

Sistema de arquivos

Os sistemas de arquivos do UNIX possuem as seguintes características fundamentais:

- Estruturados na forma de uma árvore única, iniciando pelo diretório "/", que é chamado de "raiz".
- Há suporte para arquivos, diretórios e links (atalhos).
- Os arquivos podem ter qualquer nome, usando quaisquer caracteres, com distinção entre maiúsculas e minúsculas. Os nomes são normalmente limitados a 255 caracteres.
- O caractere separador de diretórios é o "/" (barra).
- Arquivos e diretórios cujos nomes começam com "." (ponto) são considerados "ocultos" e normalmente não aparecem nas listagens de diretórios.
- As extensões são normalmente usadas apenas para facilitar a vida do usuário, mas não são importantes para o sistema operacional, que não depende delas para identificar o conteúdo de um arquivo.
- Os arquivos e diretórios possuem permissões de acesso controláveis por seus proprietários.

Os principais sistemas de arquivos usados para a formatação de discos locais em Linux são o ext2, ext3, reiser, xfs e jfs, entre outros. Os sistemas mais recentes implementam o conceito de *journaling*. Os links a seguir fornecem mais detalhes sobre esse conceito e os sistemas de arquivos mais usados:

- [SourceForge - Ext2/Ext3](#)
- <http://oss.sgi.com/projects/xfs/SiG> - XFS on Linux
- [Informações sobre outros filesystems](#)

Hierarquia de diretórios

Os diretórios de um sistema de arquivos no UNIX têm uma estrutura pré-definida, com poucas variações. Essa estrutura normalmente segue a padronização sugerida pelo documento [Filesystem Hierarchy Standard \(resumo do RedHat 9\)](#).

A seguir ilustramos os principais diretórios de um sistema Linux típico:

- /home : raiz dos diretórios home dos usuários.
- /boot : arquivos de boot (núcleo do sistema, etc)
- /var : arquivos variáveis, áreas de spool (impressão, e-mail, news), arquivos de log
- /etc : arquivos de configuração dos serviços
- /usr : aplicações voltadas aos usuários
- /tmp : arquivos temporários
- /mnt : montagem de diretórios compartilhados temporários
- /bin : aplicações de base para o sistema
- /dev : arquivos de acesso aos dispositivos físicos e conexões de rede
- /lib : bibliotecas básicas do sistema
- /proc : não é um diretório real em disco, mas a porta de acesso para estruturas do núcleo

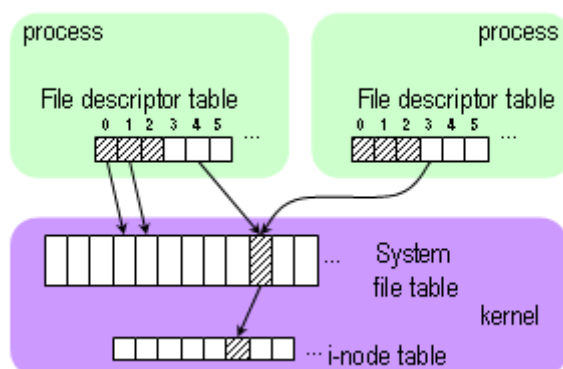
Descritores e Streams

Quando um processo abre um arquivo, o núcleo do sistema operacional precisa criar várias estruturas de dados para gerenciar seu uso. Duas estruturas de dados são importantes nesse contexto: a tabela de descritores de arquivos (*file descriptor table*) e a tabela de arquivos do sistema (*system file table*).

- Cada processo possui sua própria **tabela de descritores de arquivos**. Os descritores de arquivos

usados pelo processo nas operações sobre os arquivos são índices ou ponteiros para entradas nessa tabela local.

- O sistema operacional possui uma **tabela de arquivos do sistema**, que possui uma entrada para cada open ativo. Essa tabela possui várias entradas, sendo uma delas um apontador (ou índice) para a tabela de *i-nodes* (ou *v-nodes*), que possui uma entrada para cada arquivo aberto no sistema.



Um descritor de arquivo constitui uma interface de baixo nível para o acesso ao mesmo. Por ser simplesmente um índice em uma tabela local, geralmente é representado por uma variável do tipo `int`.

O mecanismo de *streams* provê uma interface mais abstrata para acesso aos arquivos, construída a partir dos descritores. Essa abstração provê uma maior homogeneidade no acesso aos diversos tipos de arquivos e travamento (*locking*) automático, além de um controle mais fino sobre os mecanismos de buferização do arquivo em memória. Streams são definidos por variáveis do tipo `FILE *`.

Deve-se observar que, enquanto descritores de arquivos são herdados por processos filhos, *streams* não o são.

Arquivos padrão

Cada processo sempre possui três descritores de arquivos pré-definidos, os chamados *arquivos padrão*, geralmente definidos no arquivo `unistd.h`:

- `STDIN_FILENO` (*stream* `stdin`, entrada 0) : entrada padrão (default: teclado). Usado por todas as funções de entrada de dados que não especificarem um descritor de arquivo.
- `STDOUT_FILENO` (*stream* `stdout`, entrada 1) : saída padrão (default: terminal). Usado por todas as funções de saída de dados que não especificarem um descritor de arquivo.
- `STDERR_FILENO` (*stream* `stderr`, entrada 2) : saída de erro (default: terminal). Usado pelas funções que produzem mensagens de erro.

Os arquivos padrão são geralmente associados ao terminal onde o processo foi lançado, mas podem ser redirecionados para outros arquivos através do shell (operadores `>`, `<`, `>>`, `|`, etc) ou dentro do próprio processo, através das funções de abertura de streams (`fopen`, `freopen`).

Operações básicas em arquivos

O núcleo do sistema operacional UNIX disponibiliza as seguintes chamadas de sistema (*syscalls*) para as operações básicas de entrada/saída em arquivos, que operam sobre descritores: `open`, `close`, `read`, `write` e `fcntl`. As demais operações são normalmente implementadas como funções de biblioteca que fazem uso dessas chamadas.

- Operações usando descritores
- Operações usando streams
- Operações em meta-dados
- Operações em diretórios

Atividades

Execute o programa a seguir e explique o que ocorre com sua saída:

```
#include <stdio.h>

main ()
{
    fprintf (stdout, "a ") ;
    fprintf (stderr, "imprimi a ") ;
    fprintf (stdout, "b ") ;
    fprintf (stderr, "imprimi b ") ;
    fprintf (stdout, "\n") ;
    return 0 ;
}
```

Escreva um programa mycp que efetua a cópia de um arquivo em outro:

```
mycp arq1 arq2
```

Antes da cópia, arq1 deve existir e arq2 não deve existir. Mensagens de erro devem ser geradas caso essas condições não sejam atendidas ou o nome dado a arq2 seja inválido.

Escreva um programa em C que gere uma listagem do diretório corrente no seguinte formato:

tamanho	nome e tipo
115234	arquivo
4096	diretorio/
21	link->
1024	socket@
1024	pipe=

Modifique o programa anterior para incluir na listagem as permissões das entradas do diretório:

perms	tamanho	nome e tipo
rw-r--r--	115234	arquivo
rwX-----	4096	diretorio/
rxwxrwx	21	link->
rw-rw-rw-	1024	socket@
rw-rw----	1024	pipe=

Idem, para a data de última modificação de cada entrada.

From:

<https://wiki.inf.ufpr.br/maziero/> - **Prof. Carlos Maziero**

Permanent link:

https://wiki.inf.ufpr.br/maziero/doku.php?id=pua:sistema_de_arquivos

Last update: **2008/09/21 12:36**

