Introdução à Ciência da Computação

Linux – parte final

Prof. Iago Augusto de Carvalho

iago.carvalho@unifal-mg.edu.br

1

Roteiro – 1ª parte

•Arquivo fstab;

• Função e localização

• Estrutura

•Comando mount

•Comando umount

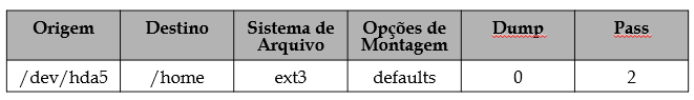
2

Função e Localização

• Arquivo fstab - Localizado em /etc/fstab

Contém as partições que o sistema deverá montar no momento do boot; 3

Estrutura



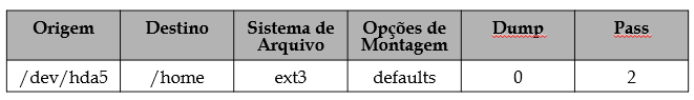
**Exemplo**

Origem: arquivo mapeado no diretório /dev

Destino: diretório em que o dispositivo vai ser montado

Sistema de Arquivo: de acordo com o padrão GNU/Linux, que pode ser ext2, ext3, ext4 ou reiserfs 4

Estrutura



**Exemplo**

Opções de Montagem: auto: se o mesmo será montado no boot;

noauto: para não ser montado;

user: para um usuário normal montar;

nouser: apenas o root pode montar;

rw: para poder gravar nessa partição;

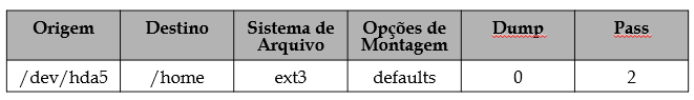
ro: para não gravar;

exec: permite a execução de arquivos na partição;

noexec: é para não permitir que binários (executáveis) sejam executados na partição; async: estabelece E/S assíncrona para o sistema de arquivos montado, é o oposto de sync;

5

Estrutura



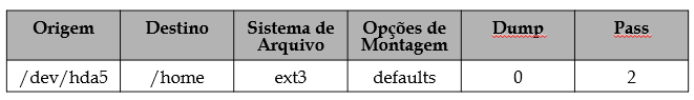
**Exemplo**

Opções de Montagem:

dev: interpreta dispositivos especiais de caracteres ou blocos no sistema de arquivos defaults: utiliza as seguintes opções: nouser, exec, auto, rw, dev

6

Estrutura

**Exemplo**

• Dump:

0 para não realizar dump;

1 para realizar;

O dump é um aplicativo de backup em nível de partição;

Com o dump ativado, um backup será feito antes da partição ser montada. Opção já se encontra desatualizada.

• Pass:

se o valor for 0, a unidade não será checada;

1 para checar a partição / ;

2 para as demais partições que quero checar;

(comando fsck será executado)

7

Comando mount

•Comando mount é usado para se montar dispositivos no sistema; •Exemplo, caso um pendrive não seja montado automaticamente, o mesmo deverá ser montado de forma explícita, usando-se o comando mount;

•Sintaxe:

• mount [opções] dispositivo diretório de montagem

• Para checar quais partições estão montadas:

• df -h

8

Comando mount

•Algumas opções bastante usadas:

• *-a* monta todas as partições especificadas no fstab, com exceção das que possuem a opção *noauto*;

• *-r* monta como somente leitura;

• *-w* monta como leitura e escrita;

• *-t tipo* especifica o sistema de arquivos a ser montado;

9

Comando umount

•Comando umount é usado para se desmontar dispositivos no sistema; •Sintaxe:

• umount [opções] dispositivo

• ou

• umount [opções] diretório

10

Roteiro – 2ª parte

• Comandos:

• Comandos de auxílio em comandos; • Comandos para se localizar arquivos; • Editor vim

11

Comandos

**whereis** - busca por arquivos executáveis, man pages, arquivos de configuração e fontes. **which** - busca por executáveis nos PATHs exportados.

Exemplos:

whereis ls

which ls

**apropos** – Trabalha com um banco de dados, buscando pela descrição do comando. Muito útil quando se esquece a sintaxe do comando ou a funcionalidade;

Exemplo:

apropos compiler

**whatis** – retorna a descrição do comando;

Exemplos:

whatis ls

whatis touch

12

Comandos

•Comandos de busca:

• find

• Exemplo:

• find / -name passwd

• locate – trabalha com uma base de dados indexada;

• updatedb realiza a indexação dos arquivos e cria a base de dados; •Somente o root pode realizar e atualizar a indexação dos arquivos;

13

Editor VIM

• Modo de comando, usado para dar ordens

ao editor;

O vim possui

• Modo de linha de comando, permite realizar

3 modos de

ações como salvar, sair do vim, além de

poder executar comandos do shell sem sair

operação:

do editor (sempre começa com : );

• Modo de inserção, usado para inserir texto;

Observação: sempre ao iniciar o vim ele estará no modo de comando;

14

Algumas opções na abertura de arquivos

$ vim → Abre o vim “vazio” e exibe a tela de apresentação;

$ vim arquivo → Abre o arquivo de nome “arquivo”;

$ vim arquivo + → Abre o arquivo de nome “arquivo”, com o cursor no final do mesmo;

$ vim arquivo +10 → Abre o arquivo de nome “arquivo”, com o cursor na linha 10; $ vim arquivo +/palavra → Abre o arquivo de nome “arquivo”, na primeira ocorrência da palavra “palavra”;

15

Algumas opções em linha de comando

:w → Salva o arquivo que está sendo editado no momento;

:q → Sai;

:wq → Salva e sai;

:x → Salva e sai;

ZZ → Salva e sai;

:w! → Salva forçado;

:q! → Sai forçado;

:wq! → Salva e sai forçado;

W → Move o cursor para o início da próxima palavra (ignorando a pontuação); b → Move o cursor para o início da palavra anterior (não ignorando a pontuação); B → Move o cursor para o início da palavra anterior (ignorando a pontuação);

16

Algumas opções em comandos

A → Insere o texto no fim da linha onde se encontra o cursor;

o → Adiciona uma linha vazia abaixo da linha corrente;

O → Adiciona uma linha vazia acima da linha corrente;

Ctrl + h → Apaga o último caractere à esquerda;

Ctrl + f → Passa para a tela seguinte;

Ctrl + b → Passa para a tela anterior;

H → Move o cursor para a primeira linha da tela;

M → Move o cursor para o meio da tela;

L → Move o cursor para a última linha da tela;

h → Move o cursor para caractere a esquerda;

j → Move o cursor para linha abaixo;

k → Move o cursor para linha acima;

l → Move o cursor para caracter a direita;

w → Move o cursor para o início da próxima palavra (não ignorando a pontuação); 17

Algumas opções em comandos

0 (zero) → Move o cursor para o início da linha corrente;

^ → Move o cursor para o primeiro caractere não branco da linha;

$ → Move o cursor para o fim da linha corrente.

nG → Move o cursor para a linha de número “n”;

u → desfaz uma ação;

CTRL + r → refaz uma alteração;

18

Comandos para a localização de textos /palavra → Procura pela palavra ou caractere acima ou abaixo do texto;

?palavra → Move para a ocorrência anterior da palavra (para repetir a busca use "n"); 19

Comandos para a remoção de

caracteres

x → Apaga o caractere onde o cursor estiver;

dd → Apaga a linha inteira onde o cursor estiver;

D → Apaga a linha a partir da posição do cursor até o fim;

J → Une a linha corrente à próxima;

5dd → Remove as próximas 5 linhas a partir da posição do atual do cursor (pode ser usado com qualquer valor numérico);

20

Copiar e colar

yy → Copia a linha onde o cursor se encontra;

5yy → Copia as próximas 5 linhas a partir da posição atual do cursor;

p → Cola o que foi copiado na linha abaixo do cursor atual;

21

Roteiro – 3ª parte

• Comando CHMOD

22

O comando CHMOD

Nos sistemas operacionais do tipo Unix, o comando **chmod** define as permissões de arquivos ou diretórios.

Um conjunto de sinalizadores associados a cada arquivo determina quem pode acessar esse arquivo e como eles podem acessá lo (*permissões)*

O nome do comando **chmod** significa “alterar modo”. Ele restringe a maneira como um arquivo pode ser acessado.

23

O comando CHMOD

•Em geral, os comandos **chmod** assumem a forma:

◦ chmod (opções) *permissões nome do arquivo*

•*permissões”* define as permissões para o proprietário do arquivo (o “usuário”), membros do grupo que possui o arquivo (o “grupo”) e qualquer outra pessoa (“outros”).

•Há duas maneiras de representar essas permissões: com símbolos (caracteres alfanuméricos) ou com números octais (os dígitos de **0** a **7**).

24

O comando CHMOD

•Digamos que você seja o proprietário de um arquivo chamado **myfile** e deseje definir suas permissões para que:

•o usuário pode ler, escrever e executar;

•os membros do seu grupo podem lê-lo e executá-lo; e

•outros podem apenas lê-lo.

25

O comando CHMOD

Esse comando fará o truque:

chmod u=rwx, g=rx, o=r myfile

Este exemplo usa notação de permissões simbólicas.

As letras **u** , **g** e **o** representam ”**usuário**“, ”**grupo**” e ”**outro**“. O sinal de igual (”**=**“) significa “definir as permissões exatamente assim” e as letras ”**r**“, ”**w**” e ”**x**” representam “leitura”, “gravação” e “execução”, respectivamente.

As vírgulas separam as diferentes classes de permissões e não há espaços entre elas.

chmod 754 myfile Comando equivalente em octal

26

O comando CHMOD

chmod 754 myfile

•Aqui, os dígitos **7** , **5** e **4** representam individualmente as permissões para o usuário, grupo e outros, nessa ordem.

•Cada dígito é uma combinação dos números **4** , **2** , **1** e **0** :

**4** significa “ler”,

**2** significa “gravação”,

**1** significa “executar” e

**0** significa “sem permissão”.

•Portanto:

**7** é a combinação de permissões **4** + **2** + **1** (leitura, gravação e execução), **5** é **4** + **0** + **1** (leitura, não gravação e execução) e

**4** é **4** + **0** + **0** (leitura, não escrever e não executar).

27

O comando CHMOD - opções

| -c , –changes | Como –verbose , mas fornece saída detalhada apenas quando uma alteração é realmente feita. |
| --- | --- |
| -f , –silent , –quiet | Modo silencioso; suprima a maioria das mensagens de erro. |
| -v , –verbose | Modo verboso; enviar uma mensagem de diagnóstico para cada arquivo processado. |
| –no-preserve-root | Não trate ‘ / ‘ (o diretório raiz ) de nenhuma maneira especial, que é a configuração padrão. |
| –preserve-root | Não opere recursivamente em ‘ / ‘. |
| –reference = RFILE | Defina permissões para corresponder às do arquivo *RFILE* , ignorando qualquer *MODO* especificado. |
| -R , –recursivo | Altere arquivos e diretórios recursivamente. |
| –versão | Informações de versão de saída. |

28

Exibindo permissões dos arquivos

ls -l file.txt

-rwxrw-r-- 1 hope hopestaff 123 fev 03 15:36 file.txt

| – | O primeiro caractere representa o tipo de arquivo: ” – ” para um arquivo regular, ” d ” para um diretório, ” l ” para um link simbólico. |
| --- | --- |
| rwx | Os próximos três caracteres representam as permissões para o proprietário do arquivo: nesse caso, o proprietário pode ler, escrever ou executar o arquivo. |
| rw | Os próximos três caracteres representam as permissões para os membros do grupo de arquivos. Nesse caso, qualquer membro do grupo proprietário do arquivo pode ler ou escrever no arquivo. O traço final é um espaço reservado; os membros do grupo não têm permissão para executar este arquivo. |

29

Exibindo permissões dos arquivos

-rwxrw-r-- 1 hope hopestaff 123 fev 03 15:36 file.txt

| r– | As permissões para “outros” (todos os outros). Outros podem apenas ler este arquivo. |
| --- | --- |
| 1 | O número de links físicos para este arquivo. |
| hope | O proprietário do arquivo. |
| hopestaff | O grupo ao qual o arquivo pertence. |
| 123 | O tamanho do arquivo em blocos . |
| 03 de fev 15:36 | O mtime do arquivo (data e hora em que o arquivo foi modificado pela última vez). |
| file.txt | O nome do arquivo. |

30

Referências

Slides adaptados do Prof. Flavio Barbieri Gonzaga e do curso Introdução ao SO Linux – Renata Spolon e Gabriel Martinez

PRITCHARD, S.; PESSANHA, B. G.; LANGFELDT, N.; STANGER, J.; DEAN, J. 2007. **Certificação Linux LPI Rápido e Prático. Guia de Referência nível 1: Exames 101 e 102**. 2ª Ed. Rio de Janeiro: Editora Alta Books.

31