Linux Kernel

Laboratório de Sistemas Operacionais

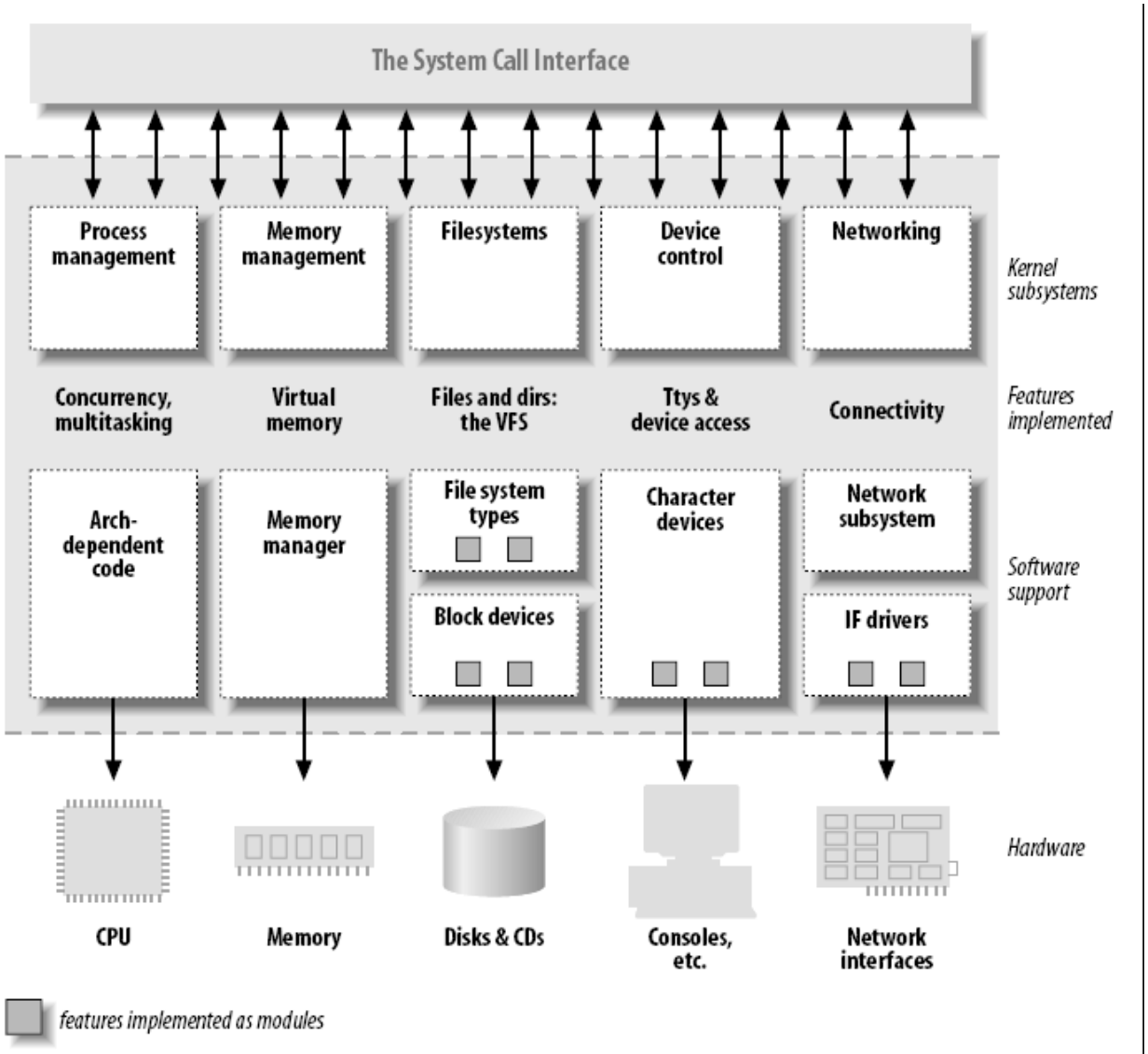
Prof. MSc. João Tavares

Introdução

- Processos concorrentes realizam tarefas diferentes
  - Processos pedem recursos do sistema
  - Processamento, memória, rede, etc.

- Kernel é uma grande área de código executável
  - Encarregada de atender as requisições dos processos
  - Distinção entre tarefas do kernel não é muito visível

Organização interna do Kernel



/usr/src/linux

- Diretório que contém o Código-fonte do Kernel
- Também pode ser encontrado on-line em https://www.kernel.org/
- Documentações adicionais no link http://jungla.dit.upm.es/~jmseyas/linux/kernel/hackers-docs.html

Código Fonte do Kernel

- Estrutura de diretórios adotada
  - Documentation/
    * informações sobre plataformas específicas e dispositivos, bem como informações gerais do kernel.
  - arch/
    * Código específico de arquitetura;
    Ex.: x86, sparc etc.
  - drivers/
    * Código específico de dispositivos;
    Ex.: som, rede etc.
  - fs/
    * Código específico de sistemas de arquivos;
    Ex.: VFS, ext4, ext3, ext2, vfat etc.

Código Fonte do Kernel

- Estrutura de diretórios adotada (continuação)

  - include/

    * Arquivos header do Kernel

    * Mantidos em subdiretórios separados de acordo com a classe do driver ou arquitetura

  - init/

    * Código associado com o processo de boot e inicialização

  - ipc/

    * código de Inter Process Communication;

    Ex.: message queues, mem. compartilhada etc.

  – **kernel**/

    * Codigo principal do kernel;

    Ex.: escalonamento, sinais etc.

Código Fonte do Kernel

- Estrutura de diretórios adotada (continuação)
  - libs/
    * Bibliotecas relacionadas com o kernel;
    Ex.: compressão, descompressão, criptografia etc.
  - mm/
    * Código relacionado com gerência de memória.
    Ex.: memória virtual, swap etc.
  - net/
    * Código relacionado com a rede.
    Ex.: protocolos de rede, firewall
  - scripts/
    * Scripts auxiliares relacionados com o código
    Ex.: patch-kernel

Uma Estrutura de Dados...

- Task é uma unidade de escalonamento do ponto de vista do kernel
  - Task_struct → representação utilizada pelo kernel
  - Definida no header include/linux/sched.h
    * Ex.: no kernel 3.6.3, estava definida na linha 937 do sched.h

- Atividade:
  - identifique a posição atual (linha) no arquivo sched.h
  - identifique que informações são registradas para cada task

Um Trecho de Código...

- Chamadas de sistema open() e read() do VFS
  - Exemplo didático
  http://www.win.tue.nl/~aeb/linux/vfs/trail.html

- Status atual implementação do kernel
  - Em fs/open.c →
    * SYSCALL_DEFINE3(open,...

  - Em fs/read_write.c →
    * SYSCALL_DEFINE3(read,...

Ferramentas

- Edição de código fonte
  - A princípio qualquer editor de textos
  - Mais usuais vim e emacs

- Desenvolvimento
  - make
    * controle de compilação/building
  - splint
    * análise estática de código, procura identificar potenciais vulnerabilidades e más práticas

    http://lclint.cs.virginia.edu/

Ferramentas

- Navegação no Código Fonte
  - grep
    * Localiza ocorrências de um string em arquivos
    Ex: $grep -r task_struct * | less

  - lxr
    * Indexador de código com interface web

Ferramentas

- Manipulação de Código Fonte
  - diff
    * Calcula diferenças entre arquivos

    Ex: $ diff -u linux-3.10.14/drivers/char/keyboard.c \
    linux-new/drivers/char/keyboard.c \
    > my_keyboard_patch
  - patch
    * aplicar patch produzido por um diff
  - git
    * Sistema de controle de versões, substituiu o CVS
    * Opera de forma totalmente distribuída
    • Permite commits offline

Principais Ações

- Compilação e instalação do kernel
  - Experimentação, Otimização e Adaptação

- Criando módulos:
  - http://tldp.org/LDP/lkmpg/2.6/html/index.html
  - http://tldp.org/HOWTO/Module-HOWTO/

- Adicionar novas chamadas de sistemas
  - É possivel fazer quase tudo sem isso!

Depurando

- Função printk() pode ser usada para gerar informação de depuração nos logs do kernel

- Alguns (mas não todos) problemas geram um dump na tela de informações de depuração
  - Conhecidas como "kernel oops"

- Para saber mais:
  http://www.urbanmyth.org/linux/oops/

Ciclo de Desenvolvimento do Kernel

- Organizado em torno de branchs em repositorio git:
  - linux-next (*live*) → desenvolvimento (*unstable*)
  * features experimentais e contribuições diversas
  - mainline (*vanilla*) → desenvolvimento (revisado)
  * 1 rodada incoporando grandes contribuições, n rodadas integrando bugfixes
  * releases a cada 2-3 meses, **mainline** antigo → **stable**
  - stable → versão considerada de produção
  * Recebe hotfixes (também incorporados ao mainline)
  * Após algum tempo torna-se:
  - **EOL** (fim-de-linha) ou
  - **longterm** → versões estáveis antigas do kernel que recebem *backporting* das correções de novos bugs

Compilação do Kernel

- **Passo 1**: Baixar o código fonte do kernel
  - Última versão disponível em http://kernel.org/
    * Nome do arquivo é **linux-***x.y.z***.tar.xz**, onde *x.y.z* e a versão do kernel.
  - Executar os comandos:

```
$ cd /tmp
$ wget https://www.kernel.org/pub/linux/kernel/
        v3.x/linux-x.y.z.tar.xz
```

- **Passo 2**: Descompactar
  - Executar os comandos:

```
$ tar -xJvf linux-x.y.z.tar.xz -C /usr/src
$ cd /usr/src
```

Compilação do Kernel

- **Passo 3**: Configurar o kernel

  - Antes de configurar o kernel e necessário ter as ferramentas de desenvolvimento instaladas

    Ex.: gcc e outras ferramentas

  - A configuração do kernel pode ser iniciada pelos comandos:

    $ make menuconfig

  - Menus de configuração baseados em texto. Muito útil para compilação via acesso remoto

    $ make xconfig ou $ make gconfig

  - Configuração em modo gráfico

  - Segunda opção otimizada para Gnome (usa GTK)

Compilação do Kernel

- **Passo 4**: Realizar a compilação propriamente dita
  - Iniciar criando a imagem comprimida do kernel:
    $ make
  - Compilar os módulos do kernel:
    $ make modules

- **Passo 5**: Instalar o novo kernel
  - Subir nível de privilégios para root
    $ su -
  - Instalar os módulos do kernel:
    $ make modules_install
  - Instalar o kernel:
    $ make install

Instala três arquivos no diretório /boot

- System.map-x.y.z
- config-x.y.z
- vmlinuz-x.y.z

x.y.z é a versão atual (ex.: 2.6.25)

Compilação do Kernel

- **Passo 6**: Criar RAM-disk com drivers que precisam ser carregados no boot

```
$ cd /boot
$ mkinitrd -o initrd.img-x.y.z x.y.z
```

  * Obs: Substitua x.y.z pela versão instalada

Obs: de fato, essa etapa é opcional se todos os componentes esseciais ao boot foram compilados na modalidade built-in

Compilação do Kernel

- **Passo 7**: Modificar configuração do *boot-loader*
  - GRUBv1: editar o arquivo /boot/grub/menu.lst

```
title     Debian GNU/Linux, kernel 2.6.25 Default
root      (hd0,0)
kernel /boot/vmlinuz root=/dev/sdb1 ro
```

(Partição raiz (“/”) do sistema.)

```
initrd   /boot/initrd.img-2.6.25
savedefault
boot
```

  - GRUBv2: arquivo /boot/grub/grub.cfg gerado com comando **grub-mkconfig**
  - No caso do Debian/Ubuntu existe a possibilidade de usar-se a deteccao automatica:

```
$ update-grub          (debian ou ubuntu)
```

Compilação do Kernel

- **Passo 8**: Reiniciar o computador para carregar o novo kernel

  ```
  $ reboot
  ```

- Mais informações: http://www.cyberciti.biz/tips/compiling-linux-kernel-26.html

Ao editar o código do kernel...

- Algumas dicas de codificação
  - Restrições / preocupações
    * Sem proteção de memória
    * Sem FPU
    * Limite rígido da Pilha (8k)
  - Código deve ser portável
  - Teste o máximo possível
  - Respeitar estilo de codificação
  - Uso de Comentários
  - Proteger estruturas de dados com *números mágicos*

Atividades complementares

- Configurar e instalar a última versão do kernel do Linux
- Analisar e descrever os principais grupos de opções de configuração durante o processo de configuração do kernel
- Analise quando se deve optar por instalar um componente como módulo? Dê exemplos

Leituras complementares

- Geral
http://www.kernelnewbies.org
http://www.linuxtopia.org/online_books/linux_kerne/kernel_configuration/index.html

- Código fonte
https://www.linux.org/

- Módulos
http://tldp.org/LDP/lkmpg/2.6/html/index.html

- Compilação do Kernel
http://www.cyberciti.biz/tips/compiling-linux-kernel-26.html

Referências Bibliográficas

- Material originalmente elaborado por Prof. Cristiano Costa. Material autorizado e cedido pelo autor. Revisado e atualizado por Prof. Luciano Cavalheiro e posteriormente pelo Prof. João Tavares.