Guia Linux

ARQUIVO

Descrição

É o componente básico de qualquer sistema **Unix-like**. Nesses sistemas, quase todas as coisas são tratadas como arquivos e possuem localização dentro do sistema de arquivos.

Tipos de arquivos

No Linux, um arquivo pode ser comum ou especial.

- **arquivo comum** – armazena informações e pode ser de texto, de áudio, de imagem, binários, etc.
- **diretório** – contém outros arquivos e/ou outros diretórios.
- **dispositivo** – representa um dispositivo físico (hardware) do sistema. Existem dois tipos:
  - **de caracteres** – as operações de E/S são feitas de forma sequencial.
  - **de blocos** – as operações de E/S são feitas usando blocos de caracteres.
- **link simbólico** – aponta para outro arquivo.
- **pipe** – utilizado para comunicação entre processos.
- **socket** – utilizado para comunicação entre processos.

No sistema, é possível identificar o tipo do arquivo usando o comando **ls** ou o comando **file.** Por exemplo, se queremos informações sobre o arquivo *teste*, basta digitar

```
ls -l teste
```

Suponha que a saída do comando é mostrada abaixo.

```
-rw-rw-r-- 1 aluno aluno 7743 Jan 20 16:52 teste
```

O campo -rw-rw-r-- indica o tipo e as **permissões** do arquivo, onde o primeiro caractere "-" informa que *teste* é um arquivo comum e os outros caracteres informam que o dono do arquivo (*aluno*)

pode ler e escrever no arquivo, o grupo do arquivo (*aluno*) também pode ler e escrever no arquivo, mas os outros usuários só podem ler.

Também podemos usar

```
file teste
```

para verificar o tipo do arquivo. A resposta abaixo informa que *teste* é um arquivo de texto que usa o conjunto de caracteres **Unicode** com a codificação **UTF-8**.

```
teste: UTF-8 Unicode text
```

A tabela abaixo mostra como os tipos de arquivos são referenciados dentro do Linux.

Tipo	Representação
arquivo comum	–
diretório	d
dispositivo de caracteres	c
dispositivo de blocos	b
link simbólico	l
pipe	p
socket	s

Operações de E/S

Existem dois mecanismos básicos para conectar um **processo** a um arquivo:

- **file descriptor** ou **fd** (descritor de arquivo) – fornece uma interface de baixo nível para as operações de E/S. É representado por um objeto do tipo “int”.
- **Stream** (fluxo) – fornece uma interface de alto nível para as operações de E/S. É representado por um objeto do tipo “FILE *”.

A interface do *fd* permite apenas funções simples para transferência de blocos de caracteres. O uso de *fd* é importante quando há a necessidade de operações de E/S em modos especiais como, por exemplo, *polled* (comunicação síncrona).

A interface do *stream* permite funções mais complexas com operações de E/S formatadas. O uso de *streams* é importante quando se deseja garantir a portabilidade para outros tipos de sistemas

(especialmente para os sistemas não-GNU).

Na realidade, a interface do *stream* é construída a partir dos descritores. Assim, os descritores de arquivo são usados para referenciar todo tipo de arquivo aberto no sistema (mesmo que isto seja ignorado pelo usuário). Isto ocorre porque apenas o **kernel** pode executar operações de E/S e ele só atende pedidos feitos através das **chamadas de sistema** (interface de baixo nível).

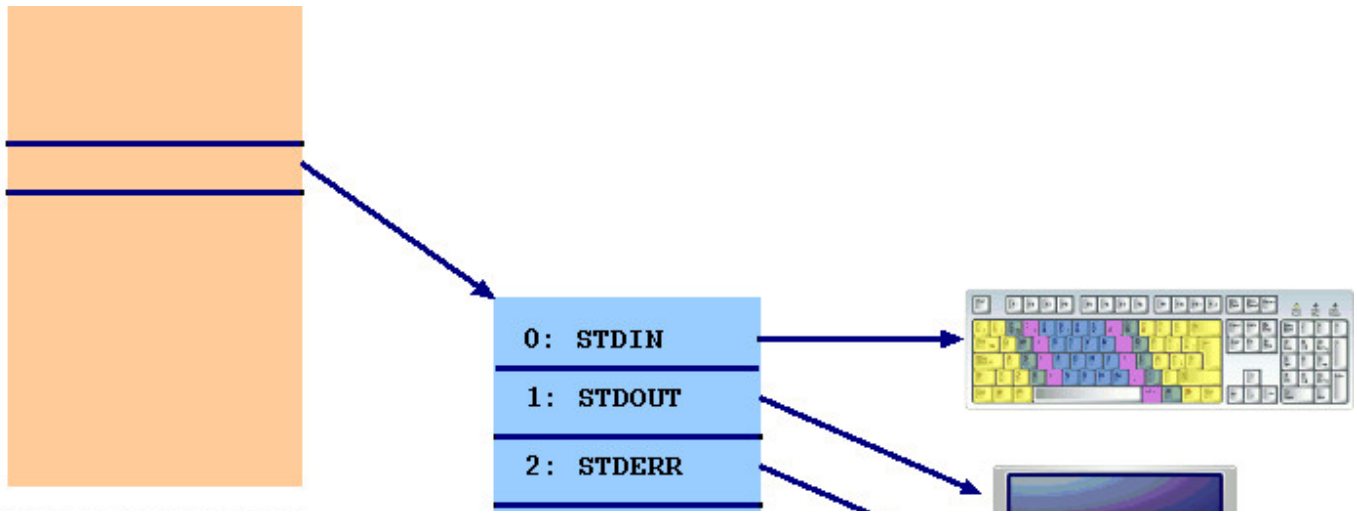
Descritor de arquivo

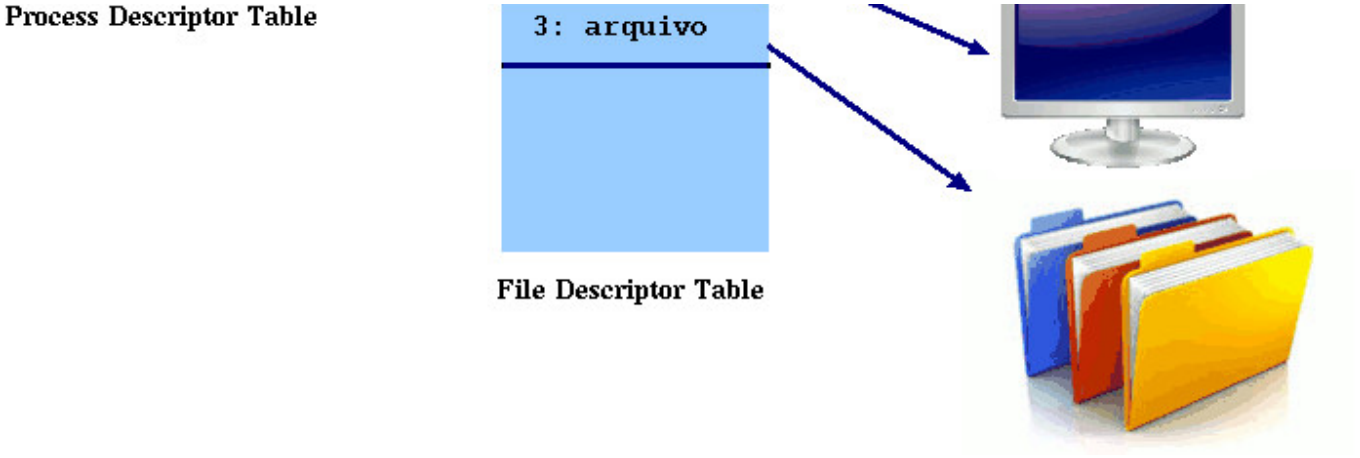
No Linux, o **shell** normalmente opera com três descritores de arquivo sempre abertos: um arquivo que serve de entrada padrão dos dados; um arquivo utilizado como saída padrão dos dados; e um arquivo onde as mensagens de erros são gravadas. Estes descritores são definidos em */usr/include /unistd.h* conforme a tabela abaixo.

FD	Nome	Descrição
0	stdin (*standard input*)	Entrada padrão (normalmente, o teclado)
1	stdout (*standard output*)	Saída padrão (normalmente, o terminal)
2	stderr (*standard error*)	Erro padrão (normalmente, o terminal)

Quando um **processo** é criado a partir do *shell*, ele herda cópias desses descritores. Se **redicionadores de E/S** são especificados na linha de comandos, o *shell* altera a definição dos descritores antes de inicializar o processo.

Cada processo possui uma tabela (*process descriptor table*) com informações necessárias a sua execução. Esta tabela aponta para uma outra tabela (*file descriptor table*) que contém ponteiros para todos os arquivos abertos do processo. Esta tabela já é criada com as três primeiras células apontando para os arquivos padrão: *stdin*, *stdout* e *stderr*. Quando algum outro arquivo é aberto, uma nova entrada na tabela é criada no primeiro slot disponível.





Exemplos

No Linux, o conjunto dos descritores abertos de um processo pode ser visto em /proc/PID/fd/ o de **PID** é o identificador do processo. Por exemplo, para ver o PID do **shell** sendo usado em um terminal, digite

```
ps
```

Suponha que a resposta é mostrada abaixo. Portanto, o **PID** do *shell bash* é 4212.

PID	TTY	TIME CMD
4212	pts/0	00:00:00 bash
4140	pts/0	00:00:00 ps

Para ver os arquivos abertos para o *bash*, basta digitar

```
ls -l /proc/4212/fd
```

A saída abaixo mostra que existem 4 arquivos abertos: 0 (*stdin*), 1 (*stdout*), 2 (*stderr*) e 255 (cópia dos fds para uso interno do *bash* caso os fds padrões sejam redirecionados). Note que os arquivos mostrados são links para o arquivo */dev/pts/11*.

```
total 0
lrwx------ 1 aluno aluno 64 Abr 25 13:28 0 -> /dev/pts/11
lrwx------ 1 aluno aluno 64 Abr 25 13:28 1 -> /dev/pts/11
lrwx------ 1 aluno aluno 64 Abr 25 13:28 2 -> /dev/pts/11
lrwx------ 1 aluno aluno 64 Abr 25 14:04 255 -> /dev/pts/11
```

Entradas em */dev/pts* são na realidade pseudo-terminais: aplicações usam esses terminais para re-

ceber e mostrar dados. Para verificar isto, digite

```
ls -l /dev/pts/11
```

Abaixo, é mostrada uma possível saída.

```
crw--w---- 1 aluno tty 136, 11 Abr 25 14:34 /dev/pts/11
```

Limites

- Para saber o número máximo de fds suportados no sistema, digite

```
cat /proc/sys/fs/file-max
```

Abaixo é mostrada uma possível saída.

```
795346
```

- Para verificar o uso atual dos arquivos no sistema, basta digitar

```
cat /proc/sys/fs/file-nr
```

A resposta abaixo mostra que 11.264 fds já foram alocados (inclui fds já liberados), zero fds alocados e ainda não usados e no máximo 795.346 arquivos podem ser abertos no sistema ao mesmo tempo.

```
11264 0 795346
```

- Para ver o número máximo de fds que um processo pode alocar, use o comando **ulimit**

```
ulimit -Sn
```

Abaixo a resposta do **kernel** 4.15.

```
1024
```

- Para ver o número máximo de fds que o administrador do sistema **(root)** pode autorizar para os processos

```
ulimit -Hn
```

A possível resposta é

4096

Observações

- O comando **fuser** identifica os processos que estão usando um determinado arquivo.
- O comando **lsof** lista os arquivos abertos.
- O comando **stat** fornece informações sobre arquivos.

Sumário | Topo