Sistemas Operacionais B

Projeto #1

1. Introdução

Este projeto deverá permitir ao aluno familiarizar-se com os detalhes de implementação de um módulo de kernel que faz uso da API criptográfica do kernel Linux. Espera-se que ao final do projeto, cada aluno seja capaz de implementar, compilar, instalar e testar um novo módulo de kernel que realize as funções de cifrar, decifrar e calcular o resumo critpográfico (hash) dos dados fornecidos pelo usuário.

2. Descrição do projeto

O projeto consiste na implementação de um módulo de kernel capaz de cifrar, decifrar e calcular resumo critpográfico (hash) dos dados fornecidos pelo usuário. Além do módulo de kernel, também deve ser implementado um programa em espaço de usuário para testar o módulo desenvolvido.

O módulo de kernel desenvolvido deve possuir a função de um driver de dispositivo criptográfico (*crypto device driver*) capaz de receber requisições e enviar respostas através do arquivo de dispositivo `/dev/crypto`.

Ao carregar o módulo de kernel, deve-se informar no parâmetro `key` a chave simétrica e no parâmetro `iv` o vetor de inicialização que serão usados para cifrar e decifrar os dados. Tanto a chave simétrica quanto o vetor de inicialização correspondem a uma string representada em hexadecimal (cada byte corresponde a dois dígitos hexa). A carga do módulo deve ser executada como no exemplo a seguir:

```
insmod cryptomodule.ko key="0123456789ABCDEF" iv="0123456789ABCDEF"
```

O envio de requisições ao dispositivo criptográfico deve ser realizado através de operações de escrita no arquivo de dispositivo. As requisições ao dispositivo devem ser realizadas no seguinte formato:

```
operação dados
```

onde:

`operação`: corresponde a um caracter que define qual operação será realizada pelo dispositivo, sendo permitidas as operações de cifrar (`c`), decifrar (`d`) ou calcular o resumo criptográfico (`h`);

`dados`: corresponde a uma string contendo os dados sobre os quais a operação será realizada representados em hexadecimal (cada byte corresponde a dois dígitos hexa).

O envio da resposta do dispositivo criptográfico contendo o resultado da operação solicitada deve ser realizado através de operações de leitura no arquivo de dispositivo. Para a operação de cifrar (`c`), a resposta deve ser uma string correspondendo aos dados fornecidos durante a requisição, cifrados com o algoritmo AES em modo CBC utilizando-se a chave fornecida durante a carga do módulo, representados em hexadecimal (cada byte corresponde a dois dígitos hexa).

Para a operação de decifrar (`d`), a resposta deve ser uma string correspondendo aos dados fornecidos durante a requisição representados em hexadecimal (cada byte corresponde a dois dígitos hexa), decifrados com o algoritmo AES em modo CBC utilizando-se a chave fornecida durante a carga do módulo.

Para a operação de cálculo de resumo criptográfico (`h`), a resposta deve ser uma string correspondendo ao resumo criptográfico em hexadecimal dos dados fornecidos durante a requisição, utilizando-se o algoritmo SHA1.

Para testar o correto funcionamento do driver de dispositivo criptográfico, deve ser implementado um programa em espaço de usuário que permita abrir o arquivo de dispositivo, enviar uma requisição fornecida pelo usuário (através de uma operação de escrita no arquivo de dispositivo) e exibir a resposta fornecida pelo dispositivo criptográfico (através de uma operação de leitura no arquivo de dispositivo).

O módulo de kernel desenvolvido também deve obrigatoriamente fazer uso de **MUTEX LOCKS** para bloquear um processo em espaço de usuário caso este processo acesse o arquivo de dispositivo `/dev/crypto` enquanto ele estiver sendo utilizado por outro processo em espaço de usuário.

Tanto o módulo de kernel quanto o programa de usuário devem ser compilados através de um Makefile.

3. Material complementar

Documentação do Kernel: pasta Documentation/crypto no código fonte do kernel.

Linux Kernel Crypto API: https://www.kernel.org/doc/html/v4.12/crypto/index.html

Writing a Linux Kernel Module - Part 2: A Character Device: http://derekmolloy.ie/writing-a-linux-kernel-module-part-2-a-character-device

Simple demo explaining usage of the Linux kernel CryptoAPI: http://www.logix.cz/michal/devel/cryptodev/cryptoapi-demo.c.xp

4. Resultado

O projeto deve ser acompanhado de um relatório com as seguintes partes obrigatórias:

- Introdução, indicando o que se pretende com o experimento;
- Detalhes de projeto do módulo de kernel desenvolvido, detalhando através de textos e diagramas o funcionamento interno do módulo e dos algoritmos criptográficos utilizados;
- Descrição dos resultados obtidos, detalhando o processo de compilação, instalação e teste do módulo de kernel desenvolvido, demonstrando as funcionalidades implementadas através de imagens e textos descrevendo o que está sendo testado e os resultados esperados e obtidos;
- Conclusão indicando o que foi aprendido com o experimento.

Entrega

A entrega do projeto deve ser feita de acordo com o cronograma previamente estabelecido.

Em todos os arquivos entregues deve constar **OBRIGATORIAMENTE** o nome e o RA dos integrantes do grupo.

Devem ser entregues os seguintes itens:

i. Relatórios de acompanhamento semanal individuais: cada membro do grupo deve preencher seus próprios relatórios de acompanhamento semanal descrevendo apenas as suas atividades desenvolvidas e as suas contribuições realizadas no código, com o número do commit correspondente;

ii. Código-fonte do módulo de kernel implementado;

iii. Código-fonte do programa de teste implementado em espaço de usuário;
iv. Makefile utilizado na compilação do módulo de kernel e do programa de teste;
v. Relatório final do trabalho, em formato pdf.

Solicita-se que **NÃO** sejam usados compactadores de arquivos.

Não serão aceitas entregas após a data definida. A não entrega acarreta em nota zero no experimento.

A interação entre os grupos é estimulada, no entanto qualquer tentativa de plágio será punida com a nota -Nmax para todos os envolvidos. Na dúvida do que é ou não plágio, consulte o docente.