**Laboratórios Extras**

**Parte 1 - Fase de Criação**  
  
Para fazer os exercícios que iremos praticar agora, é interessante você estar com 4 terminais abertos para não se perder ou ficar confuso, mas isso não é obrigatório, pois é possível fazer isso com um único terminal!  
  
Vamos fazer em terminais em modo texto, isto é, 4 shells (TTY1, TTY2, TTY3 e TTY4).  
  
Para acessar o terminal modo texto 1 (tty1) faça:  
CTRL + ALT + F1  
  
Para acessar o terminal modo texto 2 (tty2) faça:  
CTRL + ALT + F2  
  
Para acessar o terminal modo texto 3 (tty3) faça:  
CTRL + ALT + F3  
  
Para acessar o terminal modo texto 4 (tty4) faça:  
CTRL + ALT + F4  
  
Para voltar ao modo gráfico:  
CTRL + ALT + F7  
  
  
Irei criar um exemplo de diretório público, e para implantar essa política, iremos usar quase todos os comandos vistos na aula!  
  
Assim conseguimos ter uma lógica do que foi visto...  
  
Nosso exemplo será assim...  
  
Vamos imaginar que temos vários usuários na máquina e eles irão acessar um diretório público, que terá os documentos de uma determinada área da minha empresa...  
  
Então terei usuários que poderão acessar esse diretório, e teremos usuários que não poderão...   
  
Quando falamos de diretório compartilhado, o local correto para armazená-lo seria no /mnt  
  
Ou podemos ter uma partição reservada para ele! O que seria o mais interessante! ois assim consigo deixá-la independente do sistema e ao mesmo tempo limito seu tamanho com cotas de disco(assunto que veremos adiante).  
  
No primeiro terminal estaremos como root.  
  
Lembrando que vamos usar 4 terminais:  
1 - root  
2 - Usuário debian   
3 - Usuário gnu   
4 - Usuário tux   
  
Os usuários debian, gnu e tux serão criados ainda.  
  
Então no primeiro terminal como root, vamos criar um diretório público:  
#cd /mnt  
#mkdir documentos  
  
Dando um ls -ld nesse diretório veremos que o dono do mesmo é root e seu grupo tb é root:  
# ls -ld documentos/  
drwxr-xr-x 2 root root 4096 Nov 16 14:42 documentos/  
  
Agora, tenho que criar o grupo que será definido para esse diretório, para que um conjunto de usuários tenha acesso ao mesmo.  
  
Criando o grupo diretoria para "setar" no diretório documentos:  
# groupadd diretoria  
Lembrando que é apenas um exemplo! Pode ser qualquer nome de grupo!  
  
Verificando se o mesmo foi criado:  
# cat /etc/group | grep diretoria  
diretoria:x:1001:  
  
Vimos que ele foi criado e que não tem nenhum usuário que pertence a esse grupo.  
Ou seja, a linha dele está vazia:  
diretoria:x:1001: "Não tem nenhum usuário aqui!!!"  
  
Agora, vamos no diretório /mnt/documentos para definir que o grupo desse diretório será diretoria e não mais root:  
# chown root.diretoria documentos/  
  
# ls -ld documentos/  
drwxr-xr-x 2 root diretoria 4096 2004-11-16 15:28 documentos/  
  
Afinal quando um usuário cria um arquivo ou um diretório na máquina, o mesmo vem com seu grupo particular.   
  
No nosso caso, o usuário root que criou o diretório, então, temos que tirar o grupo root e colocar um público para que possamos inserir usuários nesse grupo...  
  
Assim todos os usuários que pertencerem ao grupo diretoria terão acesso a esse diretório.  
  
O dono vai ser o root mesmo, pois como é um diretório público, não tem um único dono!!!  
Então nada melhor que ser o root que ADM o sistema!   
  
Bem, já vimos que o comando chown altera dono e grupo de arquivos e diretórios...  
Então o dono continua sendo o root, pois ele que vai administrar tudo!  
  
Agora nosso próximo passo é criar os usuários que irão acessar esse diretório...  
  
Que no nosso caso será o usuário debian e o gnu!  
O usuário tux será o que NÃO TERÁ ACESSO!!!  
  
Ou seja, o tux vai ser criado mas não faz parte do grupo diretoria!!!  
  
Adicionando o Debian:  
# adduser debian  
  
E o usuário GNU agora:  
# adduser gnu  
  
E o usuário TUX agora:  
# adduser tux  
  
As informações ficam ao critério de vocês. Se quiser pode deixá-las em branco, menos a senha.  
  
Verificando se os mesmos existem no passwd:  
  
# cat /etc/passwd | grep debian  
debian:x:1002:1002:Usuário Exemplo Aula,x,x,x,x:/home/debian:/bin/bash  
  
# cat /etc/passwd | grep gnu  
gnu:x:1003:1003:Usuário Exemplo Aula,x,x,x,x:/home/gnu:/bin/bash  
  
# cat /etc/passwd | grep tux  
tux:x:1006:1006:Usuário Exemplo Aula,x,x,x,x:/home/tux:/bin/bash  
  
Com isso, você já pode abrir os outros 3 terminais, se tornando debian no segundo, gnu no terceiro e tux no quarto...  
  
Bom, eles existem no /etc/passwd então significa que já tenho meus usuários criados!  
  
Mas os mesmos estão com seus respectivos grupos:  
  
# groups debian  
debian : debian  
  
# groups gnu  
gnu : gnu  
  
# groups tux  
tux : tux  
  
Então vamos adicionar esses usuários debian e gnu no nosso grupo diretoria!  
Lembrando que só o debian e o gnu irão fazer parte desse grupo.  
O usuário tux está fora.  
  
Na prática, vamos acrescentar o grupo diretoria para esses usuários:  
  
Adicionando usuário debian ao grupo diretoria  
# gpasswd -a debian diretoria  
  
Adicionando usuário gnu ao grupo diretoria  
# gpasswd -a gnu diretoria  
  
Agora vamos verificar novamente os grupos desses usuários:  
# groups gnu  
gnu : gnu diretoria  
  
Os dois agora fazem parte do grupo diretoria:  
# cat /etc/group | grep diretoria  
diretoria:x:1001:debian,gnu  
  
  
Resumindo:  
Temos um diretório documentos que pertence ao grupo diretoria, e temos 2 usuários que são do grupo diretoria  
  
  
  
**Parte 2 - Fase de Implementação**  
  
A permissão padrão para qualquer diretório criado é de 755  
Então os usuários que pertencem ao grupo diretoria, apenas podem ler esse diretório...  
  
Façam o teste tentando criar um arquivo dentro do diretório documentos com algum usuário (debian ou gnu)  
  
Lembrando que para criar arquivos vazios é o comando abaixo (tentando criar como usuário debian):  
  
$ touch /mnt/documentos/teste.txt   
Não é possível! Permissão negada! triste  
  
Tentando com o usuário gnu:  
$ touch /mnt/documentos/teste.txt  
touch: cannot touch `/mnt/documentos/teste.txt': Permissão negada  
  
Vamos corrigir isso!   
Então como eles têm em comum o grupo diretoria, vamos aplicar as permissões no grupo diretoria...   
Ou seja no grupo que o diretório pertence!  
  
Vamos permitir que os usuários que estão no grupo diretoria possam acessar esse diretório documentos sem problemas dando permissão de gravação (w = 2)  
  
Por padrão no GNU/Linux quando um diretório é criado, ele vem com a permissão 755 ou seja umask 022  
  
Então nesse meu diretório documentos, eu tenho acesso total do DONO que no nosso caso é o root!  
  
E tenho acesso de leitura e poder entrar (execução) para o grupo e o resto.  
  
Como vai ser um diretório público, não quero que ninguém mais que não seja o root e o pessoal do grupo diretoria veja ou entre.  
  
Então a permissão para os outros(resto do sistema que não é do grupo) já vai ser 0, ou seja, sem permissão nenhuma (o-rx)  
  
Já o meu grupo, que por padrão só tem permissão de ler o que tem dentro do diretório (r leitura, x entrar), quero que ele possa gravar lá também!  
  
Então o grupo terá 7 (que equivale a rwx) ou ainda g+w  
  
E o DONO, nosso todo poderoso chefão, irá continuar a ter permissão total!!!  
  
Resumindo, faremos da seguinte maneira:  
  
# chmod 770 documentos/  
# ls -ld documentos/  
drwxrwx--- 2 root diretoria 4096 2004-11-16 15:28 documentos/  
  
Podemos também usar as letras:  
  
# chmod u=rwx,g=rwx,o-rwx documentos/  
# ls -ld documentos/  
drwxrwx--- 2 root diretoria 4096 2004-11-16 15:28 documentos/  
  
Tentem acesso esse diretório público com o usuário tux, vá no terminal do tux.  
$ cd /mnt/documentos/  
bash: cd: /mnt/documentos/: Permissão negada  
  
Pronto!!! Agora tenho os 4 terminais que eu havia falado no início:  
1 - root  
2 - debian  
3 - gnu  
4 - tux  
  
Bom, então, tudo indica que o usuário debian tem permissão de entrar no diretório /mnt/documentos e criar seus arquivos!  
Vamos testar!!!  
  
$ cd /mnt/documentos/  
$ touch arq\_debian.txt  
$ ls  
arq\_debian.txt  
  
  
Com o que fizemos, funciona!  
  
Agora vamos testar isso com o gnu:  
$ cd /mnt/documentos/  
$ touch arq\_gnu  
$ ls  
arq\_debian.txt arq\_gnu  
  
Como já vimos, com o tux sem chance até de entrar no diretório!!  
  
Ele ficou totalmente fora!!!  
  
Agora, parece que está tudo beleza. Os usuários gravando no diretório...   
  
Então eu como root vou dar uma analisada nesse diretório...   
  
Fazer o meu papel de ADM...  
  
Vamos no Terminal 1 como root checar as coisas...  
  
Vou lá no diretório para ver como andam os arquivos...  
  
# cd /mnt/documentos/  
# ls -l  
-rw-r--r-- 1 debian debian 2004-11-16 16:47 arq\_debian.txt  
-rw-r--r-- 1 gnu gnu 2004-11-16 16:57 arq\_gnu  
  
Continuando...   
OPA!!! Peraí!!!! Que bagunça é essa!!!  
  
Cada usuário está criando os arquivos, virando total dono do mesmo, pois até o grupo é o particular dele...  
  
Assim ninguém vai conseguir acessar o de ninguém e meu diretório PÚBLICO está mais para PRIVADO do que outra coisa...  
  
O ideal seria todos os arquivos estarem com o grupo diretoria...  
Pois é o grupo que eles (debian e gnu) tem em comum...  
  
**Parte 3 - Aplicando permissões especiais SGID BIT**  
  
Vamos usar a permissão especial é a sgid bit, que força todos os arquivos de um determinado diretório serem criados com o grupo do diretório e não do usuário que criou!!!  
  
Então no nosso caso, se eu colocar a sgid bit, vou estar falando para o debian e o gnu criarem arquivos com grupo diretoria, pois o diretório documentos tem esse grupo.  
  
Vamos ver como fica essa permissão na prática!  
  
E como root, vamos corrigir nossa falha...  
  
# cd /mnt/  
# ls -ld documentos/  
drwxrwx--- 2 root diretoria 4096 2004-11-16 16:57 documentos/  
  
# chmod g+s documentos/  
# ls -ld documentos/  
drwxrws--- 2 root diretoria 4096 2004-11-16 16:57 documentos/  
  
Percebam que acrescentou o s em grupo, ou seja, sgid bit, e o sistema trocou o x por s na permissão de grupo  
  
Pois estou falando de uma permissão especial para grupo, por isso o nome SGID (GID - Identificação de Grupo!)  
  
Usando o chmod apenas mandei acrescentar na permissão que já exista.. por isso o "+"  
  
Para especificar permissão sgid bit usando modo octal:  
  
Podemos também especificar pelos números!   
  
Por exemplo:  
O Sgid bit é o s na permissão de grupo, que equivale ao número 2   
  
Mas para usar número, tenho que setar sempre todas as permissões, não tenho como acrescentar apenas uma...  
  
Então ficaria:  
#chmod 2770 /mnt/documentos   
  
Sendo que o 2 na frente diz que é Suid Bit, e 770 o que já tinhamos setado antes...  
  
Agora... vamos ver se funcionou!!!  
  
Entrando em ação os terminais de usuários...  
  
Criando arquivo com o debian:  
$ cd /mnt/documentos/  
$ touch arq2\_debian.txt  
  
E criando com o gnu:  
$ cd /mnt/documentos/  
$ touch arq2\_gnu.txt  
  
Como root para ver se meu trabalho está bem feito...  
  
# cd /mnt/documentos/  
# ls -l arq2\*  
-rw-r--r-- 1 debian diretoria 0 2004-11-16 17:16 arq2\_debian.txt  
-rw-r--r-- 1 gnu diretoria 0 2004-11-16 17:16 arq2\_gnu.txt  
  
  
Então, podem como root mesmo apagar o primeiro arquivo que debian e gnu criaram, pois está errado, valendo agora só o segundo!  
  
Vejam que o arq estava errado, mas o arq2 dos dois está certo agora!!!  
  
 **Parte 4 - UMASK**  
  
Se vocês perceberem, os meus arq2 do debian e do gnu já foram criados da maneira certa, ou seja, respeitando os donos, mas deixando o grupo diretoria.  
  
Uma coisa que preciso comentar aqui...  
Quando eu falo em diretório público, estou querendo falar que sempre o meu GRUPO terá que ter permissão de gravação, fugindo do padrão (umask) criado pelo sistema.  
  
Se analisarem os arquivos, eles estão vindo com a permissão...rw-r--r--  
Ou seja, mesmo assim o grupo diretoria não tem acesso para gravar no arquivo...  
  
Já que o UMASK, e vocês podem ver pelos arquivos criados é 644  
Então estou falando que um arquivo criado pelo usuário debian só será editado pelo mesmo. Já que o grupo desse arquivo tem apenas permissão 4 (leitura)  
  
Quero que qualquer usuário do grupo diretoria, crie arquivos nesse diretório deixando permissão de gravação para o grupo também!!!  
  
Então tenho que mudar a umask apenas desses usuários!  
  
Lembrem-se que mexer no valor de UMASK não é uma coisa muito boa quando não aplicamos a um usuário em específico!  
  
Por isso vamos apenas mudar a do debian e do gnu...  
  
O valor de umask padrão do sistema e de todos usuários é 022....  
  
O que equivale a permissão:  
Umask= 0022  
Permissão em Diretório = 755  
Permissão em Arquivo = 644  
  
Então eu tenho que definir a UMASK 007 para os usuários....  
  
Umask=007 (Lembre-se que os outros usuários que não pertencem ao grupo não terão nem permissão de leitura, isto é, o diretório é público só para quem está no grupo diretoria)  
  
Permissão em Diretório = 770  
Permissão em Arquivo = 660  
  
O valor de umask está setado no arquivo .bashrc do home dos usuários...  
#cat /home/debian/.bashrc | grep umask  
  
Agora quem usa Debian terá que arrumar no .bashrc dos usuários que falamos...  
  
# vi /home/debian/.bashrc  
umask 007  
  
No arquivo a linha do umask vem comentada, então retire a cerquilha # antes da linha.  
  
Para que a alteração entre em vigor faça:  
$ source /home/debian/.bashrc  
  
O comando source lê novamente os comandos que estão no arquivo. Lembrem-se que o .bashrc é lido após o login do usuário.  
  
Pensando como Distro Debian, preciso alterar o UMASK e fazer um novo login desses usuários (debian e gnu) ou usar o comando source.  
  
Criando um terceiro arquivo como Gnu:  
  
$ cd /mnt/documentos/  
$ touch arq3\_gnu.txt  
  
Como debian agora:  
  
$ cd /mnt/documentos/  
$ touch arq3\_debian.txt  
  
Verificando com o root:  
# cd /mnt/documentos/  
# ls -ls arq3\*  
  
-rw-rw---- 1 debian diretoria Nov 19 11:16 arq3\_debian.txt  
-rw-rw---- 1 gnu diretoria Nov 19 11:18 arq3\_gnu.txt  
  
  
Agora sim tenho a permissão de gravação para o grupo também!!!  
  
Mais uma vez no Red Hat e derivados esse passo já é feito por padrão na distro!  
  
Para ficar melhor a compreensão, vai lá como root e apaga os arquivos criados com o arq e arq2, deixando apenas o 3 que está certo.  
  
Vamos ficar apenas com os arq3!!!  
  
Agora antes de complicar mais e para garantir as nossas alterações, vamos criar um 4º arquivo para os dois usuários...  
  
Com o gnu:  
$ touch arq4\_gnu.txt  
  
Com o debian:  
$ touch arq4\_debian.txt  
  
Como root:  
# ls -l  
  
-rw-rw---- 1 debian diretoria Nov 19 11:16 arq3\_debian.txt  
-rw-rw---- 1 gnu diretoria Nov 19 11:18 arq3\_gnu.txt  
-rw-rw---- 1 debian diretoria Nov 19 11:19 arq4\_debian.txt  
-rw-rw---- 1 gnu diretoria Nov 19 11:19 arq4\_gnu.txt  
  
**Parte 5 - Aplicando permissões especiais STICK BIT**Parece que agora está tudo certo...  
  
Então, meus usuários estão lá felizes da vida e gravando e compartilhando os arquivos no /mnt/documentos...  
Lembrando que apenas o debian e o gnu!  
  
Então o usuário debian muito amigo...  
  
debian@matrix:documentos$ ls  
arq3\_debian.txt arq3\_gnu.txt arq4\_debian.txt arq4\_gnu.txt  
debian@matrix:documentos$ rm arq3\_gnu.txt  
rm: remove regular empty file `arq3\_gnu.txt'? y  
  
E o Gnu para se vingar...  
gnu@matrix:documentos$ ls  
arq3\_debian.txt arq4\_debian.txt arq4\_gnu.txt  
debian@matrix:documentos$ rm arq3\_debian.txt  
rm: remove regular empty file `arq3\_debian.txt'? y  
  
Usuário apagando arquivo de outro usuário...  
E agora quem poderá me ajudar???  
  
Quem vai me ajudar... é ELA!!!   
A permissão Stick Bit!!!  
  
A permissão stick bit em um diretório faz com que apenas o dono e somente ele possa apagar o arquivo que ele e somente ele criou!!!  
  
Essa permissão já vem aplicada no diretório público /tmp.  
  
Então para esse problema temos outra permissão Especial!!!  
Então, vamos no diretório /mnt/ como root arrumar a casa novamente!!!  
  
Antes tínhamos assim:  
  
# ls -ld /mnt/documentos/  
drwxrws--- 2 root diretoria 4096 Nov 19 11:34 /mnt/documentos/  
  
Agora vamos acrescentar a stick bit que é a letra t:  
  
# chmod o+t /mnt/documentos/  
# ls -ld /mnt/documentos/  
drwxrws--T 2 root diretoria 4096 Nov 19 11:34 /mnt/documentos/  
  
Sendo que o T tem que estar na permissão do resto dos usuários, que não são donos e nem pertencem ao grupo.  
  
Não pode ser na de dono e nem de grupo, pois nenhum outro usuário pode apagar se não for o dono.  
Por isso coloquei o+t (o = other)  
  
Essa permissão tem o valor 1  
Então contando que já temos permissões setadas, tenho que colocá-las tudo novamente quando uso em números:  
  
# chmod 3770 /mnt/documentos  
  
Sendo que 3 = 2 (Sgidbit) + 1 (Stickbit)  
  
E 770 as permissões normais que já tinhamos setado!  
  
Agora vamos imediatamente lá nos terminais dos usuários acabar com a farra deles...  
  
O debian tentando apagar o do gnu:  
  
debian@matrix:documentos$ rm arq4\_gnu.txt  
rm: remove regular empty file `arq4\_gnu.txt'? y  
rm: cannot remove `arq4\_gnu.txt': Operation not permitted  
  
E gnu tentando apagar o do debian:  
  
gnu@matrix:documentos$ rm arq4\_debian.txt  
rm: remove regular empty file `arq4\_debian.txt'? y  
rm: cannot remove `arq4\_debian.txt': Operation not permitted  
  
Ou seja, só o verdadeiro dono (quem criou) pode apagar!!!  
Mesmo que a permissão do arquivo seja total para o grupo!!  
  
**Parte 6 - Aplicando permissões especiais SUID BIT**  
  
Para demonstrar o uso do SUID, que é outra permissão especial, vou dar como exemplo o comando shutdown, que é utilizado para desligar e reiniciar o sistema, mas que só pode ser executado pelo usuário root.   
  
Mesmo se você der permissão através do "chmod 755 /sbin/shutdown", o usuário comum não vai conseguir realizar a execução deste, somente o root.   
  
Exemplo prático:  
  
Vou criar um grupo no qual os usuários que poderão reiniciar ou desligar o sistema estarão:  
# groupadd shutdown  
  
No meu caso, o usuário comum leo estará nesse grupo:  
# gpasswd -a leo shutdown  
  
Precisamos mudar o grupo do arquivo executável /sbin/shutdown:  
# chown root.shutdown /sbin/shutdown  
  
Aplicando a permissão especial SUID em modo octal, apenas o root terá poder de escrita nesse arquivo:  
# chmod 4750 /sbin/shutdown  
  
Agora, vamos criar um link do arquivo para o /bin. Lembrem-se que o usuário comum só executa arquivos que estão em /bin, por isso a necessidade do link. Criando o link:  
  
# ln -s /sbin/shutdown /bin/shutdown