

# Desenvolvendo o kernel Linux: Parte I

Renê de Souza Pinto

08 de Outubro de 2013

# Licença



*Desenvolvendo o kernel Linux: Parte I*, por Renê de Souza Pinto, é licenciado sob a Atribuição-Compartilhualgual 3.0 Brasil - [http://creativecommons.org/licenses/by-sa/3.0/br/deed.pt\\_BR](http://creativecommons.org/licenses/by-sa/3.0/br/deed.pt_BR)

# Índice

- 1 Introdução
- 2 A estrutura do kernel
  - Entrando no mundo do kernel...
  - Compilando e instalando
  - Módulos
  - Obtendo ajuda...
- 3 Mecanismo das chamadas ao sistema (syscalls)
  - Prática 0
- 4 Preparando-se para a parte II
  - Detalhes finais da parte I...

# Introdução

- Um pouco de história?
  - **Multics (1965): MULT**iplexed Information and **C**omputing Service
  - **1969:** Ken Thompson e o *Space Travel*
  - **1969:** Thompson e Ritchie escrevem o Unix (em assembly) para rodar o *Space Travel* em um PDP-7
  - **1972:** Ritchie cria a Linguagem C
  - **1973:** O Unix é reescrito em C
  - **1976:** Código aberto, disponível para estudo!
  - **1980:** Várias versões do Unix são lançadas e licenciadas pela AT&T, que acaba proibindo o uso do código-fonte do Unix para estudo
  - **1984:** RMS funda o projeto GNU
  - **1984-1987:** Andrew S. Tanenbaum escreve o Minix[Tanenbaum 1987]

# Introdução

- Um pouco de história?
  - **5 de Outubro de 1991:** Uma mensagem publicada via Usenet na lista comp.os.minix marcaria a história da computação. Linus Torvalds, um estudante finlandês de ciências da computação da Universidade de Helsinki trouxe ao mundo a notícia de que estava trabalhando no projeto de um sistema operacional baseado no Minix, a versão 0.02 estava pronta e seria distribuída com seu código-fonte.
  - Nascia o **Linux!**

# Introdução

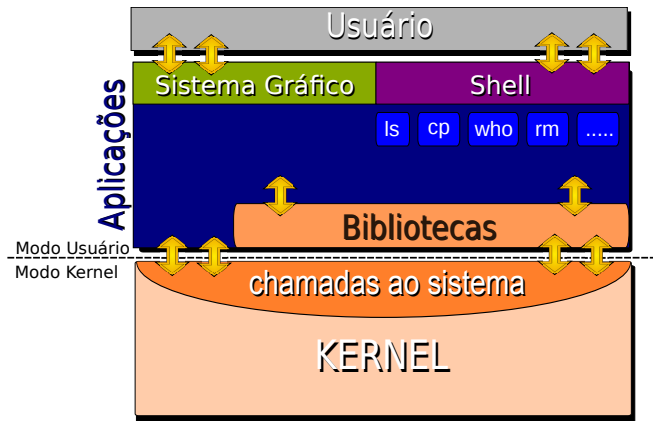
- Sistema Operacional: Segundo Tanenbaum[Tanenbaum 2000], pode ser visto como um **gerenciador de recursos** ou como uma **máquina estendida**.
- Comunica-se e controla o hardware da máquina, abstraindo-o através de uma interface (chamadas ao sistema)
- Núcleo (*kernel*):
  - Monolítico
  - Micronúcleo
  - Híbrido
  - Exonúcleo

- Monolítico X Micronúcleo:



# Introdução

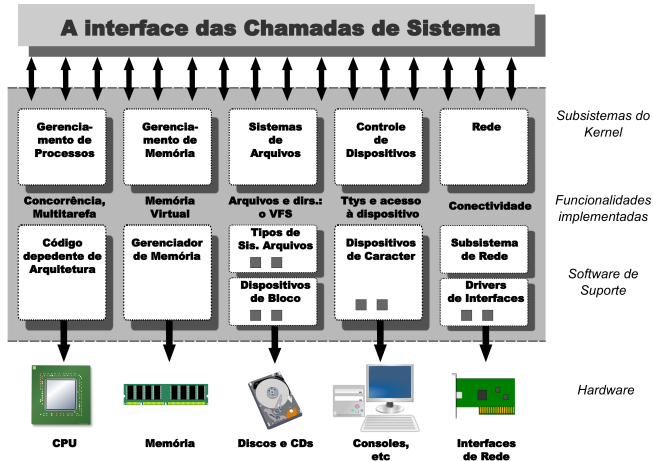
- kernel do Linux é monolítico:





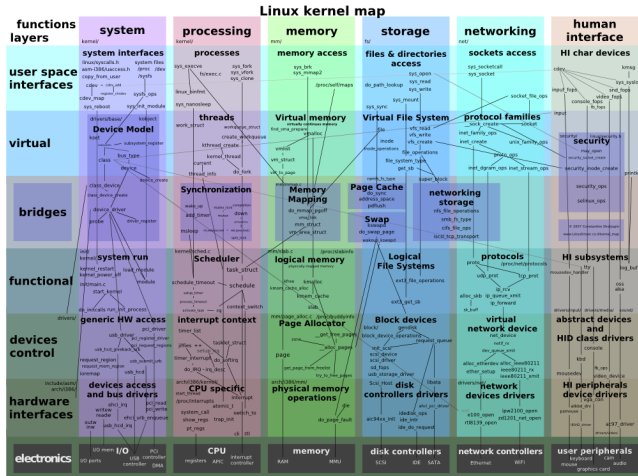
# A estrutura do kernel

- Uma visão mais geral (adaptado de [Rubini e Corbet 2001]):



# A estrutura do kernel

- Um pouco mais detalhado:



# A estrutura do kernel

- **Gerenciamento de Processos:** Escalonamento, sinais, multitarefa
- **Gerenciamento de Memória:** Mapear endereços reais no espaço virtual, alocar memória, fazer paginação (swap)
- **Sistemas de Arquivos:** Suporte a dezenas de tipos de sistemas de arquivos (EXT3, EXT4, FAT, NTFS, XFS) através da camada VFS (*Virtual File System*)
- **Controle de dispositivo:** Gerenciar acesso a dispositivos, sistema de console (TTY), *major/minor numbers*, etc
- **Rede:** Implementa a pilha de rede: Protocolos (TCP/IP, SLIP, PPP, etc), filtro/controle de pacotes, etc
- **Drivers:** Comunicação com o Hardware para permitir o controle/acesso do mesmo pelo kernel

# Entrando no mundo do kernel...

- Onde encontrar?
  - <http://www.kernel.org>
  - <ftp://ftp.kernel.org>

```
# git clone git://github.com/mirrors/linux.git  
linux
```

# Configurar / Compilar / Instalar

```
# make help
```

```
# make config  
# make menuconfig  
# make xconfig  
# make gconfig
```

```
# make  
# make modules_install  
# make firmware_install  
# make install
```

```
# make deb-pkg
```

# Compilando e instalando

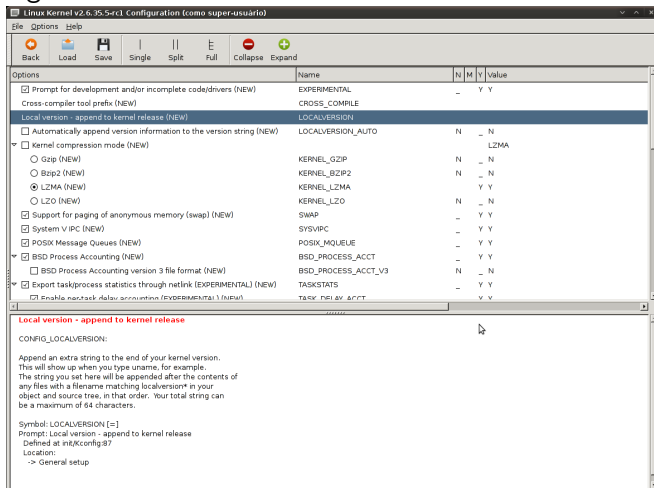
- Configuração: Todas as configurações são salvas no arquivo **.config** do diretório do código-fonte do kernel. Pode ser editado manualmente ou com o auxílio de interfaces (texto, gráfico) providas pelo próprio sistema de compilação do kernel.
- Instalação:
  - Imagem do kernel, uma cópia do arquivo de configuração e a tabela de Símbolos<sup>1</sup> ficam em **/boot**
  - Módulos ficam em **/lib/modules/VERSAO\_KERNEL**. Ex:  
*/lib/modules/3.11.0*
  - Firmwares ficam em **/lib/firmware** ou  
**/lib/firmware/VERSAO\_KERNEL**

---

<sup>1</sup> Contém os nomes e endereços do kernel para auxiliar na depuração.

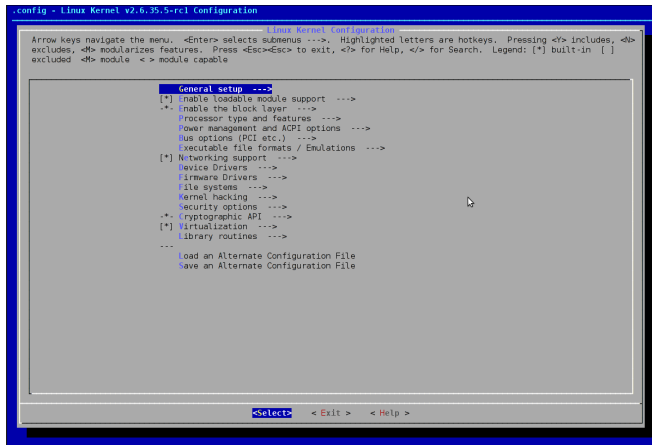
# Configurando...

- gconfig:



# Configurando...

- menuconfig:





# Configurando...

Seções principais:

- **General setup:** Opções gerais do kernel
- **Enable loadable module support:** Suporte ao carregamento/d Descarregamento de módulos
- **Enable the block layer:** Suporte para dispositivos de bloco
- **Processor type and features:** Configurações específicas do processador, arquitetura, etc.
- **Power management and ACPI options:** Controle de energia
- **Bus options (PCI etc.):** Opções de barramentos
- **Executable file formats / Emulations:** Opções de formatos de arquivos executáveis / emulações

# Configurando...

- **Networking support:** Suporte a rede (protocolos, etc).
- **Device Drivers:** Seção que contém todos os drivers dos dispositivos (hardware) suportados pelo kernel
- **Firmware Drivers:** Suporte a determinados firmwares
- **File Systems:** Suporte a sistemas de arquivos
- **Kernel hacking:** Opções para debug
- **Security options:** Opções de segurança
- **Cryptographic API:** Opções de criptografia, suporte a algoritmos e hardware do gênero
- **Virtualization:** Opções para virtualização
- **Library routines:** Aplicação de rotinas CRC (Cyclic Redundancy Check)

# Configurando...

- Configurar/Compilar o kernel para um determinado hardware implica em conhecer bem esse hardware:

```
# lspci
# dmidecode
# cat /proc/cpuinfo
# lsusb
# lsmod
```

# Módulos

- Módulos podem ser inclusos diretamente no kernel (built-in) ou compilados em arquivos separados, podendo ser inseridos/removidos do kernel dinamicamente (posteriormente)
- Para saber informações (autor, parâmetros, etc) de um módulo:

```
# modinfo <módulo>
```

- Carregar/Remover módulos:

```
# modprobe <módulo>
# insmod </path/módulo.ko>
# rmmod <módulo>
```

# Módulos

- **depmod**: Módulos podem fornecer serviços para outros módulos, assim, quando o módulo A usa um serviço de B, então A depende de B. Resolver estas dependências pode ser muito complexo. O utilitário **depmod** resolve as dependências e gera o arquivo *modules.dep*, que fica no diretório de módulos.

```
# depmod -a
```

# Obtendo ajuda...

- **/usr/src/Linux/Documentation**

```
# make htmldocs  
# make pdfdocs  
# make mandocs
```

- Livros: [Rubini e Corbet 2001], [Love et al. 2005], [Bach 1986], etc...
- <http://kernelnewbies.org/> / <http://br.kernelnewbies.org/>
- Listas de e-mail (kernelnewbies, etc)
- Leia os fontes!
- Google

# Mecanismo das *syscalls*

- As chamadas ao sistema ocorrem por interrupção:
  - Registrador EAX: Contém o número da *syscall* que se deseja executar
  - Registradores EBX, ECX, EDX: Parâmetros a serem passados para a *syscall*
  - Interrupção 0x80: Instrução *INT \$0x80* é executada
  - O kernel atende a interrupção: Muda para contexto do kernel, salva contexto atual (registradores, pilha, etc) e executa a *syscall*. Quando terminada, volta para o contexto de usuário.

# Praticando...

Vamos colocar as mãos na massa!



# Mecanismo das *syscalls*

## ● Prática 0:

- Objetivo: Executar uma chamada de sistema diretamente a partir de um programa de usuário escrito em assembly
- Vamos usar uma máquina virtual:
  - Usuário: **ckernel**
  - Senha: **ckernel**
  - Senha de root: **ckernel**
- Acesse o diretório `/usr/src/CURSO_KERNEL`, crie uma pasta com seu nome
- Crie a pasta *pratica0* e escreva o arquivo `hello.S`
- Para compilar:

```
# as --32 hello.S -o hello.o
# ld -melf_i386 hello.o -o hello
```

# Preparando-se para continuar...

- O kernel está em constante desenvolvimento, as mudanças são enviadas a lista de e-mail do kernel através de *patches* escritos por programadores espalhados por todo mundo.
- Os *paches* oficiais são sempre aplicados na versão base anterior, por exemplo:
  - patch-2.6.35 → atualiza a versão 2.6.34
  - patch-2.6.35.4 → atualiza a versão 2.6.35
- E se eu estiver com a versão 2.6.34.6 e quiser atualizar para 2.6.35.3?
  - Situação inicial: 2.6.34.6
  - Reverta para 2.6.34 (reverter patch-2.6.34.6)
  - Atualize para 2.6.35 (aplique patch-2.6.35)
  - Atualize para 2.6.35.3 (aplique patch-2.6.35.3)

# Preparando-se para continuar...

- Mas, como aplicar o *patch*?
- *patches* são gerados com a ferramenta **diff** e aplicados/revertidos com a ferramenta **patch**
- Para aplicar:

```
# patch -p<num> < <arquivo_patch>
```

- No kernel do linux:

```
# cd /usr/src/linux-2.6.35
# wget http://www.kernel.org/pub/linux/kernel/
v2.6/patch-2.6.35.5.bz2
# bunzip2 patch-2.6.35.3.bz2
# patch -p1 < patch-2.6.35.3
```

# Preparando-se para continuar...

Fiz alterações no kernel, e agora, como criar meu *patch*?

- Kernel hackers de primeira viagem: Leiam os arquivos `/usr/src/linux/Documentation/SubmittingPatches` e `/usr/src/linux/Documentation/CodingStyle`
- Suponha que a pasta com código original seja **linux-3.11.0-orig** e sua árvore de desenvolvimento seja **linux-3.11.0-foo**:

```
# diff -uprN -X linux-3.11.0-orig/Documentation/  
dontdiff \  
linux-3.11.0-orig linux-3.11.0-foo > /tmp/patch
```

- Dica: use o script `Lindent` (`scripts/Lindent`) para indentar automaticamente seu código de acordo com os padrões do código-fonte do kernel.

# Preparando-se para continuar...






- Para reverter um *patch* utilize a opção -R:

```
# cd /usr/src/linux-3.11  
# patch -Rp1 < patch-3.11.1
```

- Alterações de um *patch* que não puderem ser aplicadas (pois são incompatíveis com o arquivo original) geralmente são gravadas em um novo arquivo com o mesmo nome do original, porém com a extensão **.rej**

Ufa! Por hora é só!

# Referências I

-  BACH, M. J. *The design of the Unix Operating System*. [S.l.]: Prentice-Hall, 1986.
-  LOVE, R. et al. *Linux Kernel Development Second Edition*. [S.l.]: Pearson Education, USA, 2005.
-  RUBINI, A.; CORBET, J. *Linux device drivers*. [S.l.]: O'Reilly Media, 2001.
-  TANENBAUM, A. A UNIX clone with source code for operating systems courses. *ACM SIGOPS Operating Systems Review*, ACM, v. 21, n. 1, p. 29, 1987.
-  TANENBAUM, A. S. *Sistemas Operacionais: projeto e implementação*. [S.l.]: Bookman, 2000.