Quotas: No Linux existe a possibilidade de habilitarmos quotas para gerenciarmos melhor o uso do espaço disponível em disco. Nele é possível definirmos a quantidade de espaço em disco para cada usuário ou grupo de usuário. Contudo vale observar que é preciso habilitar o gerenciamento de quotas antes de utilizarmos este recurso. Veja os passos básicos:

1. É preciso que o suporte a quotas esteja compilado no kernel.
2. Edite /etc/fstab e adicione o gerenciamento de quota para usuário (usrquota) e para grupo (grpquota).
3. Crie o arquivo quota.user e quota.group em /home e configure as permissões de leitura e escrita para superusuário.
4. Execute *quotacheck -avug* para iniciar o banco de dados recém-criado.
5. Verifique se o banco de dados foi iniciado. (*ls -lga /home*).
6. Habilite o serviço de quota. #quota -a
7. Tenha certeza de que o serviço de quotas está habilitado no boot. Se não existir o arquivo /etc/rc.d/quotas, crie um script e abra as permissões para 755. Por fim crie um link simbólico para o arquivo de quotas para runlevel 3 e runlevel 5.
8. Faça checagem do sistema de quotas uma vez por semana colocando o comando *quotacheck* no cron do root.
   # crontab -e

Existem quatro tipos de limites de quotas a saber:

1. Limite Físico por Usuário (*user hard limit*).
2. Limite Leves por Usuário (*user soft limit*).
3. Limite Físico por Grupo (*group hard limit*).
4. Limite Leve por Grupo (*group soft limit*).

Por fim vale observar que pode-se configurar o gerenciado de quotas para um Período de Graça (*Grace period*).

Gerenciando de processos

- O conceito de multitarefa significa que o Linux é capaz de executar diversos programas e serviços ao mesmo tempo de forma preemptiva. Se tivermos apenas um processador central no computador com Linux, o sistema fará o escalonamento (rodízio) dos processos de forma tão eficiente que o usuário terá a impressão de que ele pode executar mais de um programa ao mesmo tempo.
- Quando um computador com Linux é ligado, o sistema imediatamente procura no setor de boot do disco rígido o gerenciador de boot.
- Após a carga do kernel, este inicia um processo especial chamado *init*. Este processo é o pai de todos os processos e responsável pelo restante da carga do boot do Linux.
- Depois da carga do boot, o *init* chama outro programa especial chamado getty, que é responsável pela autenticação dos usuários e por iniciar o processo de shell.
- Cada programa é pelo menos um processo e cada processo possui alguns atributos como:
  - ***Process ID (PID)***: Cada processo possui um número de identificação único.
  - ***User ID e Group ID*** (ID do usuário e ID do grupo): Os processos precisam ser executados com os privilégios de uma conta de usuário e grupo associado a eles.
  - ***Processo pai***: No Linux nenhum processo é executado de forma independente dos outros. Todos os processos no sistema, com exceção do init, possuem um processo pai, que é responsável pela sua execução.
  - ***Parent ID*** (ID do processo pai): Este atributo grava o PID do processo pai. Caso o processo pai termine sua execução antes do processo filho, o processo filho é "apadrinhado" pelo init, ganhando o Parent ID igual a 1.
  - Variáveis de ambiente: Cada processo herda do processo pai algumas variáveis de ambiente que simplesmente guardam alguns valores que podem ou não ser importantes para o processo em execução.
  - Diretório de trabalho: Os processos também são associados a um diretório de trabalho, onde podem fazer leitura e escrita de disco.
  - Temporizadores: O Kernel mantém registros da hora em que os processos são criados bem como o tempo de CPU que eles consomem durante a sua execução.

Sinais: Cada processo no Linux fica à escuta de sinais. Estes sinais são utilizados pelo Kernel, por outros processos ou pelo usuário para avisar a um determinado processo sobre algum evento em particular.
Quando um sinal é enviado para um processo, ele toma uma determinada ação dependendo do valor que este sinal carrega.
Controle de Processos: Para que um processo execute em segundo plano, é preciso que ele não espere por nenhuma ação do usuário.
Basicamente para colocar qualquer processo em segundo plano de execução, basta adicionar o caractere "&" no final da linha de comando que executará o processo:
Prioridade de Execução: O escalonador de processos é um algoritmo especial que coloca em fila todos os processos em execução e decide qual processo irá ser executado e durante quanto tempo. A lista de prioridades pode ser vista com o comando ps -lax na coluna PRI.
Comando kill Vs. Killall: O primeiro envia sinais para um ou mais processos identificados pelo PID. O segundo envia sinais para todos os processos na fila de execução que possuem um determinado nome.
Processos Zoombie: É possível, dentro de uma hierarquia de processos, que um determinado processo filho termine por algum motivo inesperado, e o processo pai se torne um processo zoobie ou difundo (defunct). Os processos zoobie não podem ser terminados com o comando kill, por que ele já não existem mais.
Isso acontece porque cada processo criado recebe um lugar na tabela de processos do Kernel. Quando ele termina, seu lugar na tabela do Kernel recebe o resultado da sua execução. O resultado da execução é retido na tabela até alguém consultá-lo quando, então, é removido da tabela.
O estado do processo é chamado zumbi quando o mesmo termina e seu resultado ainda não foi retirado da tabela do Kernel.

Sistema de Boot, Shutdown e Runlevels

Durante o *boot*, o BIOS realiza uma série de testes, cuja função é determinar com exatidão os componentes de hardware instalados no sistema. Este teste é chamado de ***POST*** *(power-on self test)*. Logo depois dos testes de *hardware*, o BIOS procura nos dispositivos de discos rígidos, disquetes e CD-ROM um endereço especial chamado de ***setor de boot***.
O Linux possui dois gerenciadores de *boot*, o LILO e o GRUB. Ambos os gerenciadores permitem que alguns parâmetros sejam passados para o *Kernel* durante a sua carga. Estes parâmetros podem passar algumas informações que ele pode não ser capaz de buscar por conta própria ou até mesmo mudar seu comportamento.
Quando os parâmetros para o *Kernel* são passados pela linha de comando, eles seguem o seguinte padrão:

- *Nome_da_imagem_do_kernel parametro=valor*
- *Nome_da_imagem_do_kernel parametro*

O gerenciador de *boot* permite que as informações sejam passadas para o *Kernel*, que irá lidar com os módulos do sistema, mas ele não lida diretamente com eles. O *Kernel* do Linux é modular, isto é, permite que módulos (também podem ser chamados *drivers*) possam ser compilados na imagem do *Kernel* ou carregados somente quando necessário.
Para enviar parâmetros para os módulos do *Kernel*, é preciso editar o arquivo de configuração dos módulos, que dependendo da distribuição pode ser /etc/conf.modules ou /etc/modules.conf.
Processo de carga do *Kernel*: Durante o processo de *boot* o Linux executa diversos processos em uma ordem programada chamada de "*nível de execução*", ou ***RunLevels***.
No Linux exitem sete níveis de execução predefinidos, de 0 a 6. Os serviços que cada *runlevel* pode executar vai depender da distribuição do Linux e da configuração que o administrador do sistema efetuou.
Por padrão, as distribuições utilizam o *nível 0* para uma sequencia de desligamento (*shutdown*) elegante e o *nível 6* para o *reboot*.
O *nível 1* também conhecido como monousuário é utilizado para *manutenções do sistema*, por exemplo, recuperar uma partição de dados com problemas. O administrador do sistema pode a qualquer momento alterar o nível de execução através do comando *init* seguido do número do *runlevel* desejado.

Administração do Sistema

Em sistema operacional multiusuário, é comum que o administrador gaste grande parte do seu tempo configurando contas de usuário. As contas de usuário são gravas em um arquivo especial chamado /etc/passwd. Neste arquivo são gravados dados como login do usuário, senha, identificador, grupo a que ele pertence, nome do usuário, diretório home e shell.
Diretório /etc/skel (Esqueleto): O diretório /etc/skel é utilizado para fornecer a estrutura básica do diretório home do usuário. Pode-se definir alguns arquivos e diretórios que todos os novos usuários devem ter no momento de sua criação.
Note que o conteúdo do *skel* somente vai ser copiado quando for utilizado a opção "-m" do comando *useradd*.
Arquivos de iniciação do Shell: Durante o processo de logon de um usuário, quando o shell *bash* inicia, ele executa o script /etc/profile. Este script pode ser customizado e diferente em cada distribuição. Sua função fundamental é configurar algumas variáveis de ambiente e fazer a sintonia do sistema para os usuários.

Arquivos de log

O Linux possui um serviço especial chamado *syslog* que faz todo trabalho sujo de filtrar e gravar mensagens importantes não só no sistema local, mas de outros sistemas que suportam o *syslog*. Além de gravar em arquivos, as mensagens podem ser enviadas para a tela de console e telas de terminal.
O arquivo de configuração /etc/syslog.conf controla o que o *syslog* vai gravar e onde. O "o que" vai ser gravado é chamado de facilidade. As facilidades são na verdade a origem da mensagem. O *syslog* trabalha com o padrão de sistema de filtros. Todas as mensagens com o nível especificado e superiores são registados de acordo com as opções usadas.