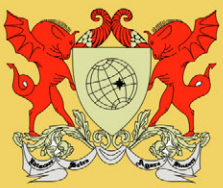
Material ampliado e adaptado do Guia Foca Linux

Curso Básico de LINUX

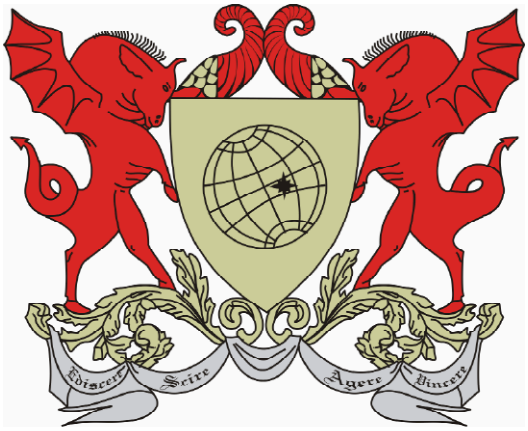
Módulo I



Universidade Federal de Viçosa

cead

Coordenadoria de Educação Aberta e à Distância



Universidade Federal de Viçosa

Nilda de Fátima Ferreira Soares - Reitora

Demétrius David da Silva - Vice-Reitor

CEAD - Coodenadoria de Educação Aberta e a Distância

Frederico Vieira Passos - Diretor

VAREJÃO-JR, C.G; COSTA,M.H. - Curso de Linux - Módulo I. Viçosa, 2010.

Layout: José Timóteo Júnior

Edição de imagens e capa: Diogo Rodrigues

Editoração Eletrônica: Diogo Rodrigues e Hamilton Henrique Teixeira Reis

Revisão Final: João Batista Mota

CEAD - Prédio CEE, Avenida PH Rolfs s/n
Campus Universitário, 36570-000, Viçosa/MG
Telefone: (31) 3899 2858 | Fax: (31) 3899 3352

SUMÁRIO

MÓDULO I

Material ampliado e adaptado do Guia Foca Linux

Apresentação do Curso

O material deste curso está organizado em módulos, para que possa ser disponibilizado na web à medida que o curso se desenvolver. Cada módulo corresponde a uma aula, e vem acompanhado de alguns exercícios, para que o estudante possa revisar o conteúdo abordado na respectiva aula.

Apesar de haver um professor ou monitor disponível (através de e-mail ou fórum) para tirar as dúvidas dos alunos, é imprescindível que o aluno se esforce para tirar suas dúvidas na internet, através dos sites de busca e fóruns disponíveis na web. Isso faz parte do aprendizado e é importante para a independência do profissional de Linux. Visitas ao Google e a listas de discussão fazem parte da vida de um administrador de sistemas Linux – se você está quebrando a cabeça com algum problema no seu Linux, provavelmente alguém já passou pelo mesmo problema e a solução está disponível na Internet. Além do mais, o linux é bem documentado e existem muitos tutoriais e *HOW TOs* disponíveis para quase tudo que se deseja fazer nele. Portanto, a primeira lição deste curso é: *"Google is your friend!"* (a propósito, buscas em inglês são na maioria das vezes mais eficientes).

Em vista dessas buscas por informação na Internet, o estudante e futuro administrador de sistemas linux deverá ler constantemente grandes quantidades de informação. É muito importante que o estudante não tenha preguiça de ler, pois, afinal, esse é o único jeito de se encontrar o que se está procurando, estudar e então praticar no seu linux. Além do mais, com as leituras durante a busca de uma solução para um problema, acaba-se aprendendo muito sobre assuntos relacionados, o que contribui bastante para a evolução do estudante e do profissional linux.

Em caso de dúvidas, entre em contato com o professor/monitor do curso ou troque ideias com os outros alunos através do nosso fórum de discussão. Lembre-se que ensinar é uma ótima maneira de fixar o aprendizado, e os fóruns da Internet são excelentes lugares para aprender, ensinar e se manter atualizado.

Dicas para um bom começo:

- Recomendo que faça a leitura deste guia e pratique imediatamente o que aprendeu. Isso facilita o entendimento do programa/comando/configuração.
- É preciso ter interesse em aprender. Se você tiver vontade em aprender algo, você terá menos dificuldade do que em algo que não gosta e está se obrigando a aprender.
- Decorar não adianta: pelo contrário, só atrapalha o aprendizado. Você precisa entender o que o comando faz, pois dessa forma estará também usando e desenvolvendo sua interpretação, e entenderá

melhor o assunto (talvez até me dê uma força para melhorar o guia).

- Curiosidade também é importante.
- Não desanime vendo outras pessoas que sabem mais que você, lembre-se que ninguém nasce sabendo. Uma pessoa pode ter mais experiência em um assunto do sistema, como compilação de programas, configuração, etc., e você pode ter mais interesse em redes, por exemplo.
- Ninguém pode saber tudo da noite para o dia. Não procure saber tudo sobre o sistema de uma só vez, senão você não entenderá nada. Caso tenha dúvidas sobre o sistema, procure ler novamente a seção do guia. Se mesmo assim não tenha entendido, procure ajuda nas páginas de manual.
- Certamente você buscará documentos na Internet que falem sobre algum assunto que este guia ainda não explica. Muito cuidado! O Linux é um sistema que cresce muito rapidamente: a cada semana uma nova versão é lançada, novos recursos são adicionados... seria maravilhoso se a documentação fosse atualizada com a mesma frequência.
- Infelizmente a atualização da documentação não segue o mesmo ritmo (principalmente aqui no Brasil). É comum você encontrar na Internet documentos da época quando o kernel (o núcleo ou a base de um sistema operacional) estava na versão 2.0.20, 2.0.30. Eles são úteis para pessoas que usem as versões antigas do Kernel Linux, mas podem trazer problemas ou causar má impressão do Linux em outras pessoas. Para evitar tais problemas, verifique a data de atualização do documento.
- O Linux é considerado um sistema mais difícil do que os outros, mas isso é porque ele requer que a pessoa realmente aprenda e conheça computadores e seus periféricos antes de fazer qualquer coisa (principalmente se você é um técnico em manutenção, redes ou instalações e deseja oferecer suporte profissional a esse sistema).
- Você conhecerá mais sobre computadores, redes, hardware, software, discos, saberá avaliar os problemas e a buscar a melhor solução. Enfim, as possibilidades de crescimento neste sistema operacional dependem do conhecimento, interesse e capacidade de cada um.
- A interface gráfica existe, mas os melhores recursos e flexibilidade estão na linha de comando. Você pode ter certeza que o aprendizado no Linux ajudará a ter sucesso e menos dificuldade em usar qualquer outro sistema operacional.
- Peça ajuda a outros usuários do Linux quando estiver em dúvida ou não souber fazer alguma coisa no sistema. Você pode entrar em contato com outros usuários através de listas de discussão.

Boa Sorte e bem vindo ao Linux!

1 Introdução

Este capítulo traz uma explicação básica sobre o sistema operacional Linux e algumas das principais distribuições existentes

1.1 Sistema Operacional

O Sistema Operacional é um programa (software) ou um conjunto de programas cuja função é servir de interface entre computador e usuário. É comum utilizar a abreviatura SO (em português) ou OS (do inglês "*Operating System"*).

Segundo Tanenbaum e Silberschatz, existem dois modos distintos de conceituar um sistema operacional: (i) pela perspectiva do usuário (visão "top-down"), é uma abstração do hardware, fazendo o papel de intermediário entre o aplicativo (programa) e os componentes físicos do computador (hardware); (ii) numa visão "bottom-up", de baixo para cima, é um gerenciador de recursos, ou seja, controla quais aplicações (processos) podem ser executadas, quando, que recursos (memória, disco, periféricos) podem ser utilizados.

Portanto, se não existissem os sistemas operacionais, todo programa desenvolvido deveria saber como comunicar-se com os dispositivos do computador que precisasse utilizar.

No Linux, o Kernel mais o conjunto de ferramentas GNU compõem o Sistema Operacional, por isso podemos chamá-lo de GNU/Linux. O kernel poderá ser construído de acordo com a configuração do seu computador e dos periféricos que ele possui.

1.2 O Linux

O termo ***Linux*** refere-se a qualquer sistema operacional do tipo Unix que utiliza o núcleo Linux. É um dos mais proeminentes exemplos de desenvolvimento com código aberto e de software livre. O seu código fonte está disponível sob GPL *(GNU General Public License (Licença Pública Geral))* para qualquer pessoa utilizar, estudar, modificar e distribuir livremente.

GNU/Linux refere-se a qualquer sistema operacional do tipo Unix que utiliza o núcleo Linux e também os programas de sistema GNU. Como os casos de sistemas de núcleo Linux sem os programas de sistema GNU são raros, frequentemente GNU/Linux e Linux são sinônimos.

Inicialmente desenvolvido e utilizado por nichos de entusiastas em computadores pessoais, o sistema Linux passou a ter a colaboração de grandes empresas, como a IBM, a Sun Microsystems, a Hewlett-Packard, e a Novell, ascendendo como principal sistema operacional para servidores – oito dos dez serviços de hospedagem mais confiáveis da Internet utilizam o sistema Linux em seus servidores web.

O Linux tornou-se o sistema capaz de funcionar no maior número de arquiteturas computacionais disponíveis. É utilizado em aparelhos variando desde supercomputadores até celulares, e vem se tornando bastante popular no mercado de computadores pessoais.

O Linux é, geralmente, considerado o Núcleo (ou "cerne", "coração",

do inglês *kernel*) do sistema operacional. Entretanto, segundo Tanenbaum e Schilberschatz, ele pode ser considerado o próprio SO, quando este é definido como gerenciador de recursos de hardware. Nos meios de comunicação, Linux refere-se ao sistema completo, incluindo o núcleo Linux e outros programas de sistema.

Sistemas completos construídos em torno do kernel Linux majoritariamente utilizam o sistema GNU, que oferece um interpretador de comandos, utilitários, bibliotecas e compiladores. Richard M. Stallman, criador e líder do projeto GNU, solicita aos utilizadores que se refiram aos sistemas baseados no Linux como o sistema completo GNU/Linux.

A maioria dos sistemas também inclui ferramentas e utilitários baseados no BSD (*Berkeley Software Distribution é um Sistema Operacional UNIX desenvolvido pela Universidade de Berkeley, na Califórnia*) e tipicamente usam XFree86 ou X.org para oferecer a funcionalidade dos sistemas de janelas X - interface gráfica.

O Linux hoje funciona em dezenas de plataformas, desde mainframes até um relógio de pulso, passando por várias arquiteturas: x86 (Intel, AMD), x86-64 (Intel EM64T, AMD64), StrongARM, PowerPC, Alpha etc., com grande penetração também em sistema embutidos, como handheld, PVR, vídeo-jogos e centros multimídia, entre outros.

1.2.1 História do Linux

O Kernel do Linux foi, originalmente, escrito por Linus Torvalds, do Departamento de Ciência da Computação da Universidade de Helsinki, Finlândia, com a ajuda de vários programadores voluntários ligados pela Usenet.

Linus Torvalds começou o desenvolvimento do kernel como um projeto particular, inspirado pelo seu interesse no Minix, um pequeno sistema UNIX desenvolvido por Andrew S. Tanenbaum. Ele limitou-se a criar, nas suas próprias palavras, "um Minix melhor que o Minix" (*"a better Minix than Minix"*). E depois de algum tempo de trabalho no projeto, sozinho, ele enviou a seguinte mensagem para comp.os.minix:

> *"Você suspira pelos bons tempos do Minix-1.1, quando os homens eram homens e escreviam seus próprios "device drivers"? Você está sem um bom projeto em mãos e está desejando trabalhar num S.O. que você possa modificar de acordo com as suas necessidades? Está achando frustrante quando tudo funciona no Minix? Chega de noite ao computador para conseguir que os programas funcionem? Então esta mensagem pode ser exatamente para você.*
>
> *Como eu mencionei há um mês atrás, estou trabalhando numa versão independente de um S.O. similar ao Minix para computadores AT-386. Ele está, finalmente, próximo do estado em que poderá ser utilizado (embora possa não ser o que você está esperando), e eu estou disposto a disponibilizar o código-fonte para ampla distribuição. Ele está na versão 0.02... contudo eu tive sucesso ao executar bash, gcc, gnu-make, gnu-sed, compressão e outras coisas nele."*

No dia 5 de outubro de 1991, Linus Torvalds anunciou a primeira versão "oficial" do Linux, (0.02). Desde então, muitos programadores têm respondido ao seu chamado, e têm ajudado a fazer do Linux o sistema operacional que é hoje.

1.2.2 Algumas Características do Linux

- É livre e desenvolvido voluntariamente por programadores experientes, hackers e colaboradores espalhados ao redor do mundo, que têm como objetivo a contribuição para a melhoria e crescimen-

to desse sistema operacional. Muitos deles estavam cansados do excesso de propaganda e baixa qualidade de sistemas comerciais existentes.

- Convivem sem nenhum tipo de conflito com outros sistemas operacionais (com o DOS, Windows, OS/2) no mesmo computador.
- Multitarefa real (capaz de executar várias tarefas ao mesmo tempo)
- Multiusuário (suporta mais de um usuário ao mesmo tempo)
- Suporte a nomes extensos de arquivos e diretórios (255 caracteres).
- Conectividade com outros tipos de plataformas como *Apple, Sun, Macintosh, Sparc, Alpha, PowerPc, ARM, Unix, Windows e DOS.*
- Proteção entre processos executados na memória RAM.
- Suporte a mais de 63 terminais virtuais (consoles).
- Modularização - O GNU/Linux somente carrega para a memória o que é usado durante o processamento, liberando totalmente a memória assim que o programa/dispositivo é finalizado.
- Devido à modularização, os drivers dos periféricos e recursos do sistema podem ser carregados e removidos completamente da memória RAM a qualquer momento. Os drivers (módulos) ocupam pouco espaço quando carregados na memória RAM (cerca de 6Kb para a Placa de rede NE 2000, por exemplo).
- Não há a necessidade de se reiniciar o sistema após modificar a configuração de qualquer periférico ou parâmetros de rede. Somente é necessário reiniciar o sistema no caso de uma instalação interna de um novo periférico, ou falha em algum hardware (como a queima do processador ou placa mãe).
- Não é preciso um processador potente para funcionar. O sistema roda bem em computadores 386Sx 25 com 4MB de memória RAM (sem rodar o sistema gráfico X, que é recomendado 8MB de RAM). Já pensou no seu desempenho em um AM2 ou um Core 2 Duo?
- O crescimento e novas versões do sistema não provocam lentidão, pelo contrário, a cada nova versão os desenvolvedores procuram buscar maior compatibilidade, acrescentar recursos úteis e melhorar o desempenho do sistema (como o que aconteceu na passagem do kernel 2.0.x para 2.2.x).
- Não é requerida uma licença para seu uso. O GNU/Linux é licenciado de acordo com os termos da GPL.
- Acessa corretamente discos formatados pelo DOS, Windows, Novell, OS/2, NTFS, SunOS, Amiga, Atari, Mac etc.
- Utiliza permissões de acesso a arquivos, diretórios e programas em execução na memória RAM.
- O LINUX NÃO É VULNERÁVEL A VÍRUS! Devido à separação de privilégios entre processos e respeitadas as recomendações padrão de política de segurança e uso de contas privilegiadas (como a de root, como veremos adiante), programas como vírus tornam-se inúteis, pois tem sua ação limitada pelas restrições de acesso do sistema de arquivos e execução. Frequentemente são criados *exploits* que tentam se aproveitar de falhas existentes em sistemas desatualizados e usá-las para danificar o sistema. Erroneamente esse tipo de ataque é classificado como vírus por pessoas mal informadas e são resolvidas com sistemas bem mantidos. Em geral, usando uma boa distribuição que tenha um bom sistema de atualização resolve em 99.9% os problemas com *exploits*. Qualquer programa (nocivo ou não) poderá alterar partes do sistema que possui permissões (será abordado como alterar permissões e tornar seu sistema mais restrito no decorrer do curso).
- A rede ***TCP/IP*** (TCP (Transmission Control Protocol) e IP (Internet Protocol). O TCP/IP é um conjunto (ou pilha) de protocolos de co-

municação entre computadores em rede. O conjunto de protocolos pode ser visto como um modelo de camadas, onde cada camada é responsável por um grupo de tarefas, fornecendo um conjunto de serviços bem definidos para o protocolo da camada superior. É mais rápida que no Windows e tem suas pilhas constantemente melhoradas. O GNU/Linux tem suporte nativo a redes TCP/IP e não depende de uma camada intermediária, como o WinSock. Em acessos à Internet via modem, a velocidade de transmissão é 10% maior.

- Roda aplicações DOS através do DOSEMU, QEMU, BOCHS. Para se ter uma idéia, é possível dar o boot em um sistema DOS qualquer dentro dele e ao mesmo tempo usar a multitarefa deste sistema.
- Roda aplicações Windows através do WINE.
- Suporte a dispositivos infravermelho.
- Suporte a rede via rádio amador.
- Suporte a dispositivos Plug-and-Play (dispositivos que são reconhecidos e configurados automaticamente pelo computador).
- Suporte a dispositivos USB.
- Suporte a Fireware (Tecnologia semelhante a do USB, porém com transferência mais rápida dos dados).
- Dispositivos Wireless.
- Vários tipos de firewalls (dispositivo de rede para aplicar política de segurança) de alta qualidade e com grande poder de segurança de graça.
- Possui recursos para atender a mais de um endereço IP na mesma placa de rede, sendo muito útil para situações de manutenção em servidores de redes ou para a emulação (simulação feita por um software que reproduz as funções de um determinado ambiente, a fim de permitir a execução de outros softwares sobre ele) de “mais computadores” virtualmente.
- Os servidores WEB e FTP podem estar localizados no mesmo computador, mas o usuário que se conecta tem a impressão que a rede possui servidores diferentes.
- O sistema de arquivos usado pelo GNU/Linux (Ext2) organiza os arquivos de forma inteligente, evitando a fragmentação e fazendo dele um poderoso sistema para gravações intensivas e aplicações multi-usuárias exigentes.
- Permite a montagem de servidores Web, E-mail, News e outros com um baixo custo e alta performance. O melhor servidor Web do mercado, o Apache, é distribuído gratuitamente junto com a maioria das distribuições Linux. O mesmo acontece com o Sendmail.
- Por ser um sistema operacional de código aberto, você pode ver o que o código fonte (instruções digitadas pelo programador) faz e adaptá-lo às suas necessidades ou de sua empresa. Essa característica é uma segurança a mais para quem precisa se defender contra roubo de dados (você não sabe o que um sistema sem código fonte faz na realidade, enquanto está processando o programa).
- Suporte a diversos dispositivos e periféricos disponíveis no mercado, tanto os novos como obsoletos.
- Pode ser executado em várias arquiteturas diferentes (Intel, Macintosh, Alpha e Arm por exemplo).
- Existem consultores técnicos especializados no suporte ao sistema espalhados por todo o mundo.

Existem também muitas outras características, as quais você descobrirá durante o uso do sistema.

1.3 Distribuições do Linux

Só o kernel GNU/Linux não é suficiente para se ter uma sistema funcional, apesar de ser o principal.

Existem grupos de pessoas, empresas e organizações que decidem "distribuir" o Linux junto com outros programas essenciais (como, por exemplo, editores gráficos, planilhas, bancos de dados, ambientes de programação, formatação de documentos e firewalls).

Esse é o significado básico de distribuição. Cada distribuição tem sua característica própria, como o sistema de instalação, o seu objetivo, a localização de programas, nomes de arquivos de configuração. A escolha de uma distribuição é pessoal e depende das necessidades de cada um.

Algumas distribuições muito conhecidas são: *Slackware, Debian, Red Hat, Mandriva, Suse, Ubuntu, Gentoo*, todas usando o SO Linux como kernel principal (a Debian é uma distribuição independente de kernel e pode ser executada sob outros kernels, como o GNU hurd).

A escolha de sua distribuição deve ser feita com muita atenção. Não adianta muito recorrer a chats e fóruns sobre qual é a melhor distribuição, muito menos ser levado pelas propagandas ou pelo vizinho. O melhor caminho para a escolha da distribuição é perguntar as características de cada uma e o porquê dessa pessoa gostar dela. Não pergunte qual é a melhor versão, pois cada um usa uma distribuição que se encaixa de acordo com suas necessidades, portanto, essa mesma distribuição pode não ser a melhor para lhe atender.

Seguem abaixo as características de algumas distribuições, seguidas do endereço do site principal:

Debian

(Saiba mais em http://www.debian.org/) Distribuição desenvolvida e atualizada através do esforço de voluntários espalhados ao redor do mundo, seguindo o estilo de desenvolvimento GNU/Linux. Por esse motivo, foi adotada como a distribuição oficial do projeto GNU. Possui suporte para língua Portuguesa, tem suporte a dez arquiteturas diferentes (como i386, Alpha, Sparc, PowerPc, Macintosh, Arm) e aproximadamente 15 sub-arquiteturas. A instalação da distribuição pode ser feita tanto através de Disquetes, CD-ROM, Tftp, Ftp, NFS ou através da combinação de vários destes em cada etapa de instalação.

Possui tanto ferramentas para administração de redes e servidores quanto para desktops, estações multimídia, jogos, desenvolvimento e web.

A atualização da distribuição ou de pacotes individuais pode ser feita facilmente através de dois comandos, não requerendo adquirir um novo CD para usar a última versão da distribuição. É a única distribuição não comercial onde todos podem contribuir com seu conhecimento para o seu desenvolvimento. Para gerenciar os voluntários, conta com centenas de listas de discussão envolvendo determinados desenvolvedores das mais diversas partes do mundo.

Para atingir um alto grau de confiabilidade, são feitos extensivos testes antes do lançamento de cada versão. Os erros encontradas nos pacotes podem ser relatados através de um sistema de tratamento de falhas, que encaminha a falha encontrada diretamente ao responsável para avaliação e correção. Qualquer um pode receber a lista de falhas ou sugestões sobre a distribuição: basta cadastrar-se numa das listas de discussão que tratam especificamente da solução de falhas encontradas na distribuição (disponível na página principal da distribuição).

Os pacotes podem ser instalados através de Tarefas, contendo seleções de pacotes de acordo com a utilização do computador, (servidor Web, desenvolvimento, TeX, jogos, desktop etc.) ou através de uma seleção indi-

vidual de pacotes, garantindo que somente os pacotes selecionados sejam instalados.

Existe um time de desenvolvedores com a tarefa específica de monitorar atualizações de segurança em serviços (apache, sendmail, e todos os outros pacotes) que possam comprometer o servidor, deixando-o vulnerável a ataques. Assim que uma falha é descoberta, é enviado uma alerta *(DSA - Debian Security Alert)* e disponibilizada uma atualização para correção das diversas versões da Debian. Isso é geralmente feito em menos de 48 horas desde a descoberta da falha até a divulgação da correção. Como quase todas as falhas são descobertas em programas, esse método também pode ser usado por administradores de outras distribuições para manterem seu sistema seguro e atualizado.

O suporte ao usuário e o desenvolvimento da distribuição são feitos através de listas de discussões e canais IRC.

Slackware

(Saiba mais em http://www.slackware.com/) Distribuição desenvolvida por Patrick Volkerding, a fim de alcançar facilidade de uso e estabilidade como prioridades principais. Foi a primeira distribuição a ser lançada no mundo e costuma trazer o que há de mais novo enquanto mantém uma certa tradição, provendo simplicidade, facilidade de uso e, com isso, flexibilidade e poder.

Desde a primeira versão, lançada em Abril de 1993, o Projeto Slackware Linux tem buscado produzir a distribuição Linux mais UNIX-like, ou seja, mais parecida com UNIX. O Slackware segue os padrões Linux como o Linux File System Standard, que é um padrão de organização de diretórios e arquivos para as distribuições.

Enquanto as pessoas diziam que a Red Hat era a melhor distribuição para o usuário iniciante, o Slackware era o melhor para o usuário mais “velho”, ou seja, programadores e administradores.

SuSE

(Saiba mais em http://www.suse.com/) Distribuição comercial Alemã com a coordenação sendo feita através dos processos administrativos dos desenvolvedores e de seu braço norte-americano. O foco da Suse é o usuário com conhecimento técnico no Linux e não o usuário iniciante no Linux (até a versão 6.2).

Sua instalação pode ser feita via CD-ROM ou DVD (é a primeira distribuição com instalação através de DVD).

O sistema de gerenciamento de pacotes é o RPM padronizado. A seleção de pacotes durante a instalação pode ser feita através da seleção do perfil de máquina (developer, estação kde, gráficos, estação gnome, servidor de rede, etc.) ou através da seleção individual de pacotes.

A atualização da distribuição pode ser feita através do CD-ROM de uma nova versão ou baixando pacotes pela internet. Usuários registrados têm direito a suporte de instalação via e-mail. A base de dados de suporte também é excelente e está disponível na web para qualquer usuário, independente de registro.

Red Hat Enterprise Linux

(Saiba mais em http://www.redhat.com/) Distribuição comercial suportada pela Red Hat e voltada a servidores de grandes e médias empresas. Também conta com uma certificação chamada RHCE, específica desta distro.

Ela não está disponível para download, apenas vendida a custos desde 179 dólares (a versão workstation) até 1499 dólares (advanced server).

Fedora

(Saiba mais em http://fedora.redhat.com/) O Fedora Linux é a distribuição de desenvolvimento aberto patrocinada pela RedHat e pela comunidade. Criada em 2002, é baseada na versão da antiga linha de produtos RedHat Linux, a distribuição mais utilizada do mundo. Ela não é suportada pela Red Hat como distribuição oficial (a empresa suporta apenas a linha Red Hat Enterprise Linux), devendo obter suporte através da comunidade ou outros meios.

A distribuição Fedora dá prioridade ao uso do computador como estação de trabalho. Além de contar com uma ampla gama de ferramentas de escritório, possui funções de servidor e aplicativos para produtividade e desenvolvimento de softwares. Considerado um dos sistemas mais fáceis de instalar e utilizar, inclui tradução para português do Brasil e suporte às plataformas Intel e 64 bits.

Por basear-se no RedHat, o Fedora conta com um o up2date (um software para manter o sistema atualizado) e utiliza pacotes de programas no formato RPM, um dos mais comuns. Por outro lado, não possui suporte a MP3, Video Players ou NTFS (Discos do Windows) em virtude de problemas legais, sendo necessário o download de alguns plugins para a utilização destas funções.

O Fedora não é distribuído oficialmente através de mídias ou CDs. Se você quiser obtê-lo, terá de procurar distribuidores independentes ou fazer o download dos quatro CDs pelo site oficial.

Kurumin

(Saiba mais em http://guiadohardware.net/kurumin/index.php/) Uma distribuição baseada em Debian que roda diretamente a partir do CD, sendo ideal para quem deseja testar uma distribuição Linux. Caso goste, o usuário pode instalá-la diretamente no disco rígido. Distribuída a partir do CD, é de muito fácil utilização e suporta uma boa quantidade de hardwares disponíveis. A versão instalada possui, inclusive, suporte para a maioria dos winmodens encontrados no Brasil.

Mandriva

O Mandriva é uma das distribuições Linux mais fáceis de usar, mais robustas e antigas, apesar da popularidade e do uso serem baixos no Brasil. Aqui (no Brasil) as distros mais usadas são as baseadas em Debian, mas pouca gente (nova) sabe que Conectiva e Mandrake já tiveram seu grande reinado. O antigo Mandrake foi uma das primeiras distribuições a usar um instalador gráfico e era, na época, de muito fácil operação, quando a maioria dos usuários de Linux era de técnicos e as distribuições, obviamente, voltadas para eles. Existia, na época, a Conectiva, a primeira distribuição nacional, com um suporte gigantesco aos usuários brasileiros e líder na América Latina.

Enfim, no dia 24 de fevereiro de 2005, a MandrakeSoft anunciou a compra da Conectiva, por US$2,3 milhões. A partir daí, a empresa passou a se chamar Mandriva.

O Mandriva possui duas versões: o pacote comercial, composto pelo Discovery (iniciantes), Powerpack (para experientes) e Powerpack+ (pequenas e médias empresas), todos eles pagos. Já o Mandriva Free não contém aplicativos ou drivers proprietários nem suporte oficial, porém tem o download disponibilizado livremente.

Gentoo Linux

É uma distribuição gratuita do sistema operacional GNU/Linux baseada na GNU General Public License. O diferencial do Gentoo Linux em relação às outras distribuições está no uso da ferramenta Portage, que dá ao usuário a possibilidade de ter um sistema adaptado ao seu perfil, pois cada pacote é compilado durante a instalação, de forma automatizada, otimizada e com todas as dependências resolvidas. Devido a essas características, o Gentoo Linux é considerado uma meta-distribuição (Meta-distribuição é um termo referente a uma distribuição de software adaptável e capaz de construir a si mesmo, baseado nas especificações dadas pelo usuário).

Essa distribuição é direcionada para usuários avançados ou experientes, devido à complexidade de executar tarefas que em outras distribuições são realizadas de forma simples. Porém, é aconselhada também a todos que queiram saber o máximo sobre GNU/Linux. Quando bem instalada e configurada, a distribuição se torna mais veloz, correndo programas emulados mais rápido que o próprio Windows.

A distribuição adotada neste curso é a Debian. Entretanto, boa parte do conteúdo abordado é válido para qualquer distribuição Linux.

1.4 Software Livre

Software livre, segundo a definição criada pela Free Sofware Foudation, é qualquer programa de computador que pode ser usado, copiado, estudado, modificado e redistribuído sem nenhuma restrição. Essa liberdade é um conceito central, que se opõe ao modelo de software proprietário, mas não ao software que é vendido almejando lucro (software comercial). A maneira usual de distribuição de software livre é anexar a ele uma licença de software livre e tornar o código fonte do programa disponível. O software livre também é conhecido pelo acrônimo FLOSS (do inglês Free/Libre Open Source Software).

Um software é considerado como livre quando atende aos quatro tipos de liberdade para usuários do software definidos pela Free Software Foundation:

- A liberdade para executar o programa, para qualquer propósito (liberdade nº 0)
- A liberdade para estudar como o programa funciona e adaptá-lo para as suas necessidades (liberdade nº1). Acesso ao código-fonte é um pré-requisito para esta liberdade
- A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo (liberdade nº2)
- A liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, de modo que toda a comunidade se beneficie (liberdade nº3). O acesso ao código-fonte é um pré-requisito para esta liberdade.

2 Alguns Conceitos Básicos

São apresentados, neste capítulo, alguns conceitos relacionados ao computador e ao sistema operacional.

2.1 Hardware e Software

- **Hardware** significa parte física do computador (disquete, impressoras, monitores, placa mãe, placa de fax, discos rígidos, etc).
- **Softwares** são os programas usados no computador (sistema operacional, processador de textos, planilha, banco de dados, scripts, comandos, etc).

2.2 Terminal Virtual (console)

Terminal (ou console) é o teclado e tela conectados em seu computador. O GNU/Linux faz uso de sua característica multi-usuária usando os "terminais virtuais". Um terminal virtual é uma segunda seção de trabalho completamente independente de outras, que pode ser acessada no computador local ou remotamente, via ssh, por exemplo.

No GNU/Linux, em modo texto, você pode acessar outros terminais virtuais segurando a tecla ALT e pressionando F1 a F6. Cada tecla de função corresponde a um número de terminal do 1 ao 6 (o sétimo é usado por padrão pelo ambiente gráfico X). O GNU/Linux possui mais de 63 terminais virtuais, mas apenas seis estão disponíveis inicialmente, por motivos de economia de memória RAM (cada terminal virtual ocupa aproximadamente 350 Kb de memória RAM. Desative a quantidade que não estiver usando para liberar memória RAM para uso de outros programas!).

Se estiver usando o modo gráfico, você deve segurar CTRL+ALT enquanto pressiona uma tecla de <F1> a <F6>.

Um exemplo prático: Se você estiver usando o sistema no Terminal 1 com o nome "joao" e desejar entrar como "root" para instalar algum programa, segure ALT enquanto pressiona <F2> para abrir o segundo terminal virtual e faça o login como "root". Será aberta uma nova seção para o usuário "root" e você poderá retornar a hora que quiser para o primeiro terminal pressionando ALT+<F1>.

2.3 Aviso de comando (Prompt)

Aviso de comando (ou Prompt), é a linha mostrada na tela do terminal para digitação de comandos, que serão passados ao interpretador de comandos para sua execução. A posição onde o comando será digitado é marcada com um "traço" piscante chamado de cursor.

É necessário o uso do cursor para sabermos onde iniciar a digitação de textos e nos orientarmos quanto à posição na tela.

O aviso de comando do usuário root é identificado por uma "#" (tralha),

e o aviso de comando de usuários é identificado pelo símbolo “$”. Isso é padrão em sistemas UNIX.

Você pode retornar a comandos já digitados pressionando as teclas Seta para cima / Seta para baixo.

A tela pode ser rolada para baixo ou para cima segurando a tecla SHIFT e pressionando PGUP ou PGDOWN. Isto é útil para ver textos que rolaram rapidamente para cima.

Logo abaixo, veja algumas dicas sobre a edição da linha de comandos (não é necessário se preocupar em decorá-los):

- **Pressione a tecla Back Space (“<--”)** para apagar um caractere à esquerda do cursor.
- **Pressione a tecla Del** para apagar o caractere acima do cursor.
- **Pressione CTRL+A** para mover o cursor para o inicio da linha de mandos.
- **Pressione CTRL+E** para mover o cursor para o fim da linha de comandos.
- **Pressione CTRL+U** para apagar o que estiver à esquerda do cursor. O conteúdo apagado é copiado para uso com CTRL+y.
- **Pressione CTRL+K** para apagar o que estiver à direita do cursor. O conteúdo apagado é copiado para uso com CTRL+y.
- **Pressione CTRL+L** para limpar a tela e manter o texto que estiver sendo digitado na linha de comando (parecido com o comando clear).

Pressione CTRL+Y para colocar o texto que foi apagado na posição atual do cursor.

2.4 Interpretador de comandos

Também conhecido como “shell”, o interpretador de comandos é o programa responsável em interpretar as instruções enviadas pelo usuário e seus programas ao sistema operacional (o kernel). Ele que executa comandos lidos do dispositivo de entrada padrão (teclado) ou de um arquivo executável. É a principal ligação entre o usuário, os programas e o kernel. O GNU/Linux possui diversos tipos de interpretadores de comandos, sendo que entre eles o mais usado é o bash. Os comandos podem ser enviados de duas maneiras para o interpretador: interativa e não-interativa:

Interativa

Os comandos são digitados no aviso de comando e passados ao interpretador de comandos um a um. Neste modo, o computador depende do usuário para executar uma tarefa, ou o próximo comando.

Não-interativa

São usados arquivos de comandos criados pelo usuário (scripts) para o computador executar tais comandos na ordem encontrada no arquivo. Neste modo, o computador executa os comandos do arquivo um por um, e dependendo do término do comando, o script pode checar qual será o próximo comando que será executado e dar continuidade ao processamento.

Este sistema é útil quando temos que digitar por várias vezes seguidas um mesmo comando, ou para compilar algum programa complexo, por exemplo.

O shell Bash possui ainda outra característica interessante: A completação dos nomes. Isto é feito pressionando-se a tecla TAB. Por exemplo, se digitar “ls tes” e pressionar <tab>, o Bash localizará todos os arquivos que iniciam com “tes” e completará o restante do nome. Caso a completação de nomes encontre mais do que uma expressão que satisfaça a pesquisa, ou nenhuma, é emitido um beep. Se você apertar novamente a tecla TAB

imediatamente depois do beep, o interpretador de comandos irá listar as diversas possibilidades que satisfaçam a pesquisa, para que você possa escolher a que lhe interessa. A completação de nomes funciona sem problemas para comandos internos.

Exemplos: ech(pressione TAB).
ls tes(pressione TAB)

2.5 Login

Login é a entrada no sistema, quando você digita seu nome e senha.

2.6 Logout

Logout é a saída do sistema. A saída do sistema é feita pelos comandos logout, exit ou CTRL+D ou quando o sistema é reiniciado ou desligado.

2.7 Comandos

Comandos são ordens que passamos ao sistema operacional para executar uma determinada tarefa.

Cada comando tem uma função específica – devemos saber a função de cada comando e escolher o mais adequado para fazer o que desejamos,. Por exemplo:

- ls - Mostra arquivos de diretórios
- cd - Para mudar de diretório

Este material inclui, mais adiante, uma lista de vários comandos organizados por categoria com a explicação sobre o seu funcionamento e as opções aceitas (incluindo alguns exemplos).

É sempre usado um espaço depois do comando para separá-lo de uma opção ou parâmetro que será passado para o processamento. Um comando pode receber opções e parâmetros:

opções

As opções são usadas para controlar como o comando será executado. Por exemplo, para fazer uma listagem mostrando o dono, grupo e tamanho dos arquivos você deve digitar ls -l.

Opções podem ser passadas ao comando através de um “-” ou “--”:

-

Opção identificada por uma letra. Podem ser usadas mais de uma opção com um único hífen. O comando ls -l -a é a mesma coisa de ls -la

--

Opção identificada por um nome. O comando ls --all é equivalente a ls -a.

Pode ser usado tanto “-” como “--”, mas há casos em que somente “-” ou “--” esta disponível.

parâmetros

Um parâmetro identifica o caminho, origem, destino, entrada padrão ou saída padrão que será passada ao comando.

Se você digitar: ls /usr/doc/copyright, /usr/doc/copyright será o parâmetro passado ao comando ls. Neste caso queremos que ele liste os arquivos do diretório /usr/doc/copyright.

É normal errar o nome de comandos, mas não se preocupe. Quando isto acontecer, o sistema mostrará a mensagem command not found (co-

mando não encontrado) e voltará ao aviso de comando. As mensagens de erro não fazem nenhum mal ao seu sistema, somente dizem que algo deu errado para que você possa corrigir e entender o que aconteceu. No GNU/Linux, você tem a possibilidade de criar comandos personalizados usando outros comandos mais simples (isto será visto mais adiante). Os comandos se encaixam em duas categorias: Comandos Internos e Comandos Externos.

Por exemplo, em "ls -la /usr/doc", ls é o comando, -la é a opção passada ao comando, e /usr/doc é o diretório passado como parâmetro ao comando ls.

2.8 Arquivo

É onde gravamos nossos dados. Um arquivo pode conter um texto feito por nós, uma música, um programa, uma planilha e vários outras informações.

Cada arquivo deve ser identificado por um nome, para que ele possa ser encontrado facilmente quando for utilizado. Se estiver fazendo um trabalho de história, nada melhor que salvá-lo com o nome *historia*. Um arquivo pode ser binário ou texto (para detalhes veja Arquivo texto e binário, Seção 2.2.3).

O GNU/Linux é Case Sensitive, ou seja, ele diferencia letras maiúsculas e minúsculas nos nomes de arquivos. O arquivo *historia* é completamente diferente de *Historia*. Essa regra também é válida para os comandos e diretórios. Prefira, sempre que possível, usar letras minúsculas para identificar seus arquivos, pois quase todos os comandos do sistema estão em minúsculas.

Um arquivo oculto no GNU/Linux é identificado por um "." no início do nome (por exemplo, .bashrc). Arquivos ocultos não aparecem em listagens normais de diretórios. Deve ser usado o comando ls –a para também listar arquivos ocultos.

2.8.1 Extensão de arquivos

A extensão serve para identificar o tipo do arquivo. A extensão é formada pelas letras após um "." no nome de um arquivo. Por exemplo:

- relatório.txt - O .txt indica que o conteúdo é um arquivo texto.
- script.sh - Arquivo de Script (interpretado por /bin/sh).
- system.log - Registro de algum programa no sistema.
- arquivo.gz - Arquivo compactado pelo utilitário gzip.
- index.html - Página de Internet (formato Hypertexto).

A extensão de um arquivo também ajuda identificar que programa precisa ser utilizado para abri-lo. Por exemplo, o arquivo relatório.txt é um texto simples e podemos ver seu conteúdo através do comando cat. Já o arquivo index.html contém uma página de Internet e precisaremos de um navegador para poder visualizá-lo (como o lynx, Firefox ou o Netscape).

A extensão (na maioria dos casos) não é requerida pelo sistema operacional GNU/Linux, mas é conveniente o seu uso para determinarmos facilmente o tipo de arquivo e de que programa precisaremos para abri-lo.

2.8.2 Tamanho de arquivos

A unidade de medida padrão nos computadores é o bit. A um conjunto de 8 bits nós chamamos de byte. Cada arquivo/diretório possui um tamanho que indica o espaço que ele ocupa no disco, e isto é medido em bytes. O byte representa uma letra em texto. Assim, se você criar um arquivo

vazio, escrever o nome Linux e salvar o arquivo, este terá o tamanho de 5 bytes. Espaços em branco e novas linhas também ocupam bytes.

Além do byte, existem as medidas Kbytes, Mbytes, Gbytes. Os prefixos K (quilo), M (mega), G (giga), T (tera) vêm da matemática. O "K" significa multiplicar por 10^3, o "M" por 10^6, e assim por diante. Estas letras servem para facilitar a leitura em arquivos de grande tamanho. Um arquivo de 1K é a mesma coisa de um arquivo de 1024 bytes. Uma forma que pode inicialmente lhe ajudar a lembrar: K vem de Kilo que é igual a 1000 - 1Kilo é igual

Por exemplo, em "ls -la /usr/doc", ls é o comando, -la é a opção passada ao comando, e /usr/doc é o diretório passado como parâmetro ao comando ls.

a 1000 gramas, certo?.

Da mesma forma, 1Mb (ou 1M) é igual a um arquivo de 1024K ou 1024x1024 = 1.048.576 bytes

1Gb (ou 1G) é igual a um arquivo de 1024Mb ou 1048576Kb ou 1.073.741.824 bytes (1 Gb é igual a 1.073.741.824 bytes; são muitos números!). Deu pra notar que é mais fácil escrever e entender como 1Gb do que como 1.073.741.824 bytes.

A lista completa em ordem progressiva das unidades de medida é a seguinte:

Símb.	10^	2^	Nome
K	3	10	Quilo
M	6	20	Mega
G	9	30	Giga
T	12	40	Tera
P	15	50	Peta
E	18	60	Eta
Z	21	70	Zetta
Y	24	80	Yotta

2.8.3 Arquivo texto e binário

Quanto ao tipo, um arquivo pode ser de texto ou binário:

Texto

Seu conteúdo é compreendido pelas pessoas. Um arquivo texto pode ser uma carta, um script, um programa de computador escrito pelo programador, arquivo de configuração, etc.

Binário

Seu conteúdo somente pode ser entendido por computadores. Contém caracteres incompreensíveis para pessoas normais. Um arquivo binário é gerado por um arquivo de programa (formato texto) através de um processo chamado de compilação. Compilação é basicamente a conversão de um programa em linguagem humana (texto) para a linguagem de máquina (binário).

2.9 Diretório

Diretório é o local utilizado para armazenar conjuntos de arquivos para melhor organização e localização, conhecidos como pastas. O diretório, como o arquivo, também é "Case Sensitive" (diretório /teste é completa-

mente diferente do diretório /Teste).

Não podem existir dois arquivos com o mesmo nome em um diretório, ou um sub-diretório com um mesmo nome de um arquivo em um mesmo diretório.

Os diretório nos sistemas Linux são especificados por uma "/". Para criar um diretório utilizamos o comando *mkdir*, que será visto mais adiante.

2.9.1 Comandos para manipulação de diretórios

ls

Lista os arquivos de um diretório.
ls [opções] [caminho/arquivo] [caminho1/arquivo1] ...

Onde:

caminho/arquivo

Diretório/arquivo que será listado.

caminho1/arquivo1

Outro diretório/arquivo que será listado. Podem ser feitas várias listagens de uma só vez.

opções

-a, --all

Lista todos os arquivos (inclusive os ocultos) de um diretório.

-A, --almost-all

Lista todos os arquivos (inclusive os ocultos) de um diretório, exceto o diretório atual e o de nível anterior.

-B, --ignore-backups

Não lista arquivos que terminam com ~ (Backup).

--color=PARAM

- Mostra os arquivos em cores diferentes, conforme o tipo de arquivo. PARAM pode ser: *never* - Nunca lista em cores (mesma coisa de não usar o parâmetro --color).
- *always* - Sempre lista em cores conforme o tipo de arquivo.
- *auto* - Somente colore a listagem se estiver em um terminal.

-d, --directory

Lista os nomes dos diretórios ao invés do conteúdo.

-f

Não classifica a listagem.

-F

Insere um caracter após arquivos executáveis ('*'), diretórios ('/'), soquete ('='), link simbólico ('@') e pipe ('|'). Seu uso é útil para identificar de forma fácil tipos de arquivos nas listagens de diretórios.

-G, --no-group

Oculta a coluna de grupo do arquivo.

-h, --human-readable

Mostra o tamanho dos arquivos em Kbytes, Mbytes, Gbytes.

-H

Faz o mesmo que -h, mas usa unidades de 1000 ao invés de 1024 para especificar Kbytes, Mbytes, Gbytes.

-l

Usa o formato longo para listagem de arquivos. Lista as permissões, data de modificação, donos, grupos, etc.

-n

Usa a identificação de usuário e grupo numérica ao invés dos nomes.

-L, --dereference

Lista o arquivo original e não o link referente ao arquivo.

-o

Usa a listagem longa sem os donos dos arquivos (mesma coisa que -lG).

-p

Mesma coisa que -F, mas não inclui o símbolo '*' em arquivos executáveis. Esta opção é típica de sistemas Linux.

-R

Lista diretórios e sub-diretórios recursivamente.

--full-time

Lista data e hora completa.

Classificação da listagem

A listagem pode ser classificada usando-se as seguintes opções:

-f

Não classifica, e usa -au para listar os arquivos.

-r

Inverte a ordem de classificação.

-c

Classifica pela data de alteração.

-X

Classifica pela extensão.

-U

Não classifica, lista os arquivos na ordem do diretório.

Uma listagem feita com o comando ls -la normalmente é mostrada da seguinte maneira:

-rwxr-xr-- 1 gleydson user 8192 nov 4 16:00 teste

Abaixo as explicações de cada parte:

-rwxr-xr--

São as permissões de acesso ao arquivo teste. A primeira letra (da esquerda) identifica o tipo do arquivo, se tiver um d é um diretório, se tiver um "-" é um arquivo normal.

As permissões de acesso serão explicadas em detalhes mais adiante.

1

Se for um diretório, mostra a quantidade de sub-diretórios existentes dentro dele. Caso for um arquivo, será 1.

gleydson

Nome do dono do arquivo teste.

user

Nome do grupo ao qual o arquivo teste pertence.

8192

Tamanho do arquivo (em bytes).

nov

Mês da criação/última modificação do arquivo.

4

Dia em que o arquivo foi criado.

16:00

Hora em que o arquivo foi criado/modificado. Se o arquivo foi criado há mais de um ano, em seu lugar é mostrado o ano da criação do arquivo.

teste

Nome do arquivo.

Exemplos do uso do comando ls:

- ls - Lista os arquivos do diretório atual.
- ls /bin /sbin - Lista os arquivos do diretório /bin e /sbin
- ls -la /bin - Listagem completa (vertical) dos arquivos do diretório /bin inclusive os ocultos.

pwd

Mostra o nome e caminho do diretório atual.

Você pode usar o comando pwd para verificar em qual diretório se encontra (caso seu aviso de comandos não mostre isso).

mkdir

Cria um diretório no sistema. Um diretório é usado para armazenar arquivos de um determinado tipo. O diretório pode ser entendido como uma *pasta* em que você guarda seus papéis (arquivos). Como uma pessoa organizada, você utilizará uma pasta para guardar cada tipo de documento. Por exemplo, criando um diretório *'vendas'* para guardar seus arquivos relacionados a vendas naquele local.

mkdir [opções] [caminho/diretório] [caminho1/diretório1]

Onde:

caminho

Caminho onde o diretório será criado.

diretório

Nome do diretório que será criado.

opções:

--verbose

Exibe uma mensagem para cada diretório criado. As mensagens de erro serão mostradas mesmo que essa opção não seja usada.

Para criar um novo diretório, você deve ter permissão de gravação. Por exemplo, para criar um diretório em /tmp com o nome de teste que será usado para gravar arquivos de teste, você deve usar o comando mkdir /tmp/teste.

Pode ser criado mais de um diretório com um único comando (mkdir /tmp/teste /tmp/teste1 /tmp/teste2).

2.9.2 cd

Entra em um diretório.Você precisa ter a permissão de execução para entrar no diretório.

cd [diretório]

Onde:

diretório - diretório que deseja entrar.

Exemplos:

- Usando cd sem parâmetros ou cd ~, você retornará ao seu diretório de usuário (diretório home).
- cd /, retornará ao diretório raíz.
- cd -, retornará ao diretório anteriormente acessado.
- cd .., sobe um diretório.
- cd ../[diretório], sobe um diretório e entra imediatamente no próximo (por exemplo, quando você está em /usr/sbin, você digita cd ../bin, o comando cd retorna um diretório (/usr) e entra imediatamente no diretório bin (/usr/bin).

2.9.3 pwd

Mostra o nome e caminho do diretório atual.

Você pode usar o comando pwd para verificar em qual diretório se encontra (caso seu aviso de comandos não mostre isso).

2.9.5 Diretório Raiz

Este é o diretório principal do sistema. Dentro dele estão todos os outros diretórios do sistema. O diretório Raíz é representado por uma “/”. Assim, se você digitar o comando cd / você estará acessando este diretório.

Nele estão localizados outros diretórios como o /bin, /sbin, /usr, /usr/local, /mnt, /tmp, /var e /home. Estes são chamados de sub-diretórios já que estão dentro do diretório “/”. A estrutura de diretórios e sub-diretórios pode ser identificada da seguinte maneira:

- /
- /bin
- /sbin
- /usr
- /usr/local
- /mnt
- /tmp

- /var
- /home

A estrutura de diretórios também é chamada de Árvore de Diretórios porque é parecida com uma árvore de cabeça para baixo. Cada diretório do sistema tem seus respectivos arquivos que são armazenados conforme regras definidas pela FHS (FileSystem Hierarchy Standard - Hierarquia Padrão do Sistema de Arquivos) versão 2.0, definindo que tipo de arquivo deve ser armazenado em cada diretório.

2.9.6 Diretório atual

É o diretório em que nos encontramos no momento. Você pode digitar o comando pwd para verificar qual é seu diretório atual.

O diretório atual também é identificado por um "." (ponto). O comando ls. pode ser usado para listar seus arquivos (é claro que isto é desnecessário porque se não digitar nenhum diretório, o comando ls sozinho listará o conteúdo do diretório atual).

2.9.7 Diretório home

Também chamado de diretório de usuário. Em sistemas GNU/Linux cada usuário (inclusive o root) possui seu próprio diretório onde poderá armazenar seus programas e arquivos pessoais.

Este diretório está localizado em /home/[login]. Neste caso se o seu login for "joao" o seu diretório home será /home/joao. O diretório home também é identificado por um ~ (til), você pode digitar tanto o comando ls /home/joao como ls ~ para listar os arquivos de seu diretório home.

O diretório home do usuário root (na maioria das distribuições GNU/Linux) está localizado em /root.

Dependendo de sua configuração e do número de usuários em seu sistema, o diretório de usuário pode ter a seguinte forma: /home/[primeira_letra_do_nome]/[login], neste caso se o seu login for "joao" o seu diretório home será /home/j/joao. Essa forma só seria realmente útil em um sistema com uma grande quantidade de usuários.

2.9.8 Diretório Superior

O diretório superior *(Upper Directory)* é identificado por .. (2 pontos).

Caso estiver no diretório /usr/local e quiser listar os arquivos do diretório /usr você pode digitar, "ls .." . Ou se você estiver no diretório /usr/local e quiser mudar para o diretório superior /usr basta digitar "cd .." . Ou ainda, se quiser ir direto do diretório /usr/local para o diretório raiz / digite "cd ../.." Este recurso também pode ser usado para copiar, mover arquivos/diretórios, etc.

2.9.9 Diretório Anterior

O diretório anterior é identificado por "-". É útil para retornar ao último diretório usado.

Se estive no diretório /usr/local e digitar cd /lib, você pode retornar facilmente para o diretório /usr/local usando cd -.

2.9.10 Caminho na estrutura de diretórios

O caminho na estrutura de diretórios são os diretórios que teremos que percorrer até chegar ao arquivo ou diretório que procuramos. Se desejar ver o arquivo /usr/doc/copyright/GPL você tem duas opções:

1. Mudar o diretório atual para /usr/doc/copyright com o comando cd /usr/doc/copyright e usar o comando cat GPL

O caminho de diretórios é necessário para dizer ao sistema operacional onde encontrar um arquivo na "árvore" de diretórios.

EXEMPLO

Um exemplo de diretório é o seu diretório de usuário. Todos seus arquivos essenciais devem ser colocados neste diretório.

Um diretório também pode conter outro diretório. Isso é útil quando temos muitos arquivos e queremos melhorar sua organização. Abaixo, um exemplo de uma empresa que precisa controlar os arquivos de pedidos que emite para as fábricas:

/pub/vendas - diretório principal de vendas
/pub/vendas/mes01-99 - diretório contendo vendas do mês 01/1999
/pub/vendas/mes02-99 - diretório contendo vendas do mês 02/1999
/pub/vendas/mes03-99 - diretório contendo vendas do mês 03/1999

- *o diretório vendas é o diretório principal.*
- *mes01-99 é um subdiretório que contém os arquivos de vendas do mês 01/1999.*
- *mes02-99 é um subdiretório que contém os arquivos de vendas do mês 02/1999.*
- *mes03-99 é um subdiretório que contém os arquivos de vendas do mês 03/1999.*

mes01-99, mes02-99, mes03-99 são diretórios usados para armazenar os arquivos de pedidos do mês e ano correspondente. Isso é essencial para organização, pois se todos os pedidos fossem colocados diretamente no diretório vendas, seria muito difícil encontrar o arquivo do cliente "João".

Você reparou que usei a palavra sub-diretório para mes01-99, mes02-99 e mes03-99? É porque eles estão dentro do diretório vendas. Da mesma forma, vendas é um sub-diretório de pub.

---

2. Usar o comando "cat" especificando o caminho completo na estrutura de diretórios e o nome do arquivo: cat /usr/doc/copyright/GPL.

As duas soluções acima permitem que você veja o arquivo GPL. A diferença entre as duas é a seguinte:

- Na primeira, você muda o diretório padrão para /usr/doc/copyright (confira digitando pwd) e depois o comando cat GPL. Você pode listar os arquivos do diretório /usr/doc/copyright com o comando "ls". /usr/doc/copyright é o caminho de diretório que devemos percorrer para chegar até o arquivo GPL.

- Na segunda, é digitado o caminho completo para o comando cat localizar o arquivo GPL: cat /usr/doc/copyright/GPL. Neste caso, você continuará no diretório padrão (confira digitando pwd). Digitando ls, os arquivos do diretório atual serão listados.

2.9.12 Estrutura básica de diretórios do Sistema Linux

O sistema GNU/Linux possui a seguinte estrutura básica de diretórios organizados segundo o FHS (Filesystem Hierarchy Standard):

/bin Contém arquivos programas do sistema que são usados com freqüência pelos usuários.

/boot Contém arquivos necessários para a inicialização do sistema.

/cdrom Ponto de montagem da unidade de CD-ROM.

/dev Contém arquivos usados para acessar dispositivos (periféricos) existentes no computador.

/etc Arquivos de configuração de seu computador local.

/floppy Ponto de montagem de unidade de disquetes

/home Diretórios contendo os arquivos dos usuários.

/lib Bibliotecas compartilhadas pelos programas do sistema e módulos do kernel.

/lost+found Local para a gravação de arquivos/diretórios recuperados pelo utilitário fsck.ext2. Cada partição possui seu próprio diretório lost+found.

/mnt Ponto de montagem temporário.

/proc Sistema de arquivos do kernel. Este diretório não existe em seu disco rígido, ele é colocado lá pelo kernel e usado por diversos programas que fazem sua leitura, verificam configurações do sistema ou modificam o funcionamento de dispositivos do sistema através da alteração em seus arquivos.

/root Diretório do superusuário root.

/sbin Diretório de programas usados pelo superusuário (root) para administração, controle e manutenção do sistema.

/tmp Diretório para armazenamento de arquivos temporários criados por programas.

/usr Contém maior parte de seus programas. Normalmente acessível somente como leitura.

/var Contém maior parte dos arquivos que são gravados com freqüência pelos programas do sistema, como e-mails, spool de impressora, etc.

2.10 Nomeando Arquivos e Diretórios

No GNU/Linux, os arquivos e diretórios podem ter o tamanho de até 255 letras. Você pode identificá-los com uma extensão (um conjunto de letras separadas do nome do arquivo por um “.”).

Os programas executáveis do GNU/Linux, ao contrário dos programas de DOS e Windows, não são executados a partir de extensões .exe, .com ou .bat. O GNU/Linux (como todos os sistemas POSIX) usa a permissão de execução de arquivo para identificar se um arquivo pode ou não ser executado, independente de sua extensão.

Por exemplo, um trabalho de história pode ser identificado mais facilmente caso fosse gravado com o nome trabalho.text ou trabalho.txt. Também é permitido gravar o arquivo com o nome Trabalho de Historia.txt, mas não é recomendado gravar nomes de arquivos e diretórios com espaços porque será necessário colocar o nome do arquivo entre “aspas” para acessá-lo (por exemplo, cat “Trabalho de Historia.txt”). Ao invés de usar espaços, prefira capitalizar o arquivo (usar letras maiúsculas e minúsculas para identifica-lo), como TrabalhodeHistoria.txt.

2.11 Curingas

Curingas (ou referência global) é um recurso usado para especificar um ou mais arquivos ou diretórios do sistema de uma só vez. Este recurso permite que você faça a filtragem do que será listado, copiado e apagado. São usados três tipos de curingas:

- * (asterisco) - Faz referência a um nome completo ou restante, de um arquivo ou diretório.
- ? (ponto de interrogação) - Faz referência a uma letra naquela posição.
- [padrão] - Faz referência a uma faixa de caracteres de um arquivo/diretório. Padrão pode ser:
- [a-z][0-9] - Faz referência a caracteres de a até z seguido de um caractere de 0 até 9.
- [az][1,0] - Faz a referência aos caracteres a e z seguido de um caractere 1 ou 0 naquela posição.
- [a-z,1,0] - Faz referência a intervalo de caracteres de a até z ou 1 ou 0 naquela posição.

A procura de caracteres é “Case Sensitive”. Assim, se você deseja que sejam localizados todos os caracteres alfabéticos você deve usar [a-z,A-Z].

Caso a expressão seja precedida por um ^, faz referência a qualquer caractere exceto o da expressão. Por exemplo, [^abc] faz referência a qualquer caractere exceto a, b e c.

Lembrando que os três tipos de curingas (“*”, “?”, “[]”) podem ser usados juntos.

Existem muitas outras formas de se fazer a mesma coisa. Depende do gosto de cada um. O importante aqui foi mostrar como especificar mais de um arquivo de uma só vez. O uso de curingas será útil ao copiar arquivos, apagar, mover, renomear e nas mais diversas partes do sistema. Aliás, esta é uma característica do GNU/Linux: permitir que a mesma coisa possa ser feita de várias maneiras diferentes.

EXEMPLO

Para entender melhor, vamos à prática:

Vamos dizer que tenha 5 arquivos no diretório /usr/teste, sendo eles teste1.txt, teste2.txt, teste3.txt, teste4.ncl, teste5.ncl.

Caso deseje listar todos os arquivos do diretório /usr/teste você pode usar o coringa "" para especificar todos os arquivos do diretório:*

*cd /usr/teste e ls * ou ls /usr/teste/*.*

Não tem muito sentido usar o comando ls com "", pois todos os arquivos serão listados se o ls for usado sem nenhum Curinga.*

Agora, para listar todos os arquivos com extensão txt (teste1.txt, teste2.txt, teste3.txt), com exceção de teste4.ncl, teste5.ncl, podemos usar inicialmente 3 métodos:

- *Usando o comando ls *.txt, que pega todos os arquivos que começam com qualquer nome e terminam com .txt.*
- *Usando o comando ls teste?.txt, que pega todos os arquivos que começam com o nome teste, tenham qualquer caractere no lugar do coringa ? e terminem com .txt. Com o exemplo acima teste*.txt também faria a mesma coisa, mas se também tivéssemos um arquivo chamado teste10.txt este também seria listado.*
- *Usando o comando ls teste[1-3].txt, que pega todos os arquivos que começam com o nome teste, tenham qualquer caractere entre o número 1-3 no lugar da 6ª letra e terminem com .txt. Neste caso se obtém uma filtragem mais exata, pois o coringa ? especifica qualquer caracter naquela posição e [] especifica números, letras ou intervalo que será usado.*
- *Agora para listar somente teste4.ncl e teste5.ncl podemos usar os seguintes métodos:*
- *ls *.ncl que lista todos os arquivos que terminam com .ncl*
- *ls teste?.ncl que lista todos os arquivos que começam com teste, contenham qualquer caracter na posição do coringa ? e terminem com .ncl.*
- *ls teste[4,5].* que lista todos os arquivos que começam com teste contenham números de 4 e 5 naquela posição e terminem com qualquer extensão.*

2.12 Nomes de dispositivos

Para se evitar que cada programa utilize um I/O *(Input/Output ou Entrada/Saida de dados no programa)* ou IRQ *(Interrupt Request, sinalização de um pedido de interrupção de hardware)* diferente para cada parte do hardware como o mouse ou o modem, no sistema GNU/Linux são usados os nomes de dispositivos. Eles são acessados através do diretório /dev.

Por exemplo: após configurar corretamente o modem, com sua porta I/O 0x2F8 e IRQ 3, ele é identificado automaticamente por /dev/ttyS1. Daqui para frente basta se referir a /dev/ttyS1 para fazer alguma coisa com o modem.

Você também pode fazer um link de /dev/ttyS1 para um arquivo chamado /dev/modem. Dessa forma, você pode fazer a configuração dos seus programas usando /dev/modem ao invés de /dev/ttyS1. Se precisar reconfigurar o seu modem e a porta serial mudar para /dev/ttyS3, será necessário somente apagar o link /dev/modem antigo e criar um novo, apontando para a porta serial /dev/ttyS3. Não será necessário reconfigurar os programas que usam o modem, pois eles estão usando /dev/modem, que está apon-

tando para a localização correta. Isso é muito útil para um bom gerenciamento do sistema.

Acesse http://focalinux.cipsga.org.br/guia/intermediario/ch-hardw.html

2.13 Listando as placas e outros hardwares em um computador

Administradores e técnicos ao configurar uma máquina precisarão saber quais os hardwares ela possui, os periféricos, dispositivos e clock, para configurar tudo e ver a necessidade de atualizações.

Dispositivos PCI/AMR/CNR (como modens, placas de vídeo e áudio) podem ser listados executando o comando cat /proc/pci. Outra forma de listar tais dispositivos é usando o lspci. Se você precisa de mais detalhes como o mapeamento de memória, use lspci -vv.

O mapeamento de memória de dispositivos pode ser mostrado com o comando cat /proc/ioports, ou usando o comando lsdev.

O barramento USB e dispositivos conectados a ele podem ser listados com o comando lsusb ou com cat /proc/bus/usb/devices.

Hardwares disponíveis na máquina, como placa mãe, clock multiplicador, discos, placas diversas, versões e números seriais de dispositivos podem ser mostrados através do comando lshw. Use lshw -html para produzir a listagem em formato HTML, bem interessante para relatórios.

3 Comandos para manipulação de arquivos

Serão apresentados comandos utilizados para manipulação de arquivos neste capítulo.

3.1 cat

Mostra o conteúdo de um arquivo binário ou de texto.

cat [opções] [diretório/arquivo] [diretório1/arquivo1]

Onde:

diretório/arquivo

Localização do arquivo do qual se deseja visualizar o conteúdo.

opções

-n, --number

Mostra o número das linhas enquanto o conteúdo do arquivo é mostrado.

-s, --squeeze-blank

Não mostra mais que uma linha em branco entre um parágrafo e outro.

-

Lê a entrada padrão.

O comando cat trabalha com arquivos texto. Use o comando zcat para ver diretamente arquivos compactados com gzip.

Exemplo: cat /usr/doc/copyright/GPL

3.2 tac

Mostra o conteúdo de um arquivo binário ou de texto (como o cat) só que em ordem inversa.

tac [opções] [diretório/arquivo] [diretório1/arquivo1]

Onde:

diretório/arquivo

Localização do arquivo do qual se deseja visualizar o conteúdo

opções

-s [string]

Usa o [string] como separador de registros.

-

Lê a entrada padrão.

Exemplo: tac /usr/doc/copyright/GPL.

3.3 rm

Apaga arquivos. Também pode ser usado para apagar diretórios e sub-diretórios vazios ou que contenham arquivos.

rm [opções][caminho][arquivo/diretório] [caminho1][arquivo1/diretório1]

Onde:

caminho

Localização do arquivo que se deseja apagar. Se omitido, assume que o arquivo esteja no diretório atual.

arquivo/diretório

Arquivo que será apagado.

opções

-i, --interactive

Pergunta antes de remover, esta é ativada por padrão.

-v, --verbose

Mostra os arquivos na medida que são removidos.

-r, --recursive

Usado para remover arquivos em sub-diretórios. Esta opção também pode ser usada para remover sub-diretórios.

-f, --force

Remove os arquivos sem perguntar.

-- arquivo

Remove arquivos/diretórios que contém caracteres especiais. O separador "--" funciona com todos os comandos do shell e permite que caracteres especiais, como "*", "?", "-", sejam interpretados como caracteres comuns.

Use com atenção o comando rm, pois uma vez que arquivos e diretórios são apagados, eles não podem mais ser recuperados.

- Exemplosrm teste.txt - Apaga o arquivo teste.txt no diretório atual.
- rm *.txt - Apaga todos os arquivos do diretório atual que terminam com .txt.
- rm *.txt teste.novo - Apaga todos os arquivos do diretório atual que terminam com .txt e também o arquivo teste.novo.
- rm -rf /tmp/teste/* - Apaga todos os arquivos e sub-diretórios do diretório /tmp/teste mas mantém o sub-diretório /tmp/teste.
- rm -rf /tmp/teste - Apaga todos os arquivos e sub-diretórios do diretório /tmp/teste, inclusive /tmp/teste.
- rm -f -- --arquivo-- - Remove o arquivo de nome --arquivo--.

3.4 cp
Copia arquivos.

cp [opções] [origem] [destino]

Onde:

origem

Arquivo que será copiado. Podem ser especificados mais de um arquivo para ser copiado usando “Curingas”, já vistos neste material.

destino

O caminho ou nome de arquivo onde será copiado. Se o destino for um diretório, os arquivos de origem serão copiados para dentro do diretório.

opções

-i, --interactive

Pergunta antes de substituir um arquivo existente.

-f, --force

Não pergunta, substitui todos os arquivos caso já existam.

-r

Copia arquivos dos diretórios e subdiretórios da origem para o destino. É recomendável usar -R ao invés de -r.

-R, --recursive

Copia arquivos e sub-diretórios (como a opção -r) e também os arquivos especiais FIFO e dispositivos.

-v, --verbose

Mostra os arquivos enquanto estão sendo copiados.

-s, --simbolic-link

Cria link simbólico ao invés de copiar.

-l, --link

Faz o link no destino ao invés de copiar os arquivos.

-p, --preserve

Preserva atributos do arquivo, se for possível.

-u, --update

Copia somente se o arquivo de origem é mais novo que o arquivo de destino ou quando o arquivo de destino não existe.

-x

Não copia arquivos que estão localizados em um sistema de arquivos diferente de onde a cópia iniciou.

O comando cp copia arquivos da ORIGEM para o DESTINO. Ambos, origem e destino, terão o mesmo conteúdo após a cópia.

- Exemplos: cp teste.txt teste1.txt - Copia o arquivo teste.txt para teste1.txt.
- cp teste.txt /tmp - Copia o arquivo teste.txt para dentro

do diretório /tmp.

- cp * /tmp - Copia todos os arquivos do diretório atual para /tmp.
- cp /bin/* . - Copia todos os arquivos do diretório /bin para o diretório em que nos encontramos no momento.
- cp -R /bin /tmp - Copia o diretório /bin e todos os arquivos/subdiretórios existentes para o diretório /tmp.
- cp -R /bin/* /tmp - Copia todos os arquivos do diretório /bin (exceto o diretório /bin) e todos os arquivos/subdiretórios existentes dentro dele para /tmp.
- cp -R /bin /tmp - Copia todos os arquivos e o diretório /bin para /tmp.

3.5 mv

Move ou renomeia arquivos e diretórios. O processo é semelhante ao do comando cp, mas o arquivo de origem é apagado após o término da cópia.

mv [opções] [origem] [destino]

Onde:

origem

Arquivo/diretório de origem.

destino

Local para onde será movido ou novo nome do arquivo/diretório.

opções

-f, --force

Substitui o arquivo de destino sem perguntar.

-i, --interactive

Pergunta antes de substituir. É o padrão.

-v, --verbose

Mostra os arquivos que estão sendo movidos.

-u, --update

Move somente se o arquivo de origem for mais novo que o de destino, ou se este não existir.

O comando mv copia um arquivo da ORIGEM para o DESTINO (semelhante ao cp), mas após a cópia, o arquivo de ORIGEM é apagado.

- Exemplos: mv teste.txt teste1.txt - Muda o nome do arquivo teste.txt para teste1.txt.
- mv teste.txt /tmp - Move o arquivo teste.txt para /tmp. Lembre-se que o arquivo de origem é apagado após ser movido.
- mv teste.txt teste.new (supondo que teste.new já exista) - Copia o arquivo teste.txt por cima de teste.new e apaga teste.txt após terminar a cópia.

4 Execução de Programas

Este capítulo explica como executar programas no GNU/Linux e o uso das ferramentas de controle de execução dos programas.

4.1 Executando um comando/programa

No aviso de comando # (root) ou $ (usuário), digite o nome do comando e tecle Enter. O programa/comando vai ser executado e receberá um número de identificação (chamado de PID - Process Identification). Esse número identifica o processo no sistema e dessa maneira tem um controle sobre sua execução (será visto mais adiante neste capítulo).

Para rodar um comando, é necessário que ele tenha permissões de execução (mais detalhes serão vistos mais adiante) e que esteja no caminho de procura de arquivos, como visto na próxima seção.

Todo o programa recebe uma identificação de usuário (UID) quando é executado, o que determina quais serão suas permissões de acesso. O programa normalmente usa o UID do usuário que o executou ou o usuário configurado pelo bit de permissão de acesso, caso este estiver definido. Também existem programas que são executados como root e modificam sua identificação de usuário para algum que tenha menos privilégios no sistema (como o Apache, por exemplo). Mais informações serão dadas posteriormente.

Exemplos de comandos: ls, df, pwd.

4.2 path

Path é o caminho de procura dos arquivos/comandos executáveis. O path (caminho) é armazenado na variável de ambiente PATH. Você pode ver o conteúdo dela usando o comando echo $PATH.

Por exemplo, o caminho */usr/local/bin:/usr/bin:/bin:/usr/bin/X11* significa que, se você digitar o comando ls, o interpretador de comandos iniciará a procura do programa ls no diretório /usr/local/bin. Caso não encontre o arquivo, ele inicia a procura em /usr/bin, e assim sucessivamente, até que encontre o arquivo procurado.

Se o interpretador de comandos chegue até o último diretório do path e não encontre o arquivo/comando digitado, é mostrada a seguinte mensagem:

bash: ls: command not found (comando não encontrado).

O caminho de diretórios vem configurado na instalação do Linux, mas pode ser alterado no arquivo /etc/profile. Caso deseje alterar o caminho para todos os usuários, esse arquivo é o melhor lugar, pois

ele é lido por todos os usuários no momento do login.

Se um arquivo/comando não está localizado em nenhum dos diretórios do path, você deve executá-lo usando um ./ na frente do comando.

Se deseja alterar o path para um único usuário, modifique o arquivo .bash_profile no seu diretório de usuário (home).

OBSERVAÇÃO: Por motivos de segurança, não inclua o diretório atual $PWD no path.

4.3 Tipos de Execução de comandos/programas

1. Um programa pode ser rodado de duas formas: Primeiro Plano - Também chamado de *foreground*. Você deve esperar o término da execução do programa para rodar um novo comando. Somente é mostrado o aviso de comando após o término de sua execução.

2. Segundo Plano - Também chamado de *background*. Você não precisa esperar o término da execução do programa para executar um novo comando. Após iniciar um programa em *background*, é mostrado um número PID (identificação do Processo) e o aviso de comando é novamente mostrado, permitindo o uso normal do sistema.

O programa rodado em background continua sendo executado internamente. Após ser concluído, o sistema retorna uma mensagem de pronto, acompanhado do número PID do processo que terminou.

Para iniciar um programa em primeiro plano, basta digitar seu nome normalmente. Para iniciar um programa em segundo plano, acrescente o caractere "&" após o final do comando.

OBS: Mesmo que um usuário execute um programa em segundo plano e saia do sistema, o programa continuará rodando até que seja concluído ou finalizado pelo usuário que iniciou a execução (ou pelo usuário root).

Exemplo: find / -name boot.b &

O comando será executado em segundo plano e deixará o sistema livre para outras tarefas. Após o comando find terminar, será mostrada uma mensagem.

4.4 Executando programas em sequência

Os comandos podem ser executados em sequência (um após o término do outro) se os separarmos com ";".

Por exemplo: echo primeiro;echo segundo;echo terceiro

4.5 ps

Algumas vezes é útil ***ver quais processos estão sendo executados no computador.*** O comando ps faz isso, e também nos mostra qual usuário executou o programa, a hora que o processo foi iniciado e várias outras informações.

ps [opções]

Onde as *opções* podem ser:

a

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

x

Mostra processos que não são controlados pelo terminal.

u

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

m

Mostra a memória ocupada por cada processo em execução.

f

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

e

Mostra variáveis de ambiente no momento da inicialização do processo.

w

Mostra a continuação da linha atual na próxima linha, ao invés de cortar o restante que não couber na tela.

--sort:[coluna]

Organiza a saída do comando ps de acordo com a coluna escolhida. Você pode usar as colunas pid, utime, ppid, rss, size, user, priority.

As opções acima podem ser combinadas para resultar numa listagem mais completa. Você também pode usar pipes "|" para filtrar a saída do comando ps. Mais detalhes sobre pipe serão vistos mais adiante.

Ao contrário de outros comandos, o comando ps não precisa do hífen "-" para especificar os comandos. Isso porque ele não utiliza opções longas e não usa parâmetros.

EXEMPLO: *digite em seu terminal*

ps au

Você verá na tela todos os processos criados por você e por outros usuários, além de mostrar nome de usuário, data e hora em que o processo foi iniciado.

4.6 top

Mostra os programas em execução ativos, parados, tempo usado na CPU, detalhes sobre o uso da memória RAM, disponibilidade para execução de programas no sistema e várias

outras informações relativas aos programas em execução.

top é um programa que continua em execução, mostrando continuamente os processos que estão rodando em seu computador e os recursos utilizados por eles. ***Para sair do top, pressione a tecla q.***

top [opções]

Onde são *opções:*

-d [tempo]

Atualiza a tela após o [tempo] (em segundos).

-s

Diz ao top para ser executado em modo seguro.

-i

Inicia o top, ignorando o tempo de processos zumbis.

-c

Mostra a linha de comando ao invés do nome do programa.

A ajuda sobre o top pode ser obtida dentro do programa . Para isso, pressione a tecla h ou vá até a página de manual (man top).

Veja abaixo algumas teclas úteis:

- espaço - Atualiza imediatamente a tela.
- CTRL+L - Apaga e atualiza a tela.
- h - Mostra a tela de ajuda do programa. São mostradas todas as teclas que podem ser usadas com o top.
- i - Ignora o tempo ocioso de processos zumbis.
- q - Sai do programa.
- k - Finaliza um processo - semelhante ao comando kill. Você será perguntado pelo número de identificação do processo (PID). Esse comando não estará disponível caso esteja usando o top com a opção -s.
- n - Muda o número de linhas mostradas na tela. Se 0 for especificado, será usada toda a tela para listagem de processos.

4.7 Controle de execução de processos

Seguem alguns comandos e métodos úteis para o controle da execução de processos no GNU/Linux.

4.7.1 Interrompendo a execução de um processo

Para cancelar a execução de algum processo rodando em primeiro plano, basta pressionar as teclas ***CTRL+C***. A execução do programa será cancelada e será mostrado o aviso de comando. Você também pode usar o comando kill, descrito logo adiante, para interromper um processo sendo executado.

4.7.2 Parando momentaneamente a execução de um processo

Para parar a execução de um processo rodado em primeiro plano, basta pressionar as teclas ***CTRL+Z***. O programa em execução será pausado e será exibido o número de seu job e o aviso de comando.

Para retornar à execução de um comando pausado, use os comandos fg ou bg, descritos à frente.

O programa permanece na memória no ponto de processamento em que parou quando ele é interrompido. Você pode usar outros comandos ou rodar outros programas enquanto o programa atual está interrompido.

4.7.3 jobs

O comando 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, como já visto neste material, ou através do comando bg.

O número de identificação de cada processo parado ou em segundo plano (job) é usado com os comandos fg e bg. Um processo interrompido pode ser finalizado usando-se o comando kill %[num], em que [num] é o número do processo obtido pelo jobs.

4.7.4 fg

Permite fazer com que um programa rodado em segundo plano ou parado, rode em primeiro plano. Você deve usar o comando jobs para pegar o número do processo executado em segundo plano ou interrompido.

fg [número]

Em que número é o número obtido através do comando jobs.

Caso seja executado sem parâmetros, o fg utilizará o último programa interrompido (o maior número obtido com o comando jobs).

Exemplo: fg 1.

4.7.5 bg

Permite fazer um programa rodando em primeiro plano ou parado, rodar em segundo plano. Para fazer um programa em primeiro plano rodar em segundo, é necessário primeiro interromper a execução do comando com ***CTRL+ Z***. Será mostrado o número da tarefa interrompida. Use esse número com o comando *bg* para iniciar a execução do comando em segundo plano.

bg [número]

Onde: número é o número do programa obtido com o pressionamento das teclas CTRL+Z ou através do comando jobs.

4.7.6 kill

Permite enviar um sinal a um comando/programa. Se usado sem

parâmetros, ***o kill enviará um sinal de término ao processo sendo executado.***

kill [opções] [sinal] [número]

Onde:

número

É o número de identificação do processo obtido com o comando ps. Também pode ser o número após o sinal de % obtido pelo comando jobs para matar uma tarefa interrompida.

sinal

Sinal que será enviado ao processo. Se omitido usa -15 como padrão.

opções

-9

Envia um sinal de destruição ao processo ou programa. Ele é terminado imediatamente, sem chances de salvar os dados ou apagar os arquivos temporários criados.

Você precisa ser o dono do processo ou o usuário root para terminá-lo ou destruí-lo. Para verificar se o processo foi finalizado, utilize o comando ps. Os tipos de sinais aceitos pelo GNU/Linux serão explicados em detalhes mais adiante.

Exemplo: kill 500, kill -9 500, kill %1.

4.7.7 killall

Permite finalizar processos através do nome.

killall [opções] [sinal] [processo]

Onde:

processo

Nome do processo que deseja finalizar

sinal

Sinal que será enviado ao processo

opções

-i

Pede confirmação sobre a finalização do processo.

-l

Lista o nome de todos os sinais conhecidos.

-q

Ignora a existência do processo.

-v

Retorna se o sinal foi enviado com sucesso ao processo.

-w

Finaliza a execução do killall somente após finalizar todos os processos.

Os tipos de sinais aceitos pelo GNU/Linux são explicados em

detalhes logo adiante.

Exemplo: killall -HUP inetd

4.7.9 Sinais do Sistema

Os sinais são meios usados para que os processos possam se comunicar e para que o sistema possa interferir em seu funcionamento. Por exemplo, se o usuário executar o comando kill para interromper um processo, isso será feito por meio de um sinal.

Quando um processo recebe um determinado sinal e conta com instruções sobre o que fazer com ele, tal ação é colocada em prática. Se não houver instruções pré-programadas, o próprio Linux pode executar a ação de acordo com suas rotinas.

O GNU/Linux suporta os sinais listados pelo comando kill –l .

Podemos saber qual a função de cada sinal consultando a man page signal,com o comando: man signal. Veja a função de alguns:

- **SIGSTOP -** interrompe a execução de um processo e só reativá-lo após o recebimento do sinal CONT;
- **SIGCONT -** instrui a execução de um processo após este ter sido interrompido;
- **SIGSEGV -** informa erros de endereços de memória;
- **SIGTERM -** termina completamente o processo, ou seja, este deixa de existir após a finalização;
- **SIGILL -** informa erros de instrução ilegal, por exemplo, quando ocorre divisão por zero;
- **SIGKILL -** “mata” um processo e é usado em momentos de criticidade.

4.7.8 killall5

Envia um sinal de finalização para todos os processos sendo executados.

killall5 [sinal]

4.8 nohup

Executa um comando ignorando os sinais de interrupção. Esse comando pode ser executado até mesmo em segundo plano, caso seja feito o logout do sistema.

nohup [comando que será executado]

As mensagens de saída do nohup são direcionadas para o arquivo $HOME/nohup.out.

Exemplo: nohup find / -uid 0 >/tmp/rootfiles.txt &.

4.9 nice e renice

nice configura a prioridade da execução de um comando/programa.

nice [opções] [comando/programa]

Onde:

comando/programa

Comando/programa que terá sua prioridade ajustada.

opções

-n [numero]

Configura a prioridade com a qual o programa será executado. Se um programa for executado com maior prioridade, ele usará mais recursos do sistema para seu processamento. Caso tenha uma prioridade baixa, ele permitirá que outros programas tenham preferência. A prioridade de execução de um programa/comando pode ser ajustada de -19 (a mais alta) até 19 (a mais baixa).

Exemplo: nice -n -19 find / -name apropos.

renice troca a prioridade das aplicações em execução.

renice [prioridade] [pid]

Onde:

prioridade

Prioridade a ser configurada para o processo. Pode ser ajustada de -19 (a mais alta) até 19 (a mais baixa).

pid

pid do processo a ser configurado.

Exemplo: renice +19 1000

4.10 fuser

Permite identificar e fechar os processos que estão utilizando arquivos e soquetes no sistema.

fuser [opções] [nome]

Onde:

nome

Especifica um nome de processo, diretório, arquivo, etc.

opções

-k

Finaliza os processos acessando o arquivo especificado. O sinal desejado deve ser especificado com a opção -signal [num], ou o sinal -9 será enviado como padrão. Não é possível matar o próprio processo fuser.

-i

Pergunta antes de destruir um processo. Será ignorada caso a opção -k não seja especificada.

-l

Lista todos os nomes de sinais conhecidos.

-m [nome]

Especifica um arquivo em um sistema de arquivos montado ou dispositivo de bloco que está montado. Serão listados todos os processos acessando aquele sistema de arquivos.

Diretórios são mostrados seguidos de uma /

-signal [número]

Usa o sinal especificado ao invés de -9 (SIGKILL) quando finalizar processos.

-u

Acrescenta o nome do dono de cada processo ao PID.

-v

Os processos são mostrados em um estilo idêntico ao ps.

4.11 tload

Representa de forma gráfica a carga do sistema.

tload [opções]

Onde:

opções

-s [número]

Mostra uma escala vertical com espaçamento especificado por [número]. É recomendável o uso de números entre 1 e 10 para melhor visualização da escala.

-d [número]

Especifica o intervalo entre atualizações, em segundos.

4.12 vmstat

Mostra estatísticas sobre o uso da memória virtual do sistema.

vmstat [intervalo] [contagem]

Onde:

intervalo

Número especificado em segundos entre atualizações.

contagem

Número de vezes que será mostrado.

Se não for especificado nenhum parâmetro, o vmstat mostra o status da memória virtual e volta imediatamente para a linha de comando. A descrição dos campos do vmstat são as seguintes:

Processos

r

Número de processos aguardando execução.

b

Número de processos em espera não interrompíveis.

w

Número de processos extraídos do arquivo de troca ou caso contrário em execução.

Memória

swpd

A quantidade de memória virtual usada em Kb.

free

Quantidade de memória livre em Kb.

buff

Quantidade de memória usada como buffer em Kb.

Memória Virtual

si

Quantidade de memória gravada para o disco em Kb/s.

so

Quantidade de memória retirada do disco em Kb/s.

Entrada/Saída

bi

Blocos enviados para um dispositivo de bloco (medido em blocos por segundo).

bo

Blocos recebidos de um dispositivo de bloco (em blocos por segundo).

Sistema

in

Número de interrupções por segundo, incluindo o clock.

cs

Número de mudanças de contexto por segundo.

Porcentagem do total de tempo da CPU

us

Tempo do usuário

sy

Tempo do sistema

id

Tempo ocioso

4.13 pidof

Retorna o PID do processo especificado

pidof [opções] [nome]

Onde:

nome

Nome do processo para o qual se deseja obter o número PID

opções

-s

Retorna somente o primeiro PID encontrado.

-x

Retorna o PID do shell que está executando o script

-o [PID]

Ignora o processo com aquele PID. O PID especial %PPID pode ser usado para nomear o processo pai do programa pidof

OBS: O programa pidof é um link simbólico ao programa killall5. Cuidado ao executar o killall5, pois as funções e opções são completamente diferentes, dependendo da forma como é chamado na linha de comando!

Exemplo: pidof -s init

4.14 pstree

Mostra a estrutura de processos em execução no sistema em forma de árvore.

pstree [opções] [pid]

Onde:

pid

Número do processo que terá sua árvore listada. Se omitido, lista todos os processos.

opções

-a

Mostra opções passadas na linha de comando.

-c

Mostra toda a estrutura (inclusive sub-processos do processo pai).

-G

Usa caracteres gráficos no desenho da árvore de processos.

-h

Destaca o processo atual e seus antecessores.

-H [pid]

Destaca o processo especificado.

-l

Não faz quebra de linha

-n

Classifica pelo número PID ao invés do nome.

-p

Mostra o número PID entre parênteses após o nome do processo.

-u

Mostra também o dono do processo.

-U

Usa o conjunto de caracteres Unicode para o desenho da árvore.

4.15 Fechando um programa quando não se sabe como sair

Quando você ainda é um iniciante no GNU/Linux, talvez execute um programa e não saiba como fechá-lo. Este capítulo do material pretende ajudá-lo a resolver esse tipo de problema.

Isso também pode ocorrer com programadores que estão construindo seus programas e por algum motivo não implementam uma opção de saída. Ou implementam, mas ela não funciona.

1. Em nosso exemplo, vou supor que executamos um programa em desenvolvimento com o nome contagem (que conta o tempo, em segundos, a partir do momento que é executado), mas que o programador esqueceu de colocar uma opção de saída. Normalmente, todos os programas UNIX podem ser interrompidos com o pressionamento das teclas <CTRL> e <C>. Tente isso, primeiramente, para finalizar um programa. Se você estiver usando um Editor de Texto, provavelmente não irá funcionar (ele vai entender como um comando de menu). Normalmente, dá certo para comandos que são executados e terminados sem a intervenção do usuário.

Se não der certo,

2. Mude para um novo console (pressionando <ALT> e <F2>), e faça o login como usuário **root**.

3. Localize o PID (número de identificação do processo) usando o comando *ps ax*. Aparecerão várias linhas, cada uma com o número do processo na primeira coluna, e a linha de comando do programa na última coluna. Caso apareçam vários processos, você pode usar ps ax|grep contagem. Nesse caso, o grep fará uma filtragem da saída do comando ps ax, mostrando somente as linhas que tem a palavra “contagem”.

4. Feche o processo, usando o comando kill PID (lembre-se de substituir PID pelo número encontrado pelo comando ps ax acima).

O comando acima envia um sinal de término de execução para o processo (neste caso, o programa contagem). O sinal de término mantém a chance do programa salvar seus dados ou apagar os arquivos temporários que criou e então ser finalizado.

5. Alterne para o console em que estava executando o programa contagem e verifique se ele ainda está rodando. Se ele estiver parado, mas o aviso de comando não estiver disponível, pressione a tecla <ENTER>. Frequentemente acontece o mesmo com o comando kill: você finaliza um programa, mas o aviso de comando não é mostrado até que se pressione <ENTER>.

6. Caso o programa ainda não foi finalizado, repita o comando kill usando a opção -9: kill -9 PID. Este comando envia um sinal de DESTRUIÇÃO do processo, fazendo ele terminar “na marra”!

Uma última dica: todos os programas estáveis (todos que

acompanham as boas distribuições GNU/Linux) têm sua opção de saída. Lembre-se que, quando um processo é finalizado, todos os dados do programa em execução podem ser perdidos (principalmente se estiver num editor de textos), mesmo usando o kill sem o parâmetro -9.

Procure a opção de saída de um programa consultando o help on line, as páginas de manual, a documentação que acompanha o programa, info pages. Detalhes de como encontrar a ajuda dos programas serão vistos mais à frente.

4.16 Eliminando caracteres estranhos

Às vezes, quando um programa mal comportado é finalizado ou quando você visualiza um arquivo binário através do comando cat, é possível que o aviso de comando (prompt) volte com caracteres estranhos.

Para fazer tudo voltar ao normal, basta digitar reset e teclar ENTER. Não se preocupe, o comando reset não reiniciará seu computador (como o botão reset do seu computador faz). Ele apenas fará tudo voltar ao normal.

Note que, enquanto você digitar reset, aparecerão caracteres estranhos ao invés das letras. Não se preocupe! Basta digitar corretamente e bater ENTER. O aviso de comando voltará ao normal.