Disciplina: Sistema Operacionais Embarcados **Código:** 120961 **Turma:** A
Professor: Diogo Caetano Garcia
Aluno/Matrícula: Fábio Barbosa Pinto – 11/0116356

Questionário: 16_LKMs_2

1. O que é memória virtual?

A memória virtual consiste em um espaço que é reservado no momento em que o sistema operacional é instalado em um computador. Essa ação independe do fato do SO ser Linux ou Windows.

2. Qual é a diferença entre o user space e o kernel space?

A RAM é dividida em duas regiões distintas - o user space e o kernel space.

User space - É um conjunto de locais onde os processos normais do usuário são executados. Esses processos não podem acessar o kernal space diretamente. Alguma parte do kernal space pode ser acessada por meio de chamadas do sistema. Essas chamadas do sistema atuam como interrupções de software no kernal space.

Kernel space - o kernel corre na parte dedicada da memória. A função do kernel space é gerenciar aplicativos / processos em execução no user space. Pode acessar toda a memória. Se um processo executar uma chamada de sistema, uma interrupção de software é enviada para o kernel, que então despacha um manipulador de interrupção apropriado.

3. Qual é a diferença de acesso à memória virtual entre o user space e o kernel space?

Adotando um sistema de 32 bits como exemplo, um ponteiro é capaz de endereçar 4 GB de memória. Destes, 2 GB são reservados para endereçar páginas privadas para cada processo, ou seja, vão compor o espaço de endereçamento privado de cada aplicação. Esta faixa, que vai de 0x00000000 ao 0x7FFFFFFF, recebe o nome de User Space e são os endereços que aplicações acessam em User Mode.

A faixa de endereços de 0x80000000 ao 0xFFFFFFFF define o então chamado System Space, que diferente do User Space, endereçam páginas de memória que não são privados a um determinado processo. Isso significa que o mesmo dado (na mesma página física) pode ser acessado através do mesmo endereço virtual em diferentes processos. Um dado em System Space pode ser acessado apenas em Kernel Mode, enquanto que um dado em User Space pode ser acessado tanto em Kernel Mode quanto em User Mode.

4. Em termos práticos, por que é feita esta distinção entre user space e kernel space?

As chamadas do sistema atuam como uma interface entre os processos do usuário e os processos do kernel. Os direitos de acesso são colocados no espaço do kernel para impedir que os usuários bagunçam o kernel, sem saber.

Assim, quando ocorre uma chamada de sistema, uma interrupção de software é enviada ao kernel. A CPU pode passar o controle temporariamente para a rotina de rotina de tratamento de interrupção associada. O processo do kernel que foi interrompido pela interrupção é retomado depois que a rotina de manipulador de interrupção termina seu trabalho.

5. Qual é a diferença entre Linux e GNU/Linux?

O Linux é o software que se comunica com o hardware do computador e o GNU é o método que temos para nos comunicar com o Linux, ou seja, com o Linux Kernel.

Eis que somando o Linux que é o Kernel, o software especial que se comunica com o hardware, seguido pelas ferramentas que usamos para nos comunicar com o Linux (Kernel), temos a soma e o resultado é o GNU/Linux.

O GNU é responsável por toda uma cadeia de ferramenta que possibilitam o desenvolvimento de programas para o Linux.

Os softwares são desenvolvidos por ferramentas GNU e o Linux Kernel ou Linux são coisas diferentes, ainda que se juntem e formem o sistema operacional.

6. O que é um Linux Kernel Module (LKM)? Quais são suas vantagens?

Um módulo carregável do núcleo, do inglês loadable kernel module (ou LKM), é um arquivo objeto que contém código para estender o núcleo em execução, ou o chamado núcleo base, de um sistema operacional. Os LKMs são normalmente usados para adicionar suporte para novos hardwares (como controladores de dispositivos) e/ou sistemas de arquivos, ou para adicionar chamadas de sistema. Quando a funcionalidade fornecida por um LKM não for mais necessária, ela pode ser descarregada com o objetivo de liberar memória e outros recursos.

- Vantagens:

Sem módulos carregáveis do núcleo, um sistema operacional teria de incluir todos as possíveis funcionalidades antecipadamente e já compiladas diretamente no núcleo base. Muitas dessas funcionalidades iria residir na memória sem serem utilizados, desperdiçando memória, e exigiria que os usuários reconstruíssem e reiniciassem o núcleo base toda vez que necessitassem de novas funcionalidades. A maioria dos sistemas operacionais que suportam os módulos carregáveis do núcleo incluirão módulos para suportar as funcionalidades mais desejadas.

7. O que são device drivers?

Também pode ser chamado de “driver de dispositivo”, tradução do termo inglês “device driver” e que talvez seja mais condizente com a função que o mesmo exerce nos computadores. O driver nada mais é que um pequeno programa cuja função é permitir aos aplicativos ou ao sistema operacional propriamente dito interagir com um dispositivo físico de hardware como uma placa de som, vídeo ou rede.

Independentemente do dispositivo ser do tipo onboard ou off-board, um driver se comunica com ele através do barramento no qual esse dispositivo encontra-se conectado ao resto da placa mãe**.**

8. Cite uma situação prática em que vale a pena desenvolver um device driver.

Por exemplo, um aplicativo de fotos, ele precisa ler algumas imagens de uma câmera fotográfica. O aplicativo terá que pedir uma solicitação ao sistema operacional, que, por sua vez, fará a solicitação ao driver.

O driver, que geralmente é desenvolvido pela mesma empresa que projetou e fabricou o dispositivo, sabe como se comunicar com o hardware para obter as fotos. Depois de o driver receber os dados da câmera, ele os devolve para o sistema operacional, que repassa para o aplicativo.

9. O que é a General Public License (GPL)? Quais são suas características?

GNU General Public License (Licença Pública Geral GNU), GNU GPL ou simplesmente GPL, é a designação da licença de software para software idealizada por Richard Matthew Stallman em 1989, no âmbito do projeto GNU da Free Software Foundation (FSF). Richard Stallman originalmente criou a licença para o Projeto GNU de acordo com as definições de software livre da Free Software Foundation.

Sendo uma licença copyleft, trabalhos derivados de um produto originalmente licenciado pela GPL só podem ser distribuídos se utilizarem a mesma licença. Isso é diferente das licenças permissivas como a licença BSD e a licença MIT, que possuem exigências mais simples.

Em termos gerais, a GPL baseia-se em 4 liberdades:

1. A liberdade de executar o programa, para qualquer propósito (liberdade nº 0)
2. A liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades (liberdade nº 1). O acesso ao código-fonte é um pré-requisito para esta liberdade.
3. A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo (liberdade nº 2).
4. A liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, de modo que toda a comunidade beneficie deles (liberdade nº 3). O acesso ao código-fonte é um pré-requisito para esta liberdade.

Com a garantia destas liberdades, a GPL permite que os programas sejam distribuídos e reaproveitados, mantendo, porém, os direitos do autor por forma a não permitir que essa informação seja usada de uma maneira que limite as liberdades originais. A licença não permite, por exemplo, que o código seja apoderado por outra pessoa, ou que sejam impostos sobre ele restrições que impeçam que seja distribuído da mesma maneira que foi adquirido.