34.110-0 – Laboratório de Sistemas Operacionais

Turma A

2010/2

Projeto 3: Módulos e Estruturas Internas do Núcleo

1 Objetivo

Aprender como expandir a funcionalidade do núcleo com a criação de módulos personalizados. Explorar e alterar as estruturas internas que guardam informações sobre processos.

2 Tarefas

1. Compilar o módulo fornecido como exemplo. Este módulo cria o arquivo `/proc/hello`, que ao ser lido sempre contém "Hello, World!". Carregar o módulo na memória e verificar o seu funcionamento.

2. Modificar o módulo fornecido para exibir, no lugar da frase fixa, o PID do processo lendo o arquivo e o PID do seu processo pai. Compilar, carregar e verificar o seu funcionamento.

3. Modificar o módulo fornecido para dar ao interpretador de comandos executando o processo de leitura permissões de root. Ou seja, executar "`cat /proc/hello`" deve elevar o privilégio do *interpretador de comandos* para root. Compilar, carregar e verificar o seu funcionamento.

3 Entregas

Você deve entregar:

- O código final de seu módulo modificado.

- Um relatório descrevendo como você implementou as funcionalidades pedidas. O seu relatório deve listar as estruturas de dados usadas pelo seu módulo, descrever brevemente a sua função e descrever como e porque elas foram acessadas. Procure descrever os mecanismos gerais do núcleo e como eles são controlados pelo seu código. Descreva também as dificuldades encontradas. Organize o seu relatório de forma clara.

Os projetos são individuais. As entregas devem ser feitas via Moodle, em um único arquivo .zip ou .tar.gz. O relatório deve estar em formato PDF.

4 Dicas

O módulo fornecido contém um arquivo `Makefile` que permite que o mesmo seja compilado "fora" dos fontes do núcleo. Porém, é necessário ter uma cópia do código fonte do núcleo devidamente configurada e compilada a disposição. Para utilizar este `Makefile`, execute o seguinte comando:

```
$ make -C <diretório-fonte-núcleo> SUBDIRS=$PWD modules
```

Lembre-se: só é possível utilizar o módulo construído desta forma quando o núcleo contra o qual ele for compilado estiver rodando. Para carregar este módulo "externo" use os comandos:

- `lsmod`: Para listar os módulos carregados.
- `insmod ./hello.ko`: Para carregar o arquivo de módulo.
- `rmmod ./hello.ko`: Para descarregar o módulo.

O módulo exemplo fornecido criará o arquivo `/proc/hello` que se for lido exibirá a frase "Hello, world!". Você pode criar um programa para ler este arquivo, mas o comando `cat` é bem mais prático:

```
$ cat /dev/hello
```

O resultado da chamada `printk()` é enviado para o arquivo de registro do sistema. No Fedora este arquivo pode ser encontrado em `/var/log/messages`, que só pode ser acessado por root. Alternativamente, use o comando `dmesg` para exibir as mensagens mais recentes.

A estrutura de dados crucial para o Linux e para este projeto é `task_struct`, definida no arquivo `linux-x.y.z/include/linux/sched.h`. Esta estrutura guarda as informações mantidas pelo Linux sobre um processo. Para acessar o `task_struct` do processo atual use a macro `current`, que se comporta como uma variável do tipo `struct task_struct *`. Logo para acessar o estado de execução de um processo, podemos fazer `current->state`. Uma descrição da lista de processo e algumas outras formas de navegá-la podem ser encontradas em `http://linuxgazette.net/133/saha.html`.

Dentro de `task_struct` existe uma outra estrutura muito importante para este projeto chamada `cred`, definida no arquivo `linux-x.y.z/include/linux/cred.h`. Esta estrutura define as credenciais de um processo, em especial sob qual id de usuário ele executa. As operações que um processo podem executar dependem deste id. O usuário root possui id 0 e qualquer processo executando sob este id de usuário tem permissões de super usuário. Para ver os ids atuais de um interpretador de comandos, execute o comando "`id`". Mais informações sobre os ids de usuários podem ser encontradas executando "`man credentials`", na seção "*User and Group Identifiers*".

Em `task_struct` as credenciais de um processo estão em `task_struct->cred`. Porém, este campo é do tipo `const struct cred *`. Isto significa que não é possível alterá-lo diretamente. Para alterar as credenciais é necessário primeiro obter uma referência não constante, alterar a estrutura e depois liberá-la com os métodos:

struct cred *get_cred(const struct cred *): Prepara para alteração uma estrutura cred e retorna um ponteiro mutável para ela.

void put_cred(const struct cred *): Marca o fim da edição e libera o ponteiro mutável.

Algumas outras funções e macros úteis (mas não essenciais para este projeto) podem ser encontradas em `cred.h`.

Ao fazer modificações no núcleo do Linux podemos corromper estruturas de dados fundamentais ao sistema. Isto pode, por exemplo, danificar os sistemas de arquivos. Então, é recomendado que você faça cópias frequentes do código do seu módulo para a máquina real.