poderia ser outro interpretador de comandos, como `bash`, `ksh` ou `csh`):

`#!/bin/sh`

Gabarito: D

58.(2016 - INSTITUTO AOCP - EBSERH)

Em sistemas operacionais Linux é possível criar scripts para automatizar tarefas rotineiras. A extensão de arquivo utilizada para Shell Script é



- A) .bin
- B) .sl
- C) .bash
- D) .sh
- E) .gz

Comentário:

Vimos em aula o “script.sh”, então o padrão utilizado como “extensão” é o “.sh”.

Gabarito: D

59.(2016 - FCC - ELETROBRAS-ELETROSUL)

Um profissional de TI está usando um computador com sistema operacional Linux que utiliza no shell o interpretador de comandos bash. Ele está logado como usuário teste e criou o seguinte arquivo shell script:

```
1 #!/bin/bash
2 echo 'Eletrosul- Centrais Elétricas S.A.'
3 $ variavel= 'Eu estou logado como usuário $user'
4 $ echo $variavel
```

Considerando que 1, 2, 3 e 4 indicam as linhas do arquivo e que este tenha sido salvo com o nome exemplo, é correto afirmar:

- A) Para o arquivo ser executável, é necessário acionar o comando `$ chmod +x exemplo`. Depois disto o arquivo poderá ser executado com `./exemplo`.
- B) A linha 1 indica que todas as outras linhas abaixo deverão ser executadas pelo compilador sh, que se localiza em `/bin/bash`.
- C) Após ser executado, o arquivo imprimirá na tela apenas frase “Eletrosul – Centrais Elétricas S. A.” utilizando o comando echo.
- D) Ao acionar o comando file arquivo é possível ver que a definição dele é Bourne-Again Shell Script, que se refere ao bash script.
- E) As linhas 3 e 4 farão com que seja impresso na tela Eu estou logado como usuário \$teste.

Comentário:

Vamos analisar o script:

`#!/bin/bash` → especifica o shell bash

`echo 'Eletrosul- Centrais Elétricas S.A.'` → imprime na tela 'Eletrosul- Centrais Elétricas S.A.'

`$ variavel= 'Eu estou logado como usuário $user'` → A variável recebe a string 'Eu estou logado com o usuário teste'



\$ echo \$variavel → desconsidere \$ inicial, então imprime 'Eu estou logado com o usuário teste'

Analisando as alternativas, a única correta é a que menciona que para o arquivo (script) ser executável, é necessário acionar o comando "chmod +x exemplo" (dá a permissão de execução a todos, pois não especificou a quem seria). Depois disto o arquivo poderá ser executado com "./exemplo".

Gabarito: A

GERENCIAMENTO DE PROCESSOS

As estruturas internas do kernel registram informações sobre cada processo, incluindo, entre outras:

- Mapa de espaço de endereços do processo;
- Estado atual (espera, parado, em execução, etc.);
- Prioridade de execução;
- Informações sobre recursos;
- Informações sobre arquivos e portas de rede;
- Máscara de sinalização (quais sinais estão bloqueados);
- Proprietário do processo.



O **PID** (Process Id) é o número de identificação de um processo, atribuído na ordem em que são criados. O **PPID** é o PID do processo pai. Não há chamada de sistema que **crie um processo** no Linux, o que acontece é que um processo existente realiza seu clone - chamada de sistema **fork()** - e o processo clone pode trocar o programa por um diferente. O original é o pai e a cópia gerada é o filho.

O **UID** (User Id) é a identificação do usuário que criou e o **GID** (Group Id) é a identificação do grupo. Para definir a prioridade de agendamento (quanto tempo de CPU o processo recebe), existe um valor de **niceness** ("gentileza"). Esses valores vão de -20 a 19 (zero é o padrão), sendo -20 a maior



prioridade e 19 a menor (Cuidado! É “invertido”, assim mesmo!). No momento do clone do processo, o valor é herdado do processo pai.

O comando **nice** define a prioridade de um processo (padrão = 10) antes de executar o programa, enquanto o comando **renice** altera a prioridade de processo já em execução. O proprietário pode aumentar o valor (diminuir a prioridade), mas não pode o contrário! O root pode tudo, claro! Vamos a alguns exemplos:

\$ nice -n 5 ~/bin/teste → Reduz prioridade (eleva nice) por um fator de 5: $10 + 5 = 15$;

\$ sudo renice -6 7943 → Configura o valor de nice como -6;

\$ sudo renice 8 -u evandro → Configura o valor de nice como 8 para processos de evandro.

Como vimos anteriormente, a chamada de sistema **fork()** clona um processo, criando um filho:

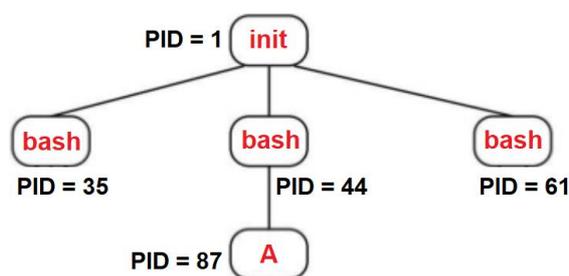
- Retorna 0 para o filho;
- O pai recebe o PID do filho.

Geralmente em seguida o filho executa um novo programa, ou seja, muda o texto do programa e os segmentos de dados e pilha são reiniciados, recebe um PID distinto e suas próprias informações contábeis. Dessa forma cada um sabe o seu papel.

Vários processos são criados na inicialização, sendo o **init** é o mais notável, sendo o responsável por:

- Executar os scripts de inicialização;
- Ser o “paizão” dos processos, pois todos são descendentes dele, exceto os criados pelo kernel;
- Chamar `exit_` quando um processo se completa (`exit` notifica o kernel de que ele está pronto para expirar, fornecendo um código - motivo, 0 = término normal);
- Receber os processos órfãos.

Em relação à **hierarquia de processos** existe apenas 1 pai e 0 ou mais filhos. Por exemplo, no Linux o processo `init` (PID = 1) é o primeiro a ser executado, logo após o carregamento do Kernel. A função dele é controlar todos os outros processos que são executados no computador. Digamos que a partir dele sejam abertos 3 *shells* (`bash`) e a partir de um deles seja executado um programa “A”. Abstraindo a existência de outros processos, a hierarquia descrita ficaria assim (PIDs inventados, com exceção do `init`):



Sinais podem ser enviados aos processos (interrupções em nível de processo) de algumas formas:



- Comunicação entre processos;
- Extinguir, interromper ou suspender processos (ex. no terminal: CTRL+C ou CTRL+Z);
- Enviados com o comando **kill** (não apenas “mata”, pode enviar outros sinais);
- Enviados pelo *kernel* (alguma infração, ex.: divisão por 0);
- Enviados pelo *kernel* (situação “interessante”, ex.: morte de processo filho).

Vamos enfatizar o seguinte, o comando **kill** envia qualquer sinal (não é porque o nome do comando é “matar” que ele serve apenas para isso, porém geralmente é o uso mais frequente). Sintaxe:

```
kill [-SINAL] PID
```

O comando **killall** tem o mesmo propósito de kill, mas é especificado o nome do processo (se houver mais de um processo com o mesmo nome, o sinal é enviado a todos):

```
killall [-SINAL] NOME_PROC
```

Alguns dos sinais são (o padrão é TERM):

#	Nome	Descrição
1	HUP	Suspender
2	INT	Interromper
3	QUIT	Abandonar
9	KILL	Matar
15	TERM	Término do software

O sinal pode ser especificado pelo nome ou pelo seu código equivalente. Por exemplo, para enviar o sinal KILL para um processo com PID 4321:

```
kill -SIGKILL 4321
```

```
kill -9 4321
```

Para enviar o sinal TERM para um processo com nome abcd:

```
killall -SIGTERM abcd
```

Os **estados dos processos** em sistemas Linux podem ser:

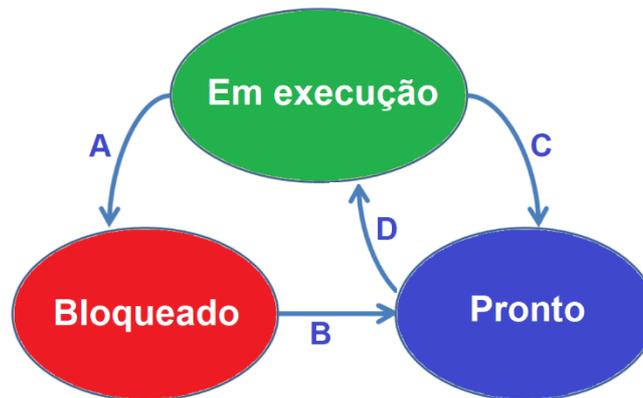
Estado	Significado
Executável	Pode ser executado (pronto).
Dormente	Aguardando algum recurso (ex.: E/S).
Zumbi	Tentando se destruir.



Parado

Suspenso (sem permissão para ser executado).

Daemons e *shells* interativos passam a maior parte do tempo “dormindo” (E/S ou conexões de rede). Vamos ver um desenho bastante utilizado em aulas de sistemas operacionais (não especificamente o Linux):



O estado executável (pronto) é quando o processo está aguardando para utilizar a CPU.

O estado dormiente (bloqueado) é quando estava em execução e fez uma chamada de sistema que não pôde ser completada imediatamente (ex.: ler um arquivo).

O estado zumbi é quando o processo terminou a execução, mas ainda não teve o seu status coletado.

O estado parado é quando está proibido de executar (sinais STOP ou TSTP, sinal CONT para reiniciar).

Para o **monitoramento de processos** existem basicamente três comandos, mas o mais conhecido é o ps. Vejamos os comandos e alguns argumentos já cobrados em provas de concurso:

- **ps**: pode exibir PID, UID, prioridade, terminal de controle de processos, memória consumida, estado atual
 - a: mostra todos os processos existentes;
 - u: exibe o nome do usuário que iniciou determinado processo e a hora que isso ocorreu;
 - x: exibe os processos que não estão associados a terminais;
 - l: exibe mais campos no resultado;
 - e: exibe as variáveis de ambiente relacionadas aos processos;
 - f: exibe a árvore de execução dos processos.
→ Visão geral sobre o sistema: ps aux
- **top**: mostra um resumo dos processos ativos e uso de recursos, atualizando regularmente (processos mais ativos aparecem no topo, padrão de atualização = 10 segundos);
- **jobs**: mostra os processos que estão parados ou rodando em segundo plano.

Processos em segundo plano são iniciados usando o símbolo “&” no final da linha de comando ou através do comando bg:

```
$ ./teste &
```



\$ bg ./teste

Quando um comando é executado em segundo plano, o terminal fica liberado para que outros comandos sejam digitados.

QUESTÕES COMENTADAS

60.(2014 - CESPE - TJ-SE)

Alguns programas podem apresentar problemas que resultem no travamento do sistema operacional, o que pode ser resolvido, no Linux, por meio do comando Kill, que finaliza o processo, funcionalidade que pode ser acessada por meio de outro terminal.

Comentário:

O comando kill serve para enviar um sinal para um processo através de seu PID. Esse sinal pode ser para finalizar um processo, entre outros. A questão não fala que o kill serve apenas para finalizar, diz que é possível finalizar um processo com ele, então está ok!

Gabarito: Certo

61.(2015 - FCC - CNMP)

O comando ps do Linux permite parâmetros (ou opções) para mostrar:

- I. os processos criados por você e de outros usuários do sistema.
- II. o nome de usuário que iniciou o processo e hora em que o processo foi iniciado.
- III. as variáveis de ambiente no momento da inicialização do processo.
- IV. a árvore de execução de comandos (comandos que são chamados por outros comandos).

Para realizar o que descrevem os itens I, II, III e IV são utilizados, correta e respectivamente, os parâmetros

- A) f / u / e / a
- B) f / a / e / u
- C) u / e / f / a
- D) e / u / a / f
- E) a / u / e / f

Comentário:

A = All, U = user, E = environment (ambiente), F = tree (esse fugiu da lógica de nomes, mas os outros dá para "adivinhar", se não souber).

Gabarito: E



62.(2016 - FCC - TRT-14ª Região)

Para listar todos os processos que estavam em execução em um computador com o sistema operacional Linux instalado, um usuário utilizou o comando ps. Para que esse comando exiba informações detalhadas de cada processo, como o nome do usuário que iniciou o processo, o número identificador do processo, a porcentagem de utilização da CPU e da memória pelo processo e a hora em que cada processo foi iniciado, este comando deve ser utilizado com o parâmetro

- A) aux.
- B) -top.
- C) vim.
- D) pstree.
- E) -zcf.

Comentário:

A = All, U = user, X = não terminal (foge um pouco da lógica dos nomes, mas os outros estão mais tranquilos).

Gabarito: A

63.(2017 - FGV - IBGE)

Em ambiente Linux, a chamada ao sistema (system call) que implementa a criação de um novo processo é:

- A) create_process;
- B) new_process;
- C) fork;
- D) spawn;
- E) duplicate.

Comentário:

Não há chamada de sistema que **crie um processo** no Linux, o que acontece é que um processo existente realiza seu clone - chamada de sistema **fork()** - e o processo clone pode trocar o programa por um diferente. O original é o pai e a cópia gerada é o filho.

Gabarito: C

64.(2018 - FGV - Prefeitura de Niterói-RJ)

Sobre o gerenciamento de processos no sistema operacional Linux, assinale a afirmativa correta.

- A) O comando ps lista os processos em execução no sistema, trazendo informações sobre o quanto de processamento ou de memória cada um deles está consumindo.



- B) O ID do processo não pode ser reutilizado depois que o processo termina.
- C) Por meio do comando `renice` é possível alterar a prioridade de execução de um processo.
- D) Por padrão, os processos executados por um usuário iniciam com nível de prioridade -20, isto é, a prioridade mais baixa possível.
- E) Processos órfãos não possuem PPID.

Comentário:

(A) O `ps` pode exibir PID, UID, prioridade, terminal de controle de processos, memória consumida, estado atual. O quanto de processamento está consumindo, não! (B) Ex.: se o processo 4000 é terminado, depois de algum tempo outro processo pode utilizar o PID 4000, sem problemas. (C) Isso aí! O `renice` altera a prioridade de um processo já em execução, enquanto o `nice` apenas atribui um valor de prioridade a um programa a ser executado. (D) O padrão é iniciar com prioridade 0 e -20 é a maior prioridade possível! A menor é 19 (isso mesmo, invertido). (E) Possuem PPID de um processo que já não existe mais. Quando isso é constatado o processo `init` assume a “paternidade” desse processo órfão.

Gabarito: C

65.(2018 - CESPE - EBSERH)

No sistema operacional Linux, é possível alterar a prioridade de um processo já iniciado com o uso do comando `nice`.

Comentário:

O `nice` é utilizado para executar um programa e o `renice` é utilizado para alterar a prioridade do processo já em execução.

Gabarito: Errado

SUPERUSUÁRIO

O superusuário é aquele que pode fazer tudo no sistema! Pode executar qualquer operação válida em qualquer arquivo ou processo. Algumas chamadas de sistema (*syscalls*) só podem ser executadas pelo *root* (nome padrão para o superusuário, mas pode ter outros).

O *root* é uma conta com **UID=0**, sendo possível modificar o nome de usuário dessa conta ou criar contas com UID=0 (ações não recomendadas, porém possíveis).

Alguns exemplos de operações restritas, que somente o *root* pode realizar:

- Modificar o diretório raiz;
- Criar arquivos de dispositivo;
- Configurar o relógio;



- Aumentar limites de uso de recursos e as prioridades de processos;
- Definir o nome do host;
- Configurar interfaces de rede;
- Abrir portas de rede privilegiadas (< 1024);
- Desligar o sistema.

Por padrão, o diretório home do root é `"/root"` e `"/sbin"` é o diretório que armazena os binários de sistema importantes que são utilizados usados pelo `root`.

Quando um usuário `"comum"` utiliza o shell, aparece `"$"` no prompt, enquanto para o `root` aparece `"#"`. Isso é importante para verificar em uma questão na prova, para saber se quem está executando é um usuário comum ou o root! Abaixo podemos ver que o root está utilizando o shell. Claro que está configurado para aparecer o nome do usuário à esquerda, mas mesmo que não estivesse, poderíamos ver o caractere `"#"`, que indica ser o root.

```
root@evandro-HP: ~/DHAV2
Arquivo Editar Ver Pesquisar Terminal Ajuda
Gerando arquivo: /media/evandro/CASOS-1TB/saida-imagem012/canal4/dir0/arq20180505175052_000000011263.h264 de tamanho: 525338
Gerando arquivo: /media/evandro/CASOS-1TB/saida-imagem012/canal7/dir0/arq20180505175044_000000011264.h264 de tamanho: 393507
Gerando arquivo: /media/evandro/CASOS-1TB/saida-imagem012/canal7/dir0/arq20180505175046_000000011265.h264 de tamanho: 92548
Processou 11266 arquivos
root@evandro-HP:~/DHAV2#
```

Para um usuário comum se tornar root, pode-se utilizar o comando `su` (*substitute user*). Note que o comando `su` serve para "substituir" um usuário e não precisa ser necessariamente o root! Mas na prática, geralmente se utiliza para se ornar o root mesmo.

O comando `su` sem argumentos solicita a senha de `root` e inicia um shell de `root`. Não registra nenhum comando executado como root, mas cria log de quem e quando se tornou root, o que é útil para auditoria de alguma coisa errada realizada no período. Para substituir por outras identidades, a sintaxe é: `su nomeUsuario` → pede a senha do `nomeUsuario`. É útil para testar em nome do `nomeUsuario`.

Se duas ou mais pessoas souberem a senha do root, pode acontecer uma situação em que alguém faz o login como root, "faz coisas que não deveria" e ninguém saberá quem foi. Por questões de segurança há a possibilidade de desabilitar o `login` de `root`.

O comando `sudo` permite a usuários comuns obter privilégios de outro usuário, em geral o root, para executar tarefas específicas dentro do sistema de maneira segura e controlável pelo administrador. Ao ser executado, há uma consulta ao arquivo `/etc/sudoers`, que possui os usuários e comandos habilitados em cada host. É solicitada a senha do próprio usuário que executou o `sudo` e, na sequência, é executado o comando que se encontra após o `sudo`. Importante salientar que é mantido um log dos comandos executados, hosts, usuários, diretório, data/hora etc. O exemplo mais comum é sem especificar um usuário:

```
$ sudo apt-get install abc
```



No exemplo acima, como não é especificado um usuário, então o “apt-get install abc” será executado como root, após o usuário ter digitado a sua senha. Pode haver configuração de tempo sem ter que digitar a senha novamente (até 5 minutos). Vamos ver mais dois exemplos:

\$ sudo -u evandro /home/evandro/prog → executa o “prog” em nome de “evandro”.

\$ sudo -g prof /home/teste/prog → seta o grupo primário para “prof” no momento da execução de “prog”.

QUESTÕES COMENTADAS

66.(FCM/IF Sudeste-MG - 2016)

O Linux permite ao superusuário (root) executar qualquer operação válida em qualquer arquivo ou processo, inclusive algumas chamadas de sistema podem ser executadas somente pelo superusuário. A operação que NÃO é exclusiva do superusuário é

- A) configurar interfaces de rede.
- B) configurar o relógio do sistema.
- C) alterar as permissões de um arquivo.
- D) aumentar os limites de uso dos recursos.
- E) definir o nome de host (hostname) do sistema.

Comentário:

De todas as alternativas mostradas, sabemos que alterar as permissões de um arquivo qualquer um pode fazer, não é? Claro que qualquer um pode alterar as permissões (comando chmod) do seu arquivo e não de outros(a não ser que tenha permissão para isso).

Gabarito: C

67.(IBFC/EBSERH - 2016)

O comando sudo do sistema operacional Linux é muito poderoso, permitindo que usuários comuns obtenham privilégios de super usuário. Por questões de segurança, o administrador precisa definir no arquivo _____, quais usuários podem executar sudo, em quais computadores podem fazê-lo e quais comandos podem executar através dele. Assinale a alternativa que complete correta e respectivamente a lacuna:

- A) /etc/sudoers
- B) /root/sudouser
- C) /root/sudoers
- D) /usr/sudouser
- E) /etc/sudouser



Comentário:

Pense o seguinte: os arquivos de configuração geralmente ficam no diretório “/etc”. E para executar o sudo existe um arquivo com as configurações de quem e do que pode ser executado com o sudo, que é o arquivo “sudoers”. Em inglês, quem caminha (“walk”) é um “walker”, então quem faz um “sudo” é um “sudoer”, assim fica mais fácil lembrar 😊.

Gabarito: A

68.(INAZ do Pará/DPE-PR - 2017)

Qual a alternativa que corresponde à linha de comando para que usuários comuns possam utilizar o comando administrativo apt-get dist-upgrade?

- A) # apt-get dist-upgrade
- B) \$ sudo apt-get gw dist-upgrade
- C) # sudo apt-get gw up dist-upgrade
- D) \$ sudo apt-get dist-upgrade
- E) # sudo apt-get dist-upgrade gw now

Comentário:

Usuário comum tem o prompt “\$”, então temos duas alternativas possíveis. Para executar o “apt-get dist-upgrade” com o “sudo” é só colocar o “sudo” na frente.

Gabarito: D

69.(CESPE/ABIN - 2018)

O Linux não impede a alteração do nome do superusuário, nem a criação de contas com UID igual a 0, embora essas ações não sejam recomendadas.

Comentário:

Como vimos na aula, não é recomendado fazer isso, mas é possível alterar o nome do root para qualquer coisa, e também é possível criar novas contas de superusuário (possuem UID = 0).

Gabarito: Certo

70.(COSEAC/UFF - 2019)

No Sistema Operacional Linux o diretório local padrão do superusuário é:

- A) /home.
- B) /usr.
- C) /root.



D) /lib.

E) /dev.

Comentário:

/home – home dos usuários “comuns”.

/usr – onde a maioria dos programas ficam instalados, executáveis e bibliotecas de todos os principais programas.

/root – é o home do root!

/lib – bibliotecas necessárias para os binários essenciais para as pastas /bin e /sbin.

/dev – links para dispositivos de hardware.

Gabarito: C

LISTA DE QUESTÕES

1. (2016 - AOCP - EBSERH)

O NFS (Network File System) permite compartilhar sistemas de arquivos entre computadores conectados em rede e pode ser parte fundamental da infraestrutura da tecnologia da informação. Sobre NFS, analise as assertivas a seguir e assinale a alternativa que aponta a(s) correta(s).

I. O NFS é considerado sem estado (stateless) e, portanto, quando um servidor NFS volta a funcionar, o estado anterior é restaurado e a informação não é perdida.

II. O NFS pode ser implementado do lado servidor e do lado cliente.

III. O NFS roda sobre o protocolo RPC (Remote Procedure Call), que define um modo independente do sistema para processos se comunicarem através de uma rede de computadores.

IV. O NFS suporta apenas UDP como protocolo de transporte, pois ele apresenta desempenho significativamente melhor que o TCP em redes locais.

A) Apenas I e II.

B) Apenas II e IV.

C) Apenas III.

D) Apenas III e I.

E) Apenas I, II e III.



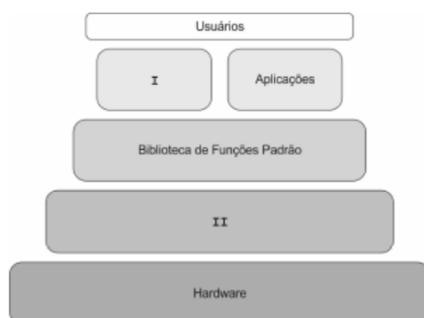
2. (2017 - FGV - IBGE)

Em ambiente Linux, a chamada ao sistema (system call) que implementa a criação de um novo processo é:

- A) create_process;
- B) new_process;
- C) fork;
- D) spawn;
- E) duplicate.

3. (2017 - FCC - TRF-5ª Região)

Considere a figura abaixo que mostra a arquitetura do sistema operacional Linux



A caixa

A) I representa a camada responsável pela interface entre o hardware e as aplicações. Dentre suas funções encontram-se gerenciamento de I/O, manutenção do sistema de arquivos, gerenciamento de memória e swapping, controle da fila de processos, etc.

B) II representa a camada que permite o acesso a recursos através da execução de chamadas feitas por processos. Tais chamadas são geradas por funções padrão suportadas pelo kernel. Dentre suas funções estão habilitar funções padrão como open, read, write e close e manter a comunicação entre as aplicações e o kernel.

C) I é um processo que executa funções de leitura de comandos de entrada de um terminal, interpreta-os e gera novos processos, sempre que requisitados. É conhecido também como interpretador de comandos.

D) II é um processo que realiza modificações no shell, permitindo que funcionalidades do Linux sejam habilitadas ou desabilitadas, conforme a necessidade. Tal processo gera ganho de performance, pois à medida que customiza o shell, o usuário torna o Linux enxuto e adaptável.

E) I é um processo que realiza modificações no kernel, permitindo que funcionalidades do Linux sejam habilitadas ou desabilitadas, conforme a necessidade. Tal processo gera ganho de performance, pois à medida que customiza o kernel, o usuário torna o Linux enxuto e adaptável.



4. (2017 - CS-UFG - DEMAÉ)

Na maior parte das distribuições Linux, a memória virtual é uma partição denominada

- A) EXT
- B) SWAP
- C) SETUP
- D) BIOS

5. (2017 - Quadrix - COFECI)

Com relação à administração de sistemas Windows e Unix/Linux, julgue o item subsequente.

Não é possível realizar uma instalação automática, por meio de uma rede, no sistema operacional Linux.

6. (2017 - Colégio Pedro II - Colégio Pedro II)

Sistema operacional é um conjunto de programas básicos e utilitários que fazem seu computador funcionar. O Debian é um sistema operacional, em cujo núcleo está o Kernel, o programa mais fundamental no computador e que faz todas as operações mais básicas, permitindo que você execute os outros programas. O Debian atualmente usa o Kernel Linux. Mas um sistema operacional não funciona somente com o Kernel, são necessários utilitários e aplicativos. O Debian utiliza estas ferramentas do projeto GNU. Por esse motivo, muitos utilizadores defendem que devemos chamar o sistema de "Debian GNU/Linux". Reconheça (1) o gerenciador de pacotes (programas) usado no Debian e em distribuições derivadas do Debian (Ubuntu, Knoppix, Big Linux etc.) que utiliza uma lista de dependências para instalar tudo o mais automaticamente possível, e (2) o arquivo de configuração utilizado por este gerenciador de pacote.

- A) Gerenciador (1) - apt-get
Arquivo (2) - source.list
- B) Gerenciador (1) - apt-get
Arquivo (2) - package.cfg
- C) Gerenciador (1) - make install
Arquivo (2) - package.cfg
- D) Gerenciador (1) - dpkg
Arquivo (2) - source.list

7. (2018 - CESPE - EMAP)

O kernel do sistema operacional Linux tem a função de interpretar os comandos executados em um terminal.



8. (2018 - FGV - MPE-AL)

Instalar, atualizar e remover pacotes no sistema operacional CentOS 7 é uma tarefa frequente para desenvolvedores de sistemas. Por isso, eventualmente podem ocorrer dúvidas sobre se determinado pacote está instalado ou qual é a versão que está sendo utilizada. Para dirimir essas dúvidas, sobre o pacote `httpd` devemos utilizar o comando

- A) `rpm -q httpd`
- B) `yum check httpd`
- C) `apt-get find httpd`
- D) `find -iname httpd`
- E) `crontab -l httpd`

9. (2018 - FGV - COMPESA)

Com relação às características e tarefas de administração do sistema operacional Linux, analise as afirmativas a seguir.

I. O esquema de partição GPT (Guid Partition Table) oferece a possibilidade de até 128 partições primárias, com tamanhos maiores do que 2TB cada. II. Ext4, XFS e exFAT são exemplos de sistemas de arquivo com jornal, utilizados pelo Linux. III. A área de swap no Linux pode ser uma partição dedicada de swap, um arquivo de swap ou uma combinação de arquivos e partições de swap.

Está correto o que se afirma em

- A) I, somente.
- B) II, somente.
- C) III, somente.
- D) I e III, somente.
- E) I, II e III.

10. (2018 - CS-UFG - SANEAGO-GO)

Um utilitário GNU, presente em diversos sistemas operacionais linux, que lista as partições dos discos rígidos, é:

- A) `fdisk`.
- B) `mcdisk`.
- C) `parted`.
- D) `gedit`.



11.(2018 - IF-RS - IF-RS)

São tipos de sistemas de arquivos utilizados no Linux:

- A) FAT 16, FAT 32, NTFS
- B) FAT 32, HFS, ExFAT
- C) Ext2, Ext3, Btrfs
- D) Ext2, Ext3, HFS
- E) Ext2, Ext3, NFS, SMB

12.(2018 - COPESE-UFT - UFT)

Embora seja possível realizar boot de um sistema Linux a partir de um pendrive, a maioria das instalações do Linux o realiza a partir do disco rígido do computador. Esse processo consiste em duas fases básicas:

1. Executar o carregador de boot a partir do dispositivo de boot; 2. Iniciar o kernel do Linux e iniciar os processos.

Assinale a alternativa que contém um gerenciador de boot para sistemas Linux:

- A) CISC (Complex Instruction Set Computer).
- B) GRUB (Grand Unified Bootloader).
- C) ORM (Object Relational Mapping).
- D) CMS (Content Management System).

13.(2018 - COPESE-UFT - UFT)

Sistemas operacionais Linux permitem logins simultâneos de diferentes usuários. Assinale a alternativa que contém o comando a ser utilizado para visualizar os usuários logados em um determinado instante.

- A) listusers
- B) ldconfig
- C) who
- D) uname

14.(2018 - IF-RS - IF-RS)

Considerando que se está logado como "root" no terminal de uma estação de trabalho com um sistema operacional Linux debian ou derivado, qual comando configura o DNS para 8.8.8.8?

- A) dns 8.8.8.8



- B) `echo 8.8.8.8 > /etc/hosts.conf`
- C) `echo nameserver 8.8.8.8 > /etc/resolv.conf`
- D) `nslookup 8.8.8.8`
- E) `nameserver 8.8.8.8`

15.(2018 - COPEVE-UFAL - UFAL)

O assistente de tecnologia da informação precisa atualizar o sistema operacional GNU/Linux do servidor WEB de uma empresa. Qual comando, usando o modo root, deve ser utilizado?

- A) `upgrade distro-all`.
- B) `upgrade apt-distro`.
- C) `apt-get dist-upgrade`.
- D) `apt-get purge "nome da distro"`.
- E) `apt-cache showpkg "nome da distro"`.

16.(2018 - COPEVE-UFAL - UFAL)

Dadas as afirmativas quanto aos conceitos de Kernel no sistema operacional GNU/Linux,

- I. Compilar o Kernel permite ao usuário remover drivers inúteis diminuindo o tempo de arranque do sistema operacional.
- II. Os módulos do Kernel do sistema estão armazenados no diretório `/boot/`.
- III. O comando `uname -a` exibe as informações do kernel do sistema.
- IV. Módulos são as partes do kernel que são carregadas somente quando são requisitadas por um aplicativo ou dispositivo.

Verifica-se que está(ão) correta(s)

- A) II, apenas.
- B) I e II, apenas.
- C) III e IV, apenas.
- D) I, III e IV, apenas.
- E) I, II, III e IV.

17.(2018 - FGV - AL-RO)

Wallace é administrador de um servidor com sistema operacional CentOS e deseja compartilhar um diretório com os integrantes da empresa por meio do protocolo NFS.



Para que a configuração desse compartilhamento possa ser efetuada, Wallace deve editar o arquivo /etc/

- A) services
- B) modules
- C) hosts
- D) export
- E) fstab

18.(2018 - COPEVE-UFAL - UFAL)

Analise a configuração de rede no GNU/Linux.

```
auto lo iface
```

```
lo inet loopback
```

```
auto eth0
```

```
allow-hotplug eth0
```

```
iface eth0
```

```
inet static address 192.168.100.1 netmask 255.255.255.0 network 192.168.100.0 broadcast 192.168.100.255
```

Dadas as afirmativas quanto à configuração de rede,

- I. O “iface eth0 inet static” informa ao sistema que a placa de rede terá o endereço IP estático.
- II. O “iface lo” informa ao sistema que a placa de rede receberá um endereço IP via servidor DHCP.
- III. Uma segunda placa de rede com fio instalada nesse computador irá receber a denominação eth1.
- IV. O “allow-hotplug” permite conectar dispositivos hotplug no computador.

verifica-se que está(ão) correta(s)

- A) II, apenas.
- B) I e III, apenas.
- C) II e IV, apenas.
- D) I, III e IV, apenas.
- E) I, II, III e IV.

19.(2018 - FGV - AL-RO)

Roberto trabalha como administrador de redes em uma empresa de cosméticos e sua principal função é a configuração de estações de trabalho dos seus colaboradores. Rotineiramente, Roberto instala e remove pacotes nas estações de trabalho às quais ele dá suporte.



Assinale a opção que indica o comando utilizado por Roberto para desinstalar pacotes RPM nas estações de trabalho com sistema operacional Linux CentOS.

- A) rpm -Uvh pacote
- B) rpm -e pacote
- C) rpm -ivh pacote
- D) rpm -qi pacote
- E) rpm -qa | grep pacote

20.(2018 - Quadrix - CFBio)

Carregamento e inicialização do kernel e execução dos scripts de inicialização do sistema são algumas das etapas do processo de inicialização de um Linux típico.

21.(2018 - Quadrix - CRM-PR)

Um computador pessoal com sistema operacional Linux, versão desktop, pode ter no máximo cinco usuários simultâneos.

22.(2018 - FADESP - IF-PA)

Sobre os pacotes do Sistema Operacional (SO) Debian 9.5, é correto afirmar que

- A) o apt é um programa que pode substituir o dpkg para o gerenciamento de pacotes do SO.
- B) os pacotes devem ser distribuídos no formato .tar.gz.
- C) o dpkg-source compila e executa código fonte de um pacote.
- D) um pacote marcado como Essential não pode ser desinstalado do Sistema Operacional.
- E) cada pacote deve especificar as dependências sobre outros pacotes para seu funcionamento.

23.(2018 - CONSULPAM - Câmara de Juiz de Fora-MG)

Em relação ao Sistema Operacional Linux, marque o item INCORRETO:

- A) Sua arquitetura é composta por um núcleo monolítico cujas funções são: gerenciar a memória, operar as entradas e saídas e o acesso aos arquivos.
- B) Outra característica do Linux é com relação aos drivers de dispositivos e suporte a rede, os quais podem ser compactadas e utilizadas como se fossem módulos ou bibliotecas (LKM em inglês Loadable Kernel Modules), separados pela parte principal, cujo carregamento pode ser ativado após a execução do núcleo.



C) No quesito portabilidade, o Linux funciona com eficiência em plataformas como x64 da Intel (EM64T e AMD64) PowerPC, Alpha, SPARC, porém é de difícil instalação nos sistemas embarcados como PVR, celulares, Tv's e Handhelds.

D) A partir da década de 90, ao passo que a distribuição do Linux se popularizou, foi também limitada, pois se torna uma alternativa no uso de software livre, contra os sistemas operacionais da Apple (Mac OS) e Microsoft (Windows).

24.(2018 - CESPE - FUB)

O Linux é um sistema operacional em que cada usuário consegue ter apenas um processo ativo por vez, processo esse que é iniciado automaticamente quando o sistema é carregado.

25.(2018 - COPESE - UFT)

O sistema operacional Linux é reconhecido por permitir diversos níveis de personalização, inclusive de suportar o uso de vários ambientes gráficos. Assinale a alternativa que NÃO constituiu uma interface gráfica usada no Linux

- A) Pantheon.
- B) XFCM.
- C) MATE Desktop.
- D) Cinnamon Desktop.

26.(2018 - CCV-UFC - UFC)

De acordo com a linha do arquivo '/ etc / passwd' exibida abaixo, qual das seguintes afirmações é verdadeira?

```
jferreira:x:502:1000:Joao Ferreira:/home/jferreira:/bin/bash
```

- A) O usuário Joao Ferreira possui a senha 'x'.
- B) Shadow passwords (senhas ocultas) são utilizados no sistema.
- C) O usuário Joao Ferreira pertence ao grupo com groupID 502.
- D) Membros do groupID 502 podem ler o diretório /home/jferreira.
- E) O nome de usuário (username) 'jferreira' pertence ao grupo 'jferreira'.

27.(2018 - Quadrix - CRM-PR)

É possível instalar, em um mesmo computador desktop, os sistemas operacionais Windows 7 e Linux, de forma independente.



28.(2018 - Quadrix - CRM-PR)

O usuário root do sistema Linux tem autonomia para acessar arquivos de outros usuários, com exceção dos usuários administradores.

29.(2018 - IDECAN - CRF-SP)

“O arquivo _____ só é legível pelo superusuário e serve para manter senhas criptografadas protegidas contra o acesso não autorizado. Ele também fornece informações sobre contas que não são disponíveis em _____.” Assinale a alternativa que completa correta e sequencialmente a afirmativa sobre o Sistema Operacional Linux.

- A) /etc/passwd; /etc/group
- B) /etc/group; /etc/passwd
- C) /etc/passwd; etc/shadow
- D) /etc/shadow; /etc/passwd

30.(2018 - UERR - IPERON)

Um administrador de uma rede baseada em Linux deseja guardar no servidor os scripts especiais para iniciar ou interromper módulos e programas diversos. Para isso, ele precisa guardar essas informações dentro do diretório:

- A) /srv
- B) /opt
- C) /etc
- D) /lib
- E) home

31.(2018 - UERR - IPERON)

Em um computador com Linux, deseja-se instalar o sistema de arquivos ext3, mas com o nível de Journaling no qual se grava mudanças em arquivos de metadados, que força que a escrita do conteúdo dos arquivos seja feita após a marcação de seus metadados. Esse nível de Journaling é denominado:

- A) ordered
- B) cshell.
- C) iso9660
- D) journal.
- E) fsreiser.



32.(2019 - CS-UFG - IF Goiano)

Os sistemas Linux buscam padronizar diretórios para a localização de arquivos. Comandos essenciais do sistema operacional, tais como cat, tar, su, rm e pwd, em geral, estão localizados nos seguintes diretórios:

- A) /lib e /var
- B) /usr/lib e /lib
- C) /bin e /usr/bin
- D) /usr/bin e /var

33.(2019 - FCC - SEFAZ-BA)

Um Auditor Fiscal da área de Tecnologia da Informação deseja desinstalar um pacote chamado java-1.6.0-openjdk.x86_64 em linha de comando, como usuário root, no Red Hat Enterprise Linux 6. Para isso, terá que utilizar o comando

- A) apt-get uninstall java-1.6.0-openjdk.x86_64
- B) apt-get remove -rf java-1.6.0-openjdk.x86_64
- C) rm -rf java-1.6.0-openjdk.x86_64
- D) yum remove java-1.6.0-openjdk.x86_64
- E) apt-get -rf java-1.6.0-openjdk.x86_64

34.(2019 - IF-MS - IF-MS)

O sistema operacional Linux possui várias partições (áreas) em sua estrutura, cada uma com uma função definida. Assinale a alternativa que apresenta a partição que abriga a pasta raiz do sistema que contém arquivos essenciais ao seu pleno funcionamento:

- A) /boot
- B) /bin
- C) /
- D) /etc
- E) /root

35.(2019 - VUNESP - Câmara de Sertãozinho-SP)

Considere que o usuário de um computador com sistema operacional Linux deseja executar um comando em segundo plano (background) em um terminal Shell. Para efetivar essa ação, o usuário deve executar o comando



- A) e teclar a combinação Ctrl+c
- B) e teclar a combinação Ctrl+x
- C) seguido de bg
- D) seguido de &
- E) seguido de !

36.(2019 - COSEAC - UFF)

No Sistema Operacional Linux o diretório local padrão do superusuário é:

- A) /home.
- B) /usr.
- C) /root.
- D) /lib.
- E) /dev.

37.(2019 - CESPE - SLU-DF)

Windows e Linux são classificados como sistemas operacionais de tempo real crítico, porque fornecem garantias absolutas de que todas as suas ações ocorrerão dentro de intervalos de tempo determinados.

38.(2019 - INAZ do Pará - CORE-SP)

“Todos os arquivos e diretórios do sistema Linux instalado no computador partem de uma única origem: o diretório raiz. Mesmo que estejam armazenados em outros dispositivos físicos.”

Disponível em: <https://canaltech.com.br/linux/entendendo-a-estrutura-de-diretorios-dolinux/>.
Acesso em: 13.12.2018

Qual pasta no sistema operacional LINUX tem a função de armazenar arquivos de dispositivos do sistema?

- A) /dev.
- B) /etc.
- C) /lib.
- D) /var.
- E) /proc.



39.(2019 - INAZ do Pará - CORE-SP)

“O agendamento de tarefas é um recurso muito interessante para a administração de sistemas operacionais. É possível programar a execução de scripts de manutenção do sistema, disparar envio de newsletters, gerar relatórios de análises de logs, entre outros”

Disponível em: [https://www.vivaolinux.com.br/dica/Agendamento-de-tarefas-no-Linux- \(cron-e-at\)](https://www.vivaolinux.com.br/dica/Agendamento-de-tarefas-no-Linux-(cron-e-at)). Acesso em: 13.12.2018

Qual programa do sistema operacional LINUX executa tarefas de modo automático, em horários pré-determinados, e executa tarefas que não foram executadas, enquanto o sistema esteve desligado?

- A) Fcron.
- B) Cron.
- C) Anacron.
- D) Schedule.
- E) Crontab.

40.(2019 - IDECAN - IF-PB)

O Debian, uma das distribuições do sistema operacional Linux, possui um conjunto de pastas com nome pré-estabelecidos. Essas pastas possuem significado específico para o sistema operacional, visando atender necessidades do mesmo. Assinale a alternativa que indica corretamente o nome da pasta que armazena arquivos referentes às instalações de programas não oficiais da distribuição do sistema operacional.

- A) /bin
- B) /lib
- C) /opt
- D) /var
- E) /proc

41.(2019 - UFRR - UFRR)

A respeito de diretórios no sistema operacional Linux, a estrutura é conhecida como árvore invertida, isto é a raiz da árvore de diretórios é o topo. Qual é a representação do diretório raiz, e nome do usuário que possui privilégio para escrever neste diretório.

- A) (raiz) e usuário root
- B) \\ (raiz) e usuário administrador
- C) / (raiz) e usuário root
- D) [(raiz) e usuário administrador



E)]/ (raiz) e usuário root

42.(2019 - IF-ES - IF-ES)

O Network File System (NFS) – Sistemas de Arquivos em Rede – tem como um dos principais propósitos dar suporte a um sistema heterogêneo, no qual clientes e servidores estejam possivelmente executando sistemas operacionais e hardwares diferentes. Sobre o NFS, é CORRETO afirmar:

- A) O NFS utiliza dois protocolos cliente-servidor, em que o primeiro é responsável pela montagem e o segundo é para acesso de diretório e arquivos.
- B) O servidor tem total gerência sobre o ponto de montagem nos clientes.
- C) Os serviços NFS são implementados apenas nos servidores Linux.
- D) Como critério de segurança, os clientes não podem ter acesso aos atributos dos arquivos.
- E) O NFS faz uso de máquinas distintas para servidores e clientes, impossibilitando que a mesma máquina seja tanto cliente quanto servidor

43.(2019 - FCC - Prefeitura de Manaus-AM)

Em computador com sistema operacional Linux e o OpenLDAP instalado, o programador editou o arquivo ldap.conf para configurar

- A) a autenticação do servidor.
- B) a execução do serviço do LDAP.
- C) o banco de dados a ser utilizado.
- D) a geração de índices do banco de dados.
- E) o acesso dos clientes ao diretório.

44.(2014 - FUNCAB - MDA)

No sistema operacional Linux, a criação de um grupo permite que um conjunto de usuários diferentes possua acesso a um mesmo arquivo. O comando do Linux que tem como objetivo alterar o grupo de um arquivo/diretório é o

- A) chown.
- B) chgroup.
- C) chset.



- D) chfile.
- E) chgrp.

45.(2018 - FGV - Prefeitura de Niterói-RJ)

Pedro é o proprietário do arquivo header.txt em um sistema Linux e gostaria de assegurar que somente ele tivesse permissão de leitura, gravação e execução a este arquivo, enquanto que todos os demais usuários com acesso ao sistema tivessem somente a permissão de leitura.

Assinale a opção que indica o comando que pode ser usado para conseguir esse objetivo.

- A) chmod ug+r header.txt
- B) chmod 766 header.txt
- C) chmod 722 header.txt
- D) chmod +r header.txt
- E) chmod 744 header.txt

46.(2018 - FGV - MPE-AL)

Para definir o script `/usr/local/bin/meuscript` como executável no sistema operacional Linux, devemos usar o comando

- A) chmod og /usr/local/bin/meuscript
- B) chmod 000 /usr/local/bin/meuscript
- C) chmod +x /usr/local/bin/meuscript
- D) chmod ugo-x /usr/local/bin/meuscript
- E) chmod 666 /usr/local/bin/meuscript

47.(2019 - IF-ES - IF-ES)

Os administradores de redes precisam entender com clareza sobre o gerenciamento de contas de usuários nos Sistemas Operacionais de Servidores Linux. Em relação ao sistema de contas Linux, analise as afirmativas abaixo:

- I – O arquivo `/etc/shadow` serve para manter senhas criptografadas seguras quanto a acesso não autorizado.
- II – Um novo usuário pode ser adicionado utilizando-se ferramentas, como `adduser` ou `useradd`.
- III – Os usuários dos sistemas Linux recebem um número UID referente a sua identificação, porém o usuário `root`, por questões de segurança, não dispõe de UID.

É CORRETO o que se afirma em:



- A) I, II e III.
- B) I e II, apenas
- C) I e III, apenas.
- D) II e III, apenas.
- E) Nenhuma das afirmativas está correta.

48.(2019 - CS-UFG - IF Goiano)

Na configuração do Sistema Linux, o comando `chmod` modifica as permissões de arquivos com respeito à execução, leitura e escrita. A expressão `chmod 751 arq2` é equivalente a:

- A) `chmod u=rx,g=rwx,o=x arq2`
- B) `chmod u=rx,g=rwx,o=r arq2`
- C) `chmod u=rwx,g=rx,o=x arq2`
- D) `chmod u=rwx,g=rx,o=r arq2`

49.(2019 - CCV-UFC - UFC)

Qual dos itens abaixo contém o comando do sistema operacional Linux capaz de mudar o grupo de um arquivo ou diretório do sistema?

- A) `df`
- B) `sed`
- C) `chown`
- D) `chmod`
- E) `passwd`

50.(2019 - FCC - TJ-MA)

Um Analista digitou o comando `chmod u=rwx,g=rx,o=r processo` para definir as permissões de acesso ao arquivo `processo`. O comando equivalente usando a notação octal é:

- A) `chmod 713 processo`
- B) `chmod 777 processo`
- C) `chmod 134 processo`
- D) `chmod 754 processo`
- E) `chmod 671 processo`



51.(2019 - VUNESP - Câmara de Tatuí-SP)

Considere a execução da sequência de comandos a seguir, em um terminal shell do Linux:

```
# cd /root
```

```
# mkdir -p /root/foo/bar
```

```
# pwd
```

O resultado impresso na tela após a execução do último comando será:

A) /root

B) /root/foo/bar

C) bar

D) /root/foo

E) /foo/bar

52.(2019 - CS-UFG - IF Goiano)

No sistema operacional GNU/Linux, usando a linha de comando, deseja-se executar o seguinte: a) criar um diretório chamado IFG; b) criar um arquivo de texto chamado Concurso.txt; c) apagar o arquivo de texto Concurso.txt; d) apagar o diretório IFG. Qual é a sequência de comandos a ser empregada?

Obs.: o sinal “,” é um mero separador entre os comandos

A) mkdir IFG , touch Concurso.txt , rm Concurso.txt , rmdir IFG

B) mkdir IFG , create Concurso.txt , kill Concurso.txt , killdir IFG

C) create IFG , touch Concurso.txt , rm Concurso.txt , rmdir IFG

D) create IFG , create Concurso.txt , kill Concurso.txt , killdir IFG

53.(2011 - CESGRANRIO - Petrobras)

Em um sistema Unix, um arquivo de script chamado teste.sh foi copiado para o diretório /tmp. No shell do sistema, o usuário submeteu dois comandos: cd /tmp e teste.sh. Após a execução do segundo comando, o shell informou uma mensagem de erro, indicando comando não encontrado. O que deve ser feito para corrigir o problema que gerou essa mensagem?

A) Certificar que o usuário não entrou com letras maiúsculas.

B) Certificar que o arquivo tem permissão para ser executado.

C) Omitir a extensão .sh ao entrar com o nome do script.

D) Incluir ./ antes do nome do script.

E) Mover o arquivo para o diretório home e executá-lo.



54.(2012 - CESGRANRIO - Petrobras)

No ambiente UNIX, existem vários interpretadores de linha de comando conhecidos como shell. É importante, para cada script, informar em que shell ele deve ser executado.

Para isso, o usuário pode especificar o shell desejado

- A) na primeira linha do script.
- B) na última linha do script.
- C) em qualquer linha do script.
- D) em um arquivo à parte.
- E) na linha de comando, após o nome do arquivo que contém o script.

55.(2014 - CEPERJ - Rio previdência)

Shell script é uma linguagem de script para Linux, nada mais do que comandos do próprio Linux que são executados em uma determinada sequência para uma determinada finalidade. Nesse contexto, duas situações são listadas a seguir.

I- No terminal ou modo gráfico, deseja-se criar um arquivo que possa ser editado para que se torne o primeiro shell script a ser criado, sendo necessário utilizar um comando CMD1.

II- Para que seja possível executar o shell script criado, é preciso atribuir a este o direito de execução; para isso é necessário usar um comando CMD2.

Exemplos de CMD1 e de CMD2 são, respectivamente:

- A) touch shell1.sh e exec +x shell1.sh
- B) touch shell1.sh e chmod +x shell1.sh
- C) create shell1.sh e chmod +x shell1.sh
- D) new shell1.sh e chmod +x shell1.sh
- E) new shell1.sh e exec +x shell1.sh

56.(2014 - FUMARC - AL-MG)

No Linux, considere a existência de um arquivo “arquivo.txt” no diretório corrente, com o seguinte conteúdo:

linha 1 valor 10 linha 2 valor 20 linha 3 valor 30

Deseja-se produzir, a partir do arquivo, a seguinte saída:

1 - 10 2 - 20 3 - 30



O script shell usado com comandos AWK que produz a saída desejada é:

- A) `awk arquivo.txt '{ print $2" - "$4 }'`
- B) `cat arquivo.txt | awk '{ print $2" - "$4 }'`
- C) `cat arquivo.txt | awk '{ print $1" - "$3"\n" }'`
- D) `cat arquivo.txt | awk '{ printf($2" - "$4) }'`

57.(2015 - VUNESP - CRO-SP)

No sistema operacional UNIX, uma aplicação pode ser desenvolvida por meio de um script que é um conjunto de comandos que podem ser executados pelo interpretador de comandos (shell). Considerando a existência do interpretador `/bin/sh`, a primeira linha de um arquivo de script deve conter:

- A) `* /bin/ sh`
- B) `$ /bin/ sh`
- C) `& /bin/ sh`
- D) `#! /bin/ sh`
- E) `#* /bin/ sh`

58.(2016 - INSTITUTO AOCP - EBSERH)

Em sistemas operacionais Linux é possível criar scripts para automatizar tarefas rotineiras. A extensão de arquivo utilizada para Shell Script é

- A) `.bin`
- B) `.sl`
- C) `.bash`
- D) `.sh`
- E) `.gz`

59.(2016 - FCC - ELETROBRAS-ELETROSUL)

Um profissional de TI está usando um computador com sistema operacional Linux que utiliza no shell o interpretador de comandos `bash`. Ele está logado como usuário teste e criou o seguinte arquivo shell script:

- 1 `#!/bin/bash`
- 2 `echo 'Eletrosul- Centrais Elétricas S.A.'`



3 \$ variavel= 'Eu estou logado como usuário \$user'

4 \$ echo \$variavel

Considerando que 1, 2, 3 e 4 indicam as linhas do arquivo e que este tenha sido salvo com o nome exemplo, é correto afirmar:

A) Para o arquivo ser executável, é necessário acionar o comando \$ chmod +x exemplo. Depois disto o arquivo poderá ser executado com ./exemplo.

B) A linha 1 indica que todas as outras linhas abaixo deverão ser executadas pelo compilador sh, que se localiza em /bin/bash.

C) Após ser executado, o arquivo imprimirá na tela apenas frase “Eletrosul – Centrais Elétricas S. A.” utilizando o comando echo.

D) Ao acionar o comando file arquivo é possível ver que a definição dele é Bourne-Again Shell Script, que se refere ao bash script.

E) As linhas 3 e 4 farão com que seja impresso na tela Eu estou logado como usuário \$teste.

60.(2014 - CESPE - TJ-SE)

Alguns programas podem apresentar problemas que resultem no travamento do sistema operacional, o que pode ser resolvido, no Linux, por meio do comando Kill, que finaliza o processo, funcionalidade que pode ser acessada por meio de outro terminal.

61.(2015 - FCC - CNMP)

O comando ps do Linux permite parâmetros (ou opções) para mostrar:

I. os processos criados por você e de outros usuários do sistema.

II. o nome de usuário que iniciou o processo e hora em que o processo foi iniciado.

III. as variáveis de ambiente no momento da inicialização do processo.

IV. a árvore de execução de comandos (comandos que são chamados por outros comandos).

Para realizar o que descrevem os itens I, II, III e IV são utilizados, correta e respectivamente, os parâmetros

A) f / u / e / a

B) f / a / e / u

C) u / e / f / a

D) e / u / a / f

E) a / u / e / f



62.(2016 - FCC - TRT-14ª Região)

Para listar todos os processos que estavam em execução em um computador com o sistema operacional Linux instalado, um usuário utilizou o comando ps. Para que esse comando exiba informações detalhadas de cada processo, como o nome do usuário que iniciou o processo, o número identificador do processo, a porcentagem de utilização da CPU e da memória pelo processo e a hora em que cada processo foi iniciado, este comando deve ser utilizado com o parâmetro

- A) aux.
- B) -top.
- C) vim.
- D) pstree.
- E) -zcf.

63.(2017 - FGV - IBGE)

Em ambiente Linux, a chamada ao sistema (system call) que implementa a criação de um novo processo é:

- A) create_process;
- B) new_process;
- C) fork;
- D) spawn;
- E) duplicate.

64.(2018 - FGV - Prefeitura de Niterói-RJ)

Sobre o gerenciamento de processos no sistema operacional Linux, assinale a afirmativa correta.

- A) O comando ps lista os processos em execução no sistema, trazendo informações sobre o quanto de processamento ou de memória cada um deles está consumindo.
- B) O ID do processo não pode ser reutilizado depois que o processo termina.
- C) Por meio do comando renice é possível alterar a prioridade de execução de um processo.
- D) Por padrão, os processos executados por um usuário iniciam com nível de prioridade -20, isto é, a prioridade mais baixa possível.
- E) Processos órfãos não possuem PPID.



65.(2018 - CESPE - EBSERH)

No sistema operacional Linux, é possível alterar a prioridade de um processo já iniciado com o uso do comando nice.

66.(FCM/IF Sudeste-MG - 2016)

O Linux permite ao superusuário (root) executar qualquer operação válida em qualquer arquivo ou processo, inclusive algumas chamadas de sistema podem ser executadas somente pelo superusuário. A operação que NÃO é exclusiva do superusuário é

- A) configurar interfaces de rede.
 - B) configurar o relógio do sistema.
 - C) alterar as permissões de um arquivo.
 - D) aumentar os limites de uso dos recursos.
 - E) definir o nome de host (hostname) do sistema.
-

67.(IBFC/EBSERH - 2016)

O comando sudo do sistema operacional Linux é muito poderoso, permitindo que usuários comuns obtenham privilégios de super usuário. Por questões de segurança, o administrador precisa definir no arquivo _____, quais usuários podem executar sudo, em quais computadores podem fazê-lo e quais comandos podem executar através dele. Assinale a alternativa que complete correta e respectivamente a lacuna:

- A) /etc/sudoers
 - B) /root/sudouser
 - C) /root/sudoers
 - D) /usr/sudouser
 - E) /etc/sudouser
-

68.(INAZ do Pará/DPE-PR - 2017)

Qual a alternativa que corresponde à linha de comando para que usuários comuns possam utilizar o comando administrativo apt-get dist-upgrade?

- A) # apt-get dist-upgrade
- B) \$ sudo apt-get gw dist-upgrade
- C) # sudo apt-get gw up dist-upgrade
- D) \$ sudo apt-get dist-upgrade



E) # sudo apt-get dist-upgrade gw now

69.(CESPE/ABIN - 2018)

O Linux não impede a alteração do nome do superusuário, nem a criação de contas com UID igual a 0, embora essas ações não sejam recomendadas.

70.(COSEAC/UFF - 2019)

No Sistema Operacional Linux o diretório local padrão do superusuário é:

- A) /home.
- B) /usr.
- C) /root.
- D) /lib.
- E) /dev.



GABARITO

- | | |
|------------|------------|
| 1. E | 40. C |
| 2. C | 41. C |
| 3. C | 42. A |
| 4. B | 43. E |
| 5. Errado | 44. E |
| 6. A | 45. E |
| 7. Errado | 46. C |
| 8. A | 47. B |
| 9. D | 48. C |
| 10. C | 49. C |
| 11. C | 50. D |
| 12. B | 51. A |
| 13. C | 52. A |
| 14. C | 53. D |
| 15. C | 54. A |
| 16. D | 55. B |
| 17. D | 56. B |
| 18. D | 57. D |
| 19. B | 58. D |
| 20. Certo | 59. A |
| 21. Errado | 60. Certo |
| 22. E | 61. E |
| 23. C | 62. A |
| 24. Errado | 63. C |
| 25. B | 64. C |
| 26. B | 65. Errado |
| 27. Certo | 66. C |
| 28. Errado | 67. A |
| 29. D | 68. D |
| 30. C | 69. Certo |
| 31. A | 70. C |
| 32. C | |
| 33. D | |
| 34. C | |
| 35. D | |
| 36. C | |
| 37. Errado | |
| 38. A | |
| 39. A | |



ESSA LEI TODO MUNDO CONHECE: PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.



1 Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.



2 Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).



3 Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).



4 Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).



5 Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).



6 Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).



7 Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.



8 O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.



Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.