Gerenciamento de permissões do sistema de arquivos a partir da linha de comando

Alteração de permissões de arquivo e diretório

O comando `chmod` tem as seguintes características:

- ele altera as permissões do arquivo e do diretório na linha de comando
- ele pode ser interpretado como "modo de alteração", pois o *modo* de um arquivo é outro nome para permissões de arquivo
- ele assume uma instrução de permissão seguida por uma lista de arquivos ou diretórios a serem alterados
- Você pode definir a instrução de permissão simbolicamente ou em notação octal (numérica)

Alteração de permissões com o método simbólico

Use o comando `chmod` para modificar as permissões de arquivo e diretório.

Quem é a classe do usuário, como na tabela a seguir. Se você não fornecer uma classe de usuário, o comando `chmod` usará o grupo `all` como padrão.

Quem	Conjunto	Descrição
u	user	O proprietario do arquivo
g	group	Membro do grupo do arquivo
o	other	Usuarios que não são proprietários do arquivo nem membros do grupo do arquivo.
a	all	Todos os três grupos anteriores.

O que é o operador que modifica *Qual*, como na tabela a seguir.

O que	Operação	Descrição
+	*add*	Adiciona as permissões ao arquivo.
-	*remove*	Remove as permissões do arquivo.
=	*set exactly*	Define exatamente as permissões fornecidas para o arquivo

Qual é o modo e especifica as permissões para os arquivos ou diretórios, como na tabela a seguir.

Qual	Modo	Descrição
r	read	Acesso de leitura ao arquivo. Listagem de acesso ao diretório.
w	*write*	Gravar permissões no arquivo ou diretório.
x	*execute*	Executar permissões do arquivo. Permite entrar no diretório e acessar arquivos e subdiretórios dentro do diretório.
X	special execute	Executar permissões para um diretório ou executar permissões para um arquivo se pelo menos um dos bits de execução estiver definido.

O método *simbólico* de alterar permissões de arquivo usa letras para representar os grupos de permissões:

- `u` para usuário
- `g` para grupo
- `o` para outro
- `a` para todos

Com o método simbólico, não é necessário definir um novo grupo de permissões. Em vez disso, você pode alterar uma ou mais permissões existentes.

- Use os caracteres de adição `+` ou subtração `-` para adicionar ou remover permissões

- Use o caractere (`=`) para substituir o conjunto inteiro por um grupo de permissões

Uma única letra representa as permissões:

- `r` para leitura
- `w` para gravação
- `x` para execução
- `X` maiúsculo como o sinalizador de permissão para adicionar permissões de execução somente se o arquivo for um diretório ou se executar já estiver definido para usuário, grupo ou outro

A lista a seguir mostra alguns exemplos de alteração de permissões com o método simbólico:

Remova a permissão de leitura e gravação para grupo e outro no arquivo `document.pdf`:

```
[user@host ~]$ chmod go-rw document.pdf
```

Adicione permissão de execução para todos no arquivo `myscript.sh`:

```
[user@host ~]$ chmod a+x myscript.sh
```

Você pode usar a opção `-R` do comando `chmod` para definir de maneira recursiva as permissões nos arquivos em uma árvore de diretórios inteira.
Por exemplo, o próximo comando adiciona recursivamente permissões de leitura, gravação e execução para os membros do grupo que são proprietários do diretório `myfolder` e os arquivos e diretórios dentro dele.

```
[user@host ~]$ chmod -R g+rwx /home/user/myfolder
```

Você também pode usar a opção `-R` do comando `chmod` com a opção `-X` para definir permissões simbolicamente.

Com a opção `X` do comando `chmod`, é possível definir a permissão de execução (pesquisa) em diretórios de modo a possibilitar o acesso a seu conteúdo sem alterações de permissão na maioria dos arquivos.

Entretanto, tenha cuidado com a opção `X`, porque se alguma permissão de execução for definida para um arquivo, a opção `X` também definirá a permissão de execução especificada naquele arquivo.

Por exemplo, o comando a seguir define recursivamente o acesso de leitura e gravação no diretório `demodir` e todos os seus filhos para o proprietário do grupo, mas somente aplica permissões de execução de grupo a diretórios e arquivos que já têm as permissões de execução definidas para usuário, grupo ou outro.

```
[root@host opt]# chmod -R g+rwX demodir
```

Alteração de permissões com o método octal

Você pode usar o comando `chmod` para alterar as permissões do arquivo com o método octal em vez do método simbólico. No exemplo a seguir, o caractere # representa um dígito.

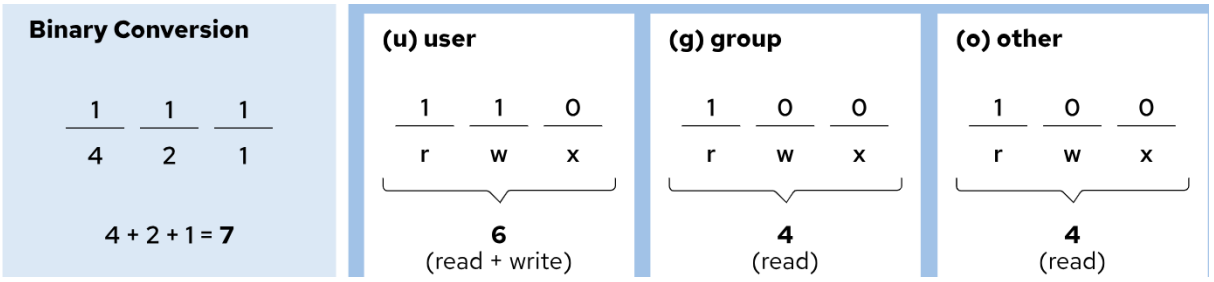
```
chmod ### file|directory
```

Usando o método octal, você pode representar permissões como um número *octal* de três dígitos (ou quatro, ao configurar permissões avançadas). Um único dígito octal pode representar qualquer valor único de 0 a 7.

Na representação octal de três dígitos, cada dígito representa um nível de acesso, da esquerda para a direita: usuário, grupo e outro. Para determinar cada dígito:

- Comece com 0
- Para adicionar permissões de leitura para esse nivel de acesso, adicione 4
- Para adicionar permissões de gravação, adicione 2
- Para adicionar permissões de execução, adicione 1

O diagrama a seguir ilustra como os sistemas interpretam o valor de permissão octal `644`.



Administradores experientes geralmente usam permissões octais para implementar em arquivos únicos ou correspondentes, e fornecem controle de permissão total.

A lista a seguir mostra alguns exemplos de alteração de permissões com o método octal:

Defina permissões de leitura e gravação para o usuário e leitura para o grupo e outros no arquivo `sample.txt`:

```
[user@host ~]$ chmod 644 sample.txt
```

Defina permissões de leitura, gravação e execução para o usuário, leitura e execução para o grupo e sem permissão para outros no diretório `sampledir`:

```
[user@host ~]$ chmod 750 sampledir
```

Alteração de propriedade de grupo ou usuário do diretório e arquivo

Você pode alterar a propriedade do arquivo usando o comando `chown` (alterar proprietário)

Por exemplo, para conceder a propriedade do arquivo `app.conf` ao usuário `student`, use o seguinte comando:

```
root@host ~]# chown student app.conf
```

A opção `-R` do comando `chown` altera recursivamente a propriedade de uma árvore de diretório inteira. O seguinte comando concede a propriedade do diretório `Pictures` e todos os arquivos e subdiretórios contidos nele ao usuário `student`:

```
[root@host ~]# chown -R student Pictures
```

Você pode também usar o comando `chown` para alterar a propriedade do grupo de um arquivo ao preceder o nome do grupo com dois pontos (`:`). Por exemplo, o seguinte comando altera a propriedade de grupo do diretório `Pictures` para `admins`:

```
[root@host ~]# chown :admins Pictures
```

Você pode usar o comando `chown` para alterar o proprietário e o grupo ao mesmo tempo usando a sintaxe *owner*:*group*. Por exemplo, para alterar a propriedade do diretório `Pictures` para o usuário `visitor` e o grupo para `guests`, use o seguinte comando:

```
[root@host ~]# chown visitor:guests Pictures
```

Em vez de usar o comando `chown`, alguns usuários alteram a propriedade do grupo usando o comando `chgrp`. Esse comando funciona de modo semelhante ao `chown`, mas é usado apenas para alterar a propriedade do grupo, e os dois-pontos (`:`) antes do nome do grupo não são necessários.

Você pode encontrar uma sintaxe alternativa `chown` que separa proprietário e grupo com um caractere de ponto em vez de dois-pontos:

```
[root@host ~]# chown owner.group filename
```

Exercício orientado: Gerenciamento de permissões do sistema de arquivos a partir da linha de comando

Gerenciamento de permissões padrão e acesso aos arquivos