Universidade de São Paulo
Brasil

# Grupo de Estudos Linux


## Compilando o kernel do Linux

Renê de Souza Pinto

# Compilando o kernel do Linux

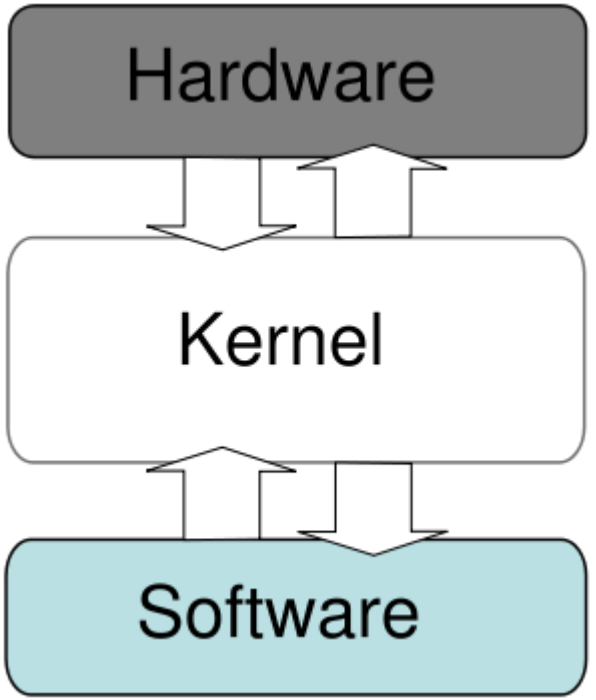
- Sumário:
  - O que é o kernel?
  - A estrutura do kernel
  - Porque compilar o kernel?
  - Compilando o kernel
  - FAQ
  - Dicas
  - Referências Bibliográficas

# O que é o kernel?

- O que é um Sistema Operacional (S.O.)?
  - Segundo Tanenbaum[1], um sistema operacional pode ser visto como gerenciador de recursos ou como uma extensão da máquina virtual

# O que é o kernel?

- É o coração do Sistema Operacional. O kernel é o responsável por prover o ambiente virtual ao programador, tornando-se a camada entre o hardware e o software.



# O que é o kernel?

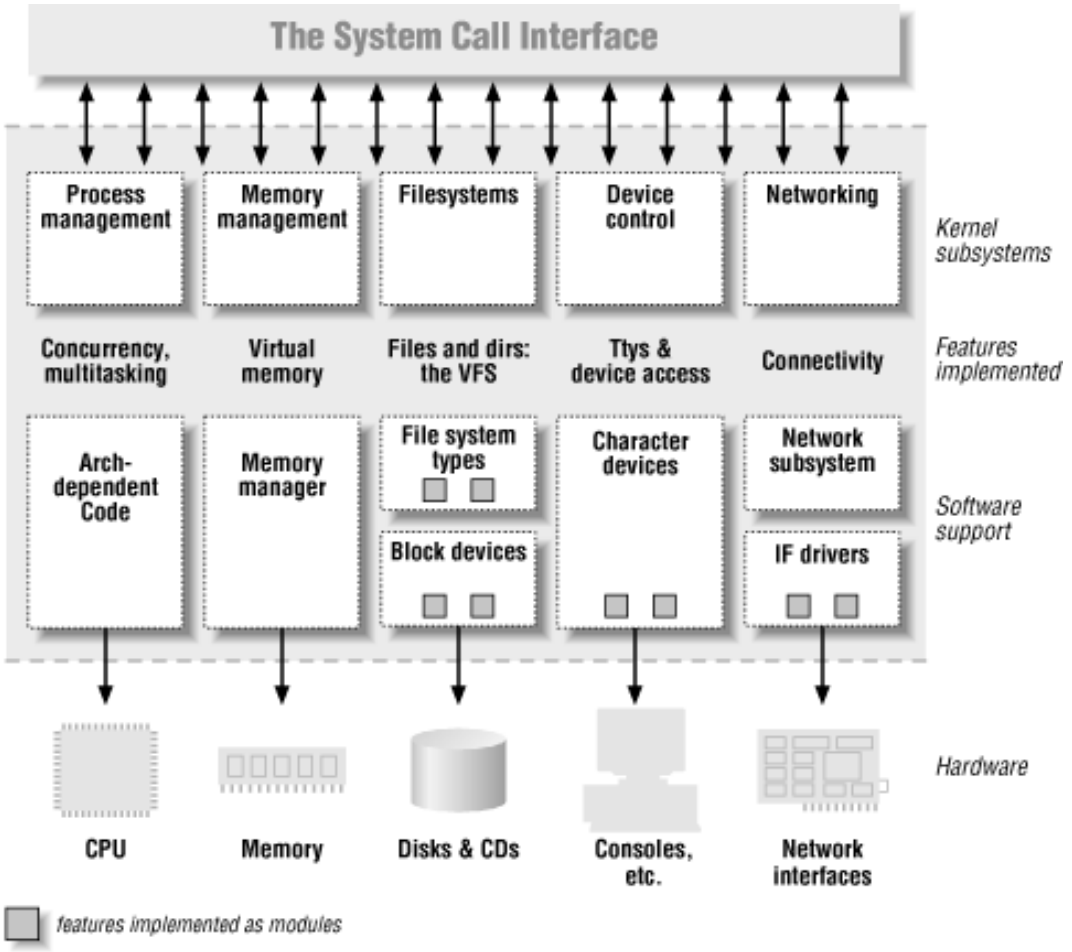
- O Shell e o Servidor X são exemplos de ambientes que permitem o usuário usufruir de todo ambiente criado pelo kernel, ou seja, fazem a camada entre o kernel e o usuário.

# O que é o kernel?

- O kernel prove um conjunto de instruções, chamadas *system calls*, ou simplesmente chamadas de sistema. Através destas chamadas é que se torna possível o acesso a todos os recursos da máquina e providos pelo sistema.
- O kernel do Linux segue o padrão POSIX (Portable Operating System Interface for Unix). Este padrão define as chamadas de sistemas, o que torna possível a compatibilidade entre sistemas Unix-Like.

# A estrutura do kernel

- A figura abaixo ilustra a organização do kernel do Linux:



# A estrutura do kernel

- Controle de Processos: Gerencia os processos do sistema (sinais, escalonador, etc)

# A estrutura do kernel

- Controle de Processos: Gerencia os processos do sistema (sinais, escalonador, etc)
- Controle de Memória: Gerencia a memória (paginação, swap, etc)

# A estrutura do kernel

- Controle de Processos: Gerencia os processos do sistema (sinais, escalonador, etc)
- Controle de Memória: Gerencia a memória (paginação, swap, etc)
- Sistema de arquivos: Cuida dos sistemas de arquivos suportados pelo sistema (ext3, reiserfs, nfs, etc)

# A estrutura do kernel

- Controle de Processos: Gerencia os processos do sistema (sinais, escalonador, etc)
- Controle de Memória: Gerencia a memória (paginação, swap, etc)
- Sistema de arquivos: Cuida dos sistemas de arquivos suportados pelo sistema (ext3, reiserfs, nfs, etc)
- Controle de dispositivos: Gerencia os dispositivos (hardware) do sistema

# A estrutura do kernel

- Controle de Processos: Gerencia os processos do sistema (sinais, escalonador, etc)
- Controle de Memória: Gerencia a memória (paginação, swap, etc)
- Sistema de arquivos: Cuida dos sistemas de arquivos suportados pelo sistema (ext3, reiserfs, nfs, etc)
- Controle de dispositivos: Gerencia os dispositivos (hardware) do sistema
- Controle da Rede: Gerencia os pacotes da rede (coleta, identificação, envio, etc)

# A estrutura do kernel

- O kernel é implementado através de módulos, estes módulos podem estar integrados ao mesmo (*built-in*) ou carregados e des-carregados dinamicamente.
- Módulos permitem que o kernel se adapte ao sistema em que roda, tornando-o dinâmico, rápido e até mesmo inteligente!

# Porque compilar o kernel?

- Compilar seu próprio kernel permite adaptar o sistema e otimizá-lo totalmente para a máquina em que o mesmo irá atuar.

# Porque compilar o kernel?

- Compilar seu próprio kernel permite adaptar o sistema e otimizá-lo totalmente para a máquina em que o mesmo irá atuar.
- Permite selecionarmos os módulos que dão suporte ao hardware necessário e descartar os módulos dos hardwares que não compõem a máquina em que o kernel irá atuar.

# Porque compilar o kernel?

- Compilar seu próprio kernel permite adaptar o sistema e otimizá-lo totalmente para a máquina em que o mesmo irá atuar.
- Permite selecionarmos os módulos que dão suporte ao hardware necessário e descartar os módulos dos hardwares que não compõem a máquina em que o kernel irá atuar.
- Economia de memória

# Porque compilar o kernel?

- Compilar seu próprio kernel permite adaptar o sistema e otimizá-lo totalmente para a máquina em que o mesmo irá atuar.
- Permite selecionarmos os módulos que dão suporte ao hardware necessário e descartar os módulos dos hardwares que não compõem a máquina em que o kernel irá atuar.
- Economia de memória
- kernel enxuto e dinâmico

# Compilando o kernel

- O kernel do Linux é distribuído oficialmente pelo site:
  - http://www.kernel.org

# Compilando o kernel

- Ultima versão estável do kernel até o momento:
  - 2.6.18.1

# Compilando o kernel

- Ultima versão estável do kernel até o momento:
  - 2.6.18.1
- Nada melhor do que entendermos a compilação do kernel compilando um!

# Compilando o kernel

- Ultima versão estável do kernel até o momento:
  - 2.6.18.1
- Nada melhor do que entendermos a compilação do kernel compilando um!
- NOTA: Para evitar erros de acesso, os passos descritos a partir de agora devem ser feitos através do super-usuário (root)

# Compilando o kernel

- Baixe o código-fonte do kernel a ser compilado. Neste exemplo usaremos a versão 2.6.18.1
  - http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.18.1.tar.bz2

# Compilando o kernel

- Baixe o código-fonte do kernel a ser compilado. Neste exemplo usaremos a versão 2.6.18.1
  - http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.18.1.tar.bz2
- Extraia para o diretório: /usr/src
  - *tar -xvjf linux-2.6.18.1.tar.bz2 -C /usr/src*

# Compilando o kernel

- Baixe o código-fonte do kernel a ser compilado. Neste exemplo usaremos a versão 2.6.18.1
  - http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.18.1.tar.bz2
- Extraia para o diretório: /usr/src
  - *tar -xvjf linux-2.6.18.1.tar.bz2 -C /usr/src*
- Crie o link simbólico chamado linux:
  - *ln -s linux-2.6.18.1 linux*

# Compilando o kernel

- Baixe o código-fonte do kernel a ser compilado. Neste exemplo usaremos a versão 2.6.18.1
  - http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.18.1.tar.bz2
- Extraia para o diretório: /usr/src
  - *tar -xvjf linux-2.6.18.1.tar.bz2 -C /usr/src*
- Crie o link simbólico chamado linux:
  - *ln -s linux-2.6.18.1 linux*
- Acesse o link criado:
  - *cd linux*

# Compilando o kernel

- As informações sobre a compilação do kernel ficam armazenadas no arquivo *.config* (por default), mas através das interfaces descritas a seguir podemos carregar e salvar essas informações a partir de outros arquivos.

# Compilando o kernel

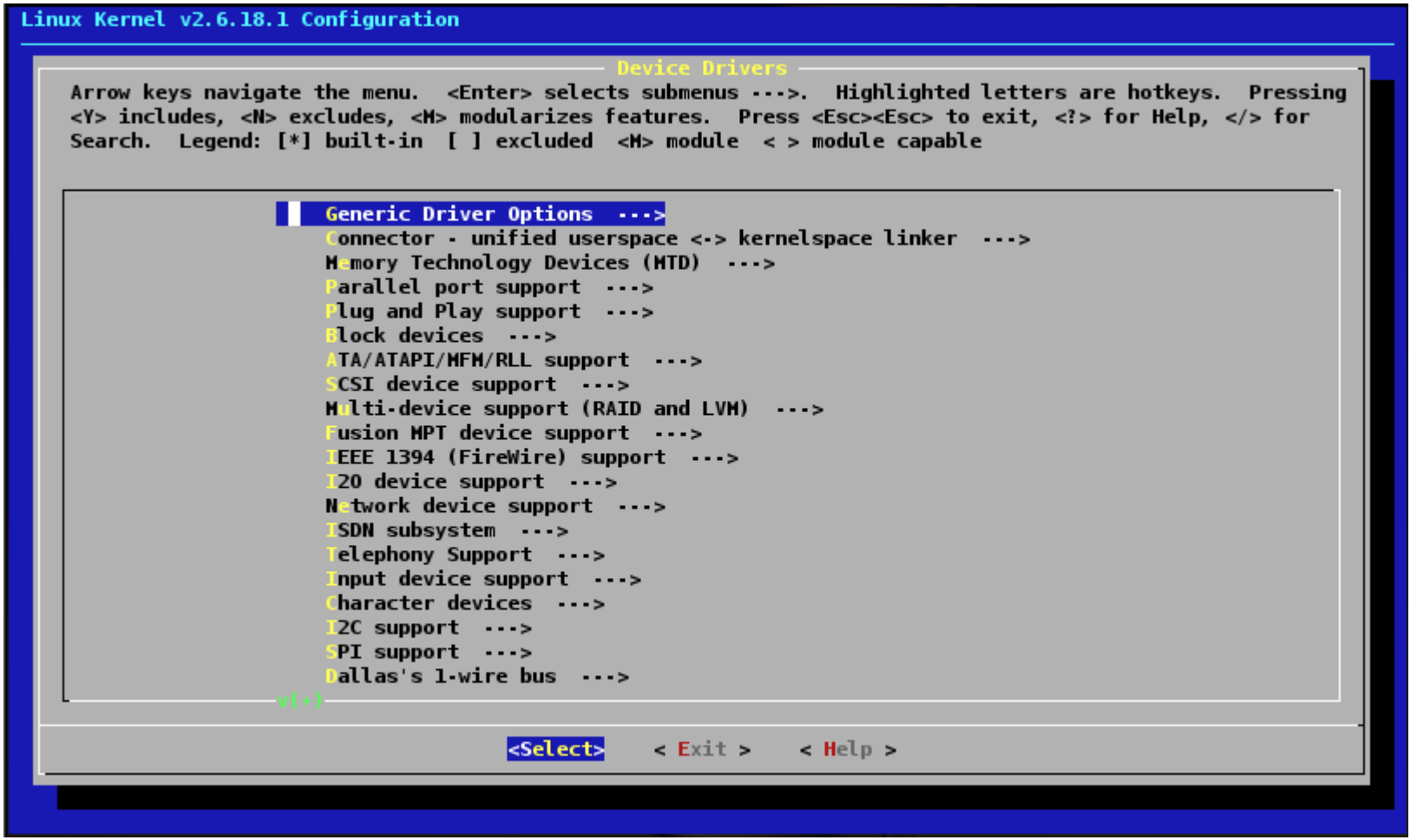
- Existem quatro interfaces para a configuração da compilação do kernel:
  - ***config***: Uma interface puramente texto, aonde é perguntado ao usuário como deverá ser compilado cada item do kernel.
  - ***menuconfig***: Uma interface no modo texto, porém muito mais amigável e intuitiva. É a mais utilizada.
  - ***xconfig***: Uma interface gráfica, segue o mesmo padrão do *menuconfig*.
  - ***gconfig***: Segue o mesmo modelo do xconfig, porém com interface feita em *gtk*[5].

# Compilando o kernel

- Para iniciar a configuração através de uma das interfaces utilizamos o comando *make*:
  - *make config*
  - *make menuconfig*
  - *make xconfig*
  - *make gconfig*
- Utilizaremos a interface menuconfig:
  - *make menuconfig*

# Compilando o kernel

- A figura abaixo ilustra a interface menuconfig:



# Compilando o kernel

- Para navegar no *menuconfig* basta utilizarmos as setas direcionais.
- Opções são selecionadas através da tecla <ENTER>.
- Para marcar uma opção do kernel como built-in (integrado) utilizamos a tecla y.
- Para compilar uma opção como um módulo utilizamos a tecla m.
- Para desabilitar uma opção utilizamos a tecla n.

# Compilando o kernel

- Seções Principais:
  - *Code maturity level options*: Permite a compilação de códigos que ainda estão em fase de teste, não tendo a garantia de seu funcionamento adequado.
  - *General setup*: Opções gerais do kernel.
  - *Loadable module support*: Suporte aos módulos.
  - *Block layer:* Suporte para dispositivos grandes de armazenamento, escalonadores de E/S, etc.
  - *Processor type and features*: Configurações do processador, arquitetura, etc.
  - *Power management options*: Controle de energia.

# Compilando o kernel

- Seções Principais:
  - *Bus options*: Opções de barramentos.
  - *Executable file formats*: Opções de formatos de arquivos executáveis.
  - *Networking*: Suporte a rede (protocolos).
  - *Device Drivers*: Sessão que contém todos os drivers dos dispositivos (hardware) suportados pelo kernel.
  - *File Systems*: Suporte a sistemas de arquivos.
  - *Instrumentation support*: Mecanismos para debugar (inserir mensagens) dinamicamente no kernel.
  - *kernel hacking*: Opções para debug.

# Compilando o kernel

- Seções Principais:
  - *Security options*: Opções de segurança.
  - *Cryptograhic options*: Opções de criptografia, suporte a hardware do gênero.
  - *Library routines*: Aplicação de rotinas CRC (Cyclic Redundancy Check) em módulos.

# Compilando o kernel

- É insdispensável conhecer todo o hadware presente na máquina em que o kernel irá atuar (modelos das placas de vídeo, som, rede, etc.)

# Compilando o kernel

- É insdispensável conhecer todo o hadware presente na máquina em que o kernel irá atuar (modelos das placas de vídeo, som, rede, etc.)
- Neste exemplo compilaremos um kernel para uma máquina com as seguintes configurações:
  - Processador AMD Sempron 3000+
  - HD IDE 40Gb
  - Placas de Vídeo, Som, Rede e Modem integradas, com chipset SiS.

# Compilando o kernel

- O comando *lspci* pode user utilizado para sabermos o hardware presente na máquina.
  - Exemplo:

```
# lspci
00:00.0 Host bridge: Silicon Integrated Systems [SiS] 760/M760 Host (rev 03)
00:01.0 PCI bridge: Silicon Integrated Systems [SiS] SG86C202
00:02.0 ISA bridge: Silicon Integrated Systems [SiS] SiS963 [MuTIOL Media IO] (rev 25)
00:02.1 SMBus: Silicon Integrated Systems [SiS] SiS961/2 SMBus Controller
00:02.5 IDE interface: Silicon Integrated Systems [SiS] 5513 [IDE]
00:02.6 Modem: Silicon Integrated Systems [SiS] AC'97 Modem Controller (rev a0)
00:02.7 Multimedia audio controller: Silicon Integrated Systems [SiS] AC'97 Sound Controller (rev a0)
00:03.0 USB Controller: Silicon Integrated Systems [SiS] USB 1.0 Controller (rev 0f)
00:03.1 USB Controller: Silicon Integrated Systems [SiS] USB 1.0 Controller (rev 0f)
00:03.2 USB Controller: Silicon Integrated Systems [SiS] USB 2.0 Controller
00:04.0 Ethernet controller: Silicon Integrated Systems [SiS] SiS900 PCI Fast Ethernet (rev 91)
00:06.0 CardBus bridge: Texas Instruments PCI1410 PC card Cardbus Controller (rev 02)
00:0b.0 Network controller: Broadcom Corporation BCM4318 [AirForce One 54g] 802.11g Wireless LAN Controller (rev 02)
00:18.0 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron] HyperTransport Technology Configuration
00:18.1 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron] Address Map
00:18.2 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron] DRAM Controller
00:18.3 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron] Miscellaneous Control
01:00.0 VGA compatible controller: Silicon Integrated Systems [SiS] 661/741/760/761 PCI/AGP VGA Display Adapter
```

# Compilando o kernel

- Se disponível, o arquivo /proc/cpuinfo contém informações sobre o processador da máquina.
  - Exemplo:

```
# cat /proc/cpuinfo
processor       : 0
vendor_id       : AuthenticAMD
cpu family      : 15
model           : 44
model name      : Mobile AMD Sempron(tm) Processor 3000+
stepping        : 2
cpu MHz         : 800.000
cache size      : 128 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 1
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2
syscall nx mmxext fxsr_opt 3dnowext 3dnow up pni lahf_lm ts fid vid ttp tm stc
bogomips        : 1600.60
```

# Compilando o kernel

- Inicie o *menuconfig*:
  - *`make menuconfig`*
- Vamos passar pelas sessões principais e descrever as opções que serão habilitadas, salvos os casos em que descreveremos algumas opções que não serão habilitadas.

# Compilando o kernel

- Code maturity level options:
  - *[ ] Prompt for development and/or imcomplete code/drivers*
    - Se habilitado, permite que o kernel utilize drivers ainda em desenvolvimento e sem garantia de funcionamento correto. Drivers desta categoriais são marcados pela expressão [EXPERIMENTAL]. Utilize por sua conta e risco, neste caso, deixaremos desabilitado.

# Compilando o kernel

- General setup:
  - *() Local version – append to kernel release*
    - Permite que uma string de no máximo 64 caracteres seja adicionada na string de versão do kernel. Essa string será exibida quando, por exemplo, o comando uname for utilizado.
  - **[*]** *Support for paging of anonymous memory (swap)*
    - Suporte ao paginamento de memória em arquivos (swap).

# Compilando o kernel

- General setup:
  - **[*]** *System V IPC*
    - Permite que os processos sejam sincronizados e troquem informações entre si. (IPC, sigla para Inter Process Comunication). Opção essencial, deve ficar habilitado.
  - **[*]** kernel .config support
    - Adiciona as informações de compilação no kernel, assim, podemos extrair da imagem de um kernel o arquivo *.config* e utilizá-lo em outra compilação.
  - **[*]** *Enable access to .config through /proc/config.gz*
    - Caso a opção acima esteja habilitada, permite que o arquivo .config seja acessado em /proc/config.gz

# Compilando o kernel

- Loadable module support:
  - **[*]** *Enable loadable module support*
    - Habilita a inserção (carregamento) de módulos no kernel.
  - **[*]** *Module unloading*
    - Permite que módulos sejam retirados (descarregados) do kernel.
  - **[*]** *Automatic kernel module loading*
    - Permite que algumas partes do kernel carreguem módulos automaticamente quando necessário.

# Compilando o kernel

- Block layer:
  - Não marcaremos nenhuma opção nesta sessão.
- Processor type and features:
  - *Subarchitecture Type*
    - Selecione PC-compatible
  - *Processor family*
    - Selecione o processador adequado, no nosso caso, Opteron/Athlon64/Hammer/K8.
  - *Preemption Model*
    - Selecione o modo de preempção mais adequado, no nosso caso, *Voluntary Kernel Preemption (Desktop).*

# Compilando o kernel

- Processor type and features:
  - **[*]** *Machine Check Exception*
    - Permite que o processador avise ao kernel sobre algum erro ocorrido, assim o kernel pode avisar o usuário ou até mesmo desligar automaticamente o sistema, em caso de erros críticos. Para saber se seu processador é compatível com a esta tecnologia cheque o arquivo /proc/cpuinfo, se a flag mce estiver presente, o processador é compatível.
  - **[*]** *MTRR (Memory Type Range Register) support*
    - Permite controlar o acesso da memória do sistema pelo processador de forma mais eficiente, fazendo com que operações de leitura ou escrita fiquem até 2,5 vezes mais rápida.

# Compilando o kernel

- Processor type and features:
  - **[*]** *Use register arguments*
    - Instrui o gcc para utilizar uma função chamada ABI, a qual passa os três primeiros parâmetros de uma função através dos registradores, e não da pilha, gerando ganho de velocidade.

# Compilando o kernel

- Power management options (ACPI, APM):
  - **[*]** *Legacy Power Management API*
    - Habilita suporte a API antiga do gerenciador de energia, ainda utilizada por alguns programas.
  - *ACPI (Advanced Configuration and Power Interface) Support:*
    - **[*]** *ACPI Support*
      - Suporte ao controle de energia (ACPI), deve ser habilitado.
    - **[*]** *Sleep States*
      - Permite colocar o computador em modo de espera.

# Compilando o kernel

- Power management options (ACPI, APM):
  - *ACPI (Advanced Configuration and Power Interface) Support:*
    - **[*]** *AC Adapter*
      - Permite que o kernel identifique quando o sistema está sendo alimentado por uma fonte (adaptador). Utilizado em notebooks.
    - **[*]** *Battery*
      - Permite que o kernel pegue informações da bateria, como marca, modelo, quantidade de carga disponível. Utilizado em notebooks.
    - **[*]** *Video*
      - Habilita operações básicas de controle em placas de vídeo onboard.

# Compilando o kernel

- Power management options (ACPI, APM):
  - *ACPI (Advanced Configuration and Power Interface) Support:*
    - **[*]** *Fan*
      - Permite o controle das ventoinhas (ligado, desligado, status).
    - **[*]** *Processor*
      - Habilita suporte de processadores com recursos de economia de energia, como por exemplo, setar frequências menores quando a utilização da CPU for baixa.
    - **[*]** *Thermal Zone*
      - Habilita suporte aos sensores da motherboard.

# Compilando o kernel

- Power management options (ACPI, APM):
  - *APM (Advanced Power Management) BIOS Support:*
    - **[*]** *APM (Advanced Power Management) BIOS support*
      - Habilita o uso de funções da BIOS para economia de energia. O APM é responsável por fornecer informações de energia (status da bateria, sensores, etc) e gerar eventos quando necessário (mudança de status, etc).
    - **[*]** *Enable PM at boot time*
      - Habilita o APM no momento do boot. Importante deixar habilitado.

# Compilando o kernel

- Bus options (PCI, PCMCIA, EISA, MCA, ISA):
  - **[*]** *PCI support:*
    - Habilita suporte ao barramento PC.
      - *PCI access mode*
        - A opção any, permite que o barramento seja acessado de diversas maneiras. Selecione-a.
  - **[ ]** *PCI Express support*
    - Habilita suporte ao barramento PCI Express, deve ser selecionada somente quando a motherboard possuir este barramento.
  - **[ ]** *ISA support*
    - Habilita suporte ao barramento ISA (antigo). Selecione-a só quando necessário (idem item anteiror).

# Compilando o kernel

- Executable file formats:
  - **[*]** *Kernel support for ELF binaries*
  - **[*]** *Kernel support for a.out and ECOFF binaries*
  - **[*]** *Kernel support for MISC binaries*
    - Estas três opções habilitam o suporte a vários formatos de arquivos binários, é aconselhável selecionar as três para evitar problemas de compatibilidade.

# Compilando o kernel

- Networking:
  - **[*]** *Networking support*
    - Habilita suporte a rede, deve ser selecionada.
  - *Networking options:*
    - **[*]** *Packet socket*
      - Habilita a comunicação direta com dispositivos de rede, sem passar por um protocolo do kernel.
      - **[*]** *Packet socket: mmapped IO*
        - Permite uma comunicação mais rápida no modelo citado acima.
    - **[*]** *Unix domain sockets*
      - Suporte ao sockets (mecanismo de comunicação utilizado em sistemas Unix).

# Compilando o kernel

- Networking:
  - *Networking options:*
    - **[*]** *TCP/IP networking*
      - Habilita suporte ao protocolo TCP/IP. Definitivamente você deve habilitar esta opção :)
      - **[*]** *IP: IPsec transport mode*
      - **[*]** *IP: IPsec tunnel mode*
        - Opções do protocolo que são aconselháveis a ficarem habilitadas.
    - **[*]** *Network packet filtering (replaces ipchains)*
      - Opções para filtrar e cortar os pacotes da rede. A utilização de Firewalls requer esta opção habilitada.

# Compilando o kernel

- Device Drivers:
  - As opções desta sessão devem ser selecionadas de acordo com o hardware presente na máquina em que o kernel irá atuar.

# Compilando o kernel

- File systems:
  - **<*>** *Second extended fs support*
    - Habilita suporte para o sistema de arquivos ext2 (padrão Linux).
  - **<*>** *Ext3 journalling file system support*
    - Habilita suporte para o sistema de arquivos ext3 (baseado no ext2 só que com sistema journalling).
  - **<*>** *Reiserfs support*
    - Habilita suporte para o sistema de arquivos Reiserfs (muito utilizado atualmente).

# Compilando o kernel

- File systems:
  - **<*>** *ROM file system support*
    - Habilita suporte para se montar um pequeno sistema de arquivos na memória RAM, a maioria das distribuições utilizam este sistema para montar um pequeno disco virtual com um sistema básico, que será responsável por carregar os módulos necessários pelo sistema e depois transferir a raiz para o sistema de arquivos do disco rígido.
  - CD-ROM/DVD Filesystems:
    - **[*]** *ISO 9660 CDROM file system support*
      - Sistema de arquivos padrão utilizado pelos CD-ROMS.

# Compilando o kernel

- File systems:
  - CD-ROM/DVD Filesystems:
    - **[*]** *Microsoft Joliet CDROM extensions*
      - Uma extensão do ISO9660 criada pela Microsoft para permitir nomes de arquivos grandes no formato unicode.
    - **[*]** *UDF file system support*
      - Novo sistema de arquivos utilizados em CD-ROMs e DVDs.
  - DOS/FAT/NT Filesystems:
    - **[*]** MSDOS fs support
      - Habilita suporte para partições criadas e utilizadas pelo MS-DOS.

# Compilando o kernel

- File systems:
  - DOS/FAT/NT Filesystems:
    - **[*]** VFAT (Windows-95) fs support
      - Habilita suporte para partições criadas e utilizadas pelo Windows com suporte a nomes longos de arquivos.
    - **[*]** *NTFS file system support*
      - Habilita suporte ao sistema de arquivs NTFS (utilizado pelo Windows 2000, XP, 2003).

# Compilando o kernel

- File systems:
  - Pseudo filesystems:
    - **[*]** */proc/kcore support*
      - Permite que toda memória da máquina seja acessada pelo arquivo virtual /proc/kcore.
    - **[*]** *Virtual memory file system support (former shm fs)*
      - Permite montarmos sistemas de arquivos virtuais na memória.
  - Network File Systems:
    - **[*]** *SMB file system support*
      - Permite montarmos compartilhamentos Windows.

# Compilando o kernel

- File systems:
  - Network File Systems:
    - **[*]** *CIFS support*
      - Habilita suporte para um cliente CIFS (Common Internet File System), um sistema de arquivos sucessor do SMB (Server Message Block).
  - Native Language Support
    - () Default NLS Option
      - Devemos especificar qual a codificação padrão utilizada nos sistemas de arquivos para guardar os nomes dos arquivos. Utilizaremos o Português (cp860).

# Compilando o kernel

- File systems:
  - Native Language Support
    - **<*>** *Codepage 860 (Portuguese)*
      - Habilita a codificação (codepage) de nomes dos arquivos para o português.
    - **<*>** *NLS ISO 8859-1 (Latin 1; Western European Languages)*
      - Habilita a codificação (codepage) de nomes dos arquivos para o padrão ISO8859-1.
    - **<*>** *NLS UTF-8*
      - Habilita a codificação (codepage) de nomes dos arquivos para o padrão UTF-8, que provavelmente será o padrão do futuro.

# Compilando o kernel

- Kernel hacking
- Security options
- Cryptographic options
- Library routines
  - Não marcaremos nenhuma opção nestas sessões.

# Compilando o kernel

- UFA! Chega de configurações

# Compilando o kernel

- UFA! Chega de configurações
- Clique na opção *Exit* do *menuconfig* e responda *Yes* para o salvamento das alterações no arquivo *.config*.

# Compilando o kernel

- UFA! Chega de configurações
- Clique na opção *Exit* do *menuconfig* e responda *Yes* para o salvamento das alterações no arquivo *.config*.
- Agora estamos prontos para iniciar a compilação!

# Compilando o kernel

- Algumas opções:
  - *make bzImage* – Compila somente a imagem do kernel
  - *make modules* – Compila somente os módulos
  - *make modules_install* – Instala os módulos
  - *make* – Faz a compilação completa (imagem e módulos)
  - *make clean* – Limpa todos os arquivos gerados por uma compilação

# Compilando o kernel

- Vamos à compilação:
  - *make*
  - Se tudo der certo, a mensagem abaixo será exibida:

```
Kernel: arch/i386/boot/bzImage is ready  (#1)
Building modules, stage 2.
MODPOST
```

# Compilando o kernel

- A imagem do kernel foi gerada no arquivo bzImage (arch/i386/boot/bzImage).

# Compilando o kernel

- A imagem do kernel foi gerada no arquivo bzImage (arch/i386/boot/bzImage).
- É hora de instarmos os módulos:
  - *make modules_install*
    - Os módulos serão instalados em /lib/modules, numa pasta nomeada com a versão do kernel. Por exemplo, para o kernel 2.6.18.1 os módulos ficarão no diretório /lib/modules/2.6.18.1

# Compilando o kernel

- A imagem do kernel foi gerada no arquivo bzImage (arch/i386/boot/bzImage).
- É hora de instarmos os módulos:
  - *make modules_install*
    - Os módulos serão instalados em /lib/modules, numa pasta nomeada com a versão do kernel. Por exemplo, para o kernel 2.6.18.1 os módulos ficarão no diretório /lib/modules/2.6.18.1
- Agora a imagem do kernel:
  - *cp arch/i386/booz/bzImage /boot/meu_kernel*
    - Tudo o que devemos fazer e copiar a imagem para a pasta /boot, com o nome que desejarmos.

# Compilando o kernel

- Nosso kernel está compilado e instalado! Basta configurarmos o gerenciador de boot e rebootar a máquina com o novo kernel!

# Compilando o kernel

- Nosso kernel está compilado e instalado! Basta configurarmos o gerenciador de boot e rebootar a máquina com o novo kernel!
- Configuração para o Grub:

```
title=Meu_Kernel
root(XXX,X)
kernel=(XXX,X)/boot/nome_imagem root=/dev/XXX
```

**Por exemplo:**

```
title=Meu_Kernel
root(hd0,0)
kernel=(hd0,0)/boot/kernel_rene root=/dev/hda3 vga=791
```

# Compilando o kernel

- Configuração para o Lilo:

```
image = /boot/nome_imagem
root  = /dev/XXX
label = Meu_Kernel
```

**Por exemplo:**

```
image = /boot/kernel_rene
root  = /dev/hda3
label = Meu_Kernel
```

# Compilando o kernel

- Pronto! Vamos rebootar logo e utilizar nosso novo kernel!

# FAQ

- O kernel não inicia, exibe a mensagem de “kernel panic”
  - Cheque as configurações de compilação, alguma opção vital que deveria ter sido selecionada não foi, selecione-a, recompile o kernel e tente novamente.
- O kernel está rodando mais alguns dispositivos sumiram do /dev
  - As opções que dão suporte à estes dispositivos não foram compiladas ou estão como módulos, inicialmente tente o comando modprobe para carregar os módulos necessários, caso não funcione, verifique as configurações de compilação e selecione as opções corretas para o suporte do hardware.

# FAQ

- Habilitei as opções como built-in, agora minha distro acusa que não pode carregar os módulos.
  - Os módulos não serão carregados justamente porque já estão integrados ao kernel. Há duas alternativas aqui: Compilar as opções necessárias como módulos, ou remover o carregamento pela distro.
- Meu kernel está funcionando perfeitamente, muito mais rápido e dinâmico!
  - Parabéns, agora sai da frente do computador e vá tomar uma breja bem gelada!

# Dicas

- Nunca sobrescreva uma imagem de um kernel que esteja funcionando por outro recém compilado (e ainda não testado). Só o faça quando tiver certeza de que o kernel compilado funciona! É por isso também que existem os gerenciadores de boot.

# Dicas

- Nunca sobrescreva uma imagem de um kernel que esteja funcionando por outro recém compilado (e ainda não testado). Só o faça quando tiver certeza de que o kernel compilado funciona! É por isso também que existem os gerenciadores de boot.
- Leia o help de cada opção antes de selecioná-la ou não, apesar de consumir tempo, o ajudará a entender melhor o sistema e decidir como esta opção será compilada.

# Dicas

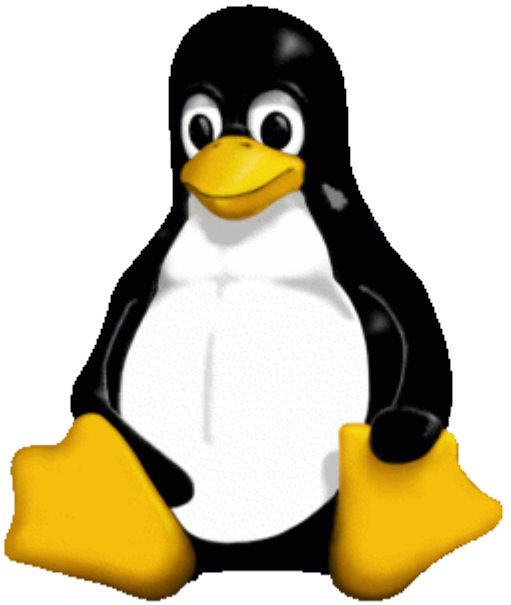
- Nunca sobrescreva uma imagem de um kernel que esteja funcionando por outro recém compilado (e ainda não testado). Só o faça quando tiver certeza de que o kernel compilado funciona! É por isso também que existem os gerenciadores de boot.
- Leia o help de cada opção antes de selecioná-la ou não, apesar de consumir tempo, o ajudará a entender melhor o sistema e decidir como esta opção será compilada.
- Tenha paciência, compilar o kernel às vezes é uma tarefa árdua, chata e demorada!

# Dicas

- USE LINUX!

# Referências Bibliográficas


- [1] Tanenbaum, A. S. **Modern Operating Systems**. 2nd Ed., 2001. Prentice Hall.
- [2] Rubini, A. **Linux Device Drivers**. 2nd Ed. 2001. O'REILLY.
- [3] Wikipedia. http://wikipedia.org
- [4] The Linux Kernel Archives. http://www.kernel.org
- [5] The GIMP Toolkit. http://www.gtk.org/



# FIM!