Módulos no Linux

Introdução

Em nossa máquina existem vários tipos de dispositivos, onde cada um tem a sua função. Temos a nossa placa de vídeo, de rede, de som etc...

Todos esses dispositivos fazem parte do hardware existente nela; todo hardware sempre se comunica diretamente com o kernel, e depois com os programas que vão utilizá-lo, como por exemplo o player de mp3 que precisa do dispositivo de som para executá-lo corretamente.

Como dito antes, o kernel se comunica com os dispositivos de hardware para poder fazê-los funcionar. Só que o hardware tem que interpretar corretamente as instruções fornecidas pelo Sistema Operacional, e vice-versa, pois o S.O. não sabe exatamente como o hardware atua.

Para essa tarefa de "traduzir" as funções de um para o outro é que existem os módulos (**drivers**), que sabem exatamente como aquele dispositivo de hardware deve se comportar no sistema.

Introdução

O papel do módulo é se comunicar com o dispositivo de hardware através de todas as requisições enviadas via SO, fazendo com que o kernel consiga gerenciá-lo corretamente.

Tipos de módulos

Os módulos têm que estar sempre em alerta no momento em que o kernel passa uma ordem vinda do dispositivo para eles. Só que entre os módulos existem algumas diferenças: existem aqueles que ficam sempre disponíveis desde o momento do boot do sistema. Por essa razão, esses módulos que estão intrínsecos no kernel são denominados **built-in.**

Para verificar os módulos que são built-in, temos que analisar tudo o que o kernel carrega na inicialização, com o comando dmesg:

```
# dmesg
```

Tipos de módulos

Um outro grupo de módulos que existe no sistema são aqueles que só são carregados para a memória a partir do momento em que são acionados dentro do sistema, para dar suporte ao dispositivo de hardware.

Esses módulos, que não são built-in, ficam localizados no diretório **/lib/modules/<versão do kernel>/kernel/drivers**.

Ou seja, para cada versão de kernel, eles (módulos) ficam localizados em um caminho diferente dentro do sistema. Para saber o caminho correto, basta verificar qual a versão de kernel utilizada:

```
# uname -r
```

Reconhecendo os Dispositivos

Dentro do sistema, podemos listar informações sobre todos aqueles dispositivos de hardware que foram reconhecidos pelo kernel para sabermos qual módulo (driver) depois ele irá necessitar para funcionar.

Com o comando **lspci**, ele mostrará essa listagem completa:

```
# lspci
0000:00:1f.5 Multimedia audio : Intel Corp. 82801EB/ER AC'97 Audio
0000:03:08.0 Ethernet controller: 3Com 905-TX Ethernet 10/100 Ethernet
0000:00:02.0 VGA compatible controller: SiS 630/730 PCI (rev 02)
```

Reconhecendo os Módulos

Agora que já temos no kernel as informações sobre os dispositivos, podemos checar se existem módulos no sistema que possam atender aos requisitos do dispositivo. Então o kernel pode consultar os módulos para saber detalhes deles.

Por exemplo, o meu dispositivo de rede visto no slide anterior era uma placa **3COM 905-TX** , preciso saber no diretório dos módulos se há algum módulo que pode atender aos requisitos deste hardware com o comando **modinfo**, que trará diversas informações sobre o módulo:

 # modinfo 3c59x

Listando os Módulos Ativos

Vimos que aquele módulo, o **3c59x**, atende aos requisitos para a nossa placa!

Então o kernel sabe que ele corresponde as expectativas... Então é preciso fazer a ativação do dispositivo para ocorrer o suporte ao hardware correspondente:

 # modprobe 3c59x

Para saber se o módulo está ativo:

 # lsmod | grep 3c59x

3c59x 30080 0

mii 4864 1 3c59x

Listando os Módulos Ativos

Quando os módulos ficam ativos, às vezes eles também podem depender de um outro módulo para ajudá-lo, essa é a chamada dependência do módulo.

O **modprobe** já resolve isso, quando ele chama o módulo do dispositivo, chamando também a ajuda dele. Porém, eu posso fazer isso individualmente, com o **insmod**.

No nosso exemplo, na resposta do **lsmod**, você viu que além do **3c59x**, ele sobe outro módulo, o **mii**:

```
# insmod mii
# insmod 3c59x
```

Desativando os Módulos

Depois que o módulo atendeu as solicitações do kernel, podemos fazer com que ele seja desativado da memória, removendo o suporte ao dispositivo:

 # modprobe -r 3c59x

Se você quisesse, também poderia desativar só o módulo, mas a dependência não (ou seja, aquele módulo **mii** vai continuar ativo na memória):

 # rmmod 3c59x

Bibliografia

Linux – Guia do Administrador do Sistema
Autor: Rubem E. Pereira
Editora: Novatec

Manual Completo do Linux (Guia do Administrador)
Autor: Evi Nemeth, Garth Snyder, Trent R. Hein
Editora: Pearson Books

Guia Foca GNU/Linux
http://focalinux.cipsga.org.br/