



Introdução ao Desenvolvimento do Kernel Linux

Versões 2.6.X

<http://www.tchelinux.org>

Palestrante: Douglas Schilling Landgraf

Email: dougslan@gmail.com

Kernel Linux

- **Sobre o palestrante**
- **Sobre a palestra**
- **Dicas iniciais**

Onde começar ? TODO List ?



[http://www.kernel**newbies**.org](http://www.kernelnewbies.org)

[http://**br**.kernelnewbies.org/](http://br.kernelnewbies.org/)



[http://kernel**janitors**.org](http://kerneljanitors.org)

[http://kernelnewbies.org/KernelJanitors/**Todo**](http://kernelnewbies.org/KernelJanitors/Todo)

[http://kernelnewbies.org/**KernelMentors**](http://kernelnewbies.org/KernelMentors)



[http://www.**kernel**.org](http://www.kernel.org)

Onde obter o código fonte ?

http://www.kernel.org

ftp://ftp.kernel.org/pub

rsync://rsync.kernel.org/pub/

Licença:

- GPL (versão 2)

<http://www.gnu.org/copyleft/gpl.html>

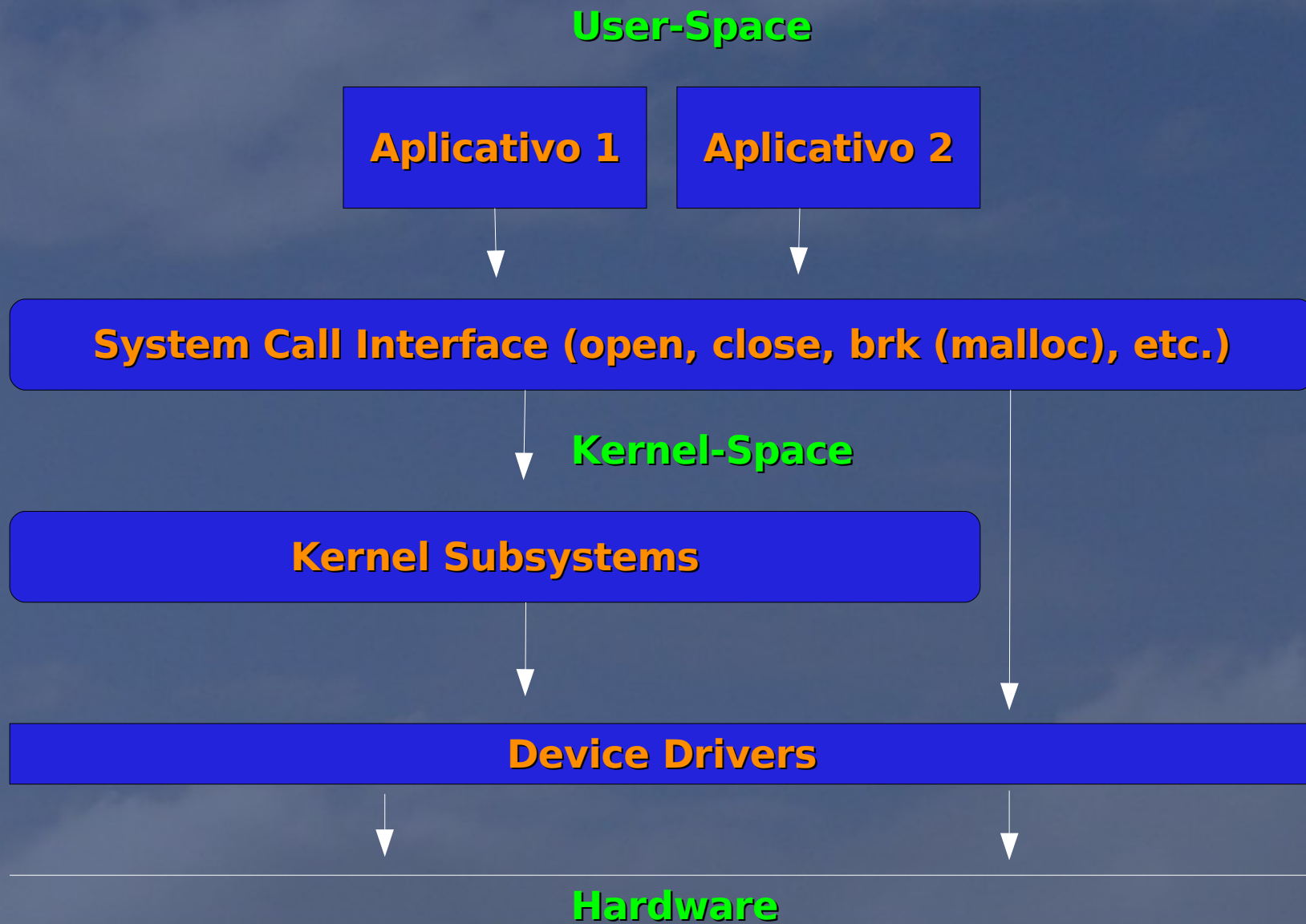
Como funciona?

- Podemos **baixar** o software e **alterar**, desde que publiquemos este software com as **licenças originais, incluindo** a disponibilização do **código fonte**.

Definição:

- É no kernel que estão **definidas funções para operação com periféricos** (mouse, discos, impressoras, interface serial/interface paralela, usb etc.), gerenciamento de memória, entre outros.
- Conjunto de programas que fornece, para os programas de usuário (aplicativos), uma **interface para utilizar os recursos do sistema**.

Kernel



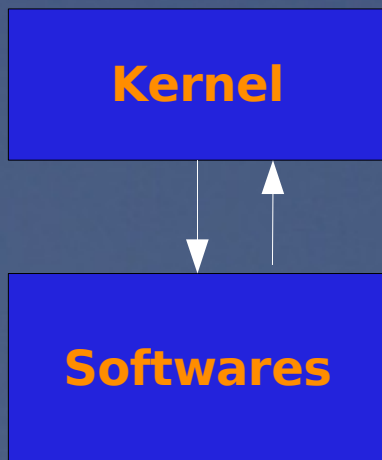
Design: Monolítico versus MicroKernel

- **Monolítico:**
 - Criado em meados de 1980
 - Um grande e único processo (imenso)
 - Comunicação trivial (todos rodam em um único processo)
- **MicroKernel:**
 - Separado em dois processos (“servers” / user-space)
 - Comunicação via IPC (interprocess communication)
 - Modularidade

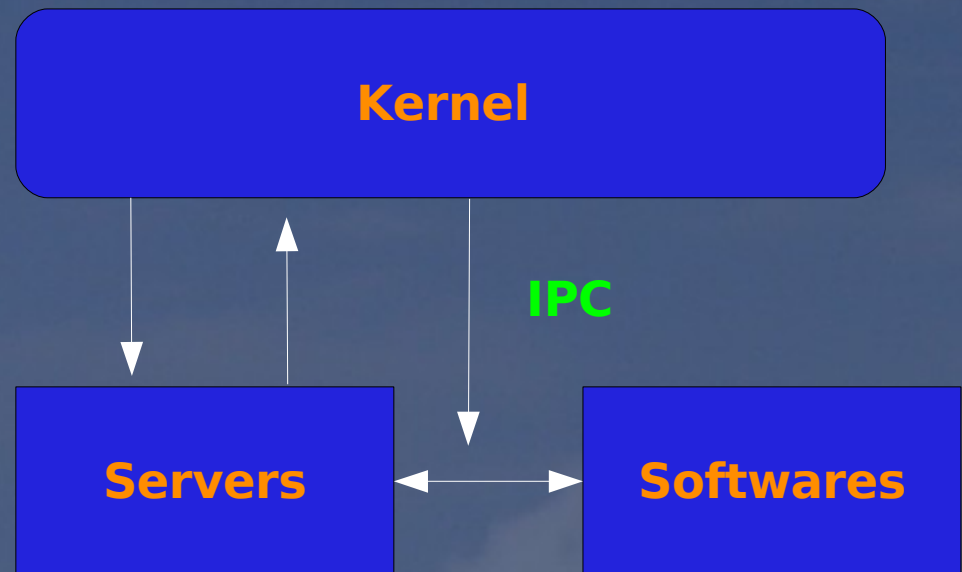
Kernel

Exemplo:

Monolítico



MicroKernel



IPC = Interprocess Communication

Design:

- Executa em um **único processo**
- Kernel Linux é **monolítico**
- Implementa **recursos do microkernel design**
 - Capacidade de **carregar dinamicamente módulos**
 - Suporte a **processadores SMP** (symmetrical multiprocessor)
 - Sistema de acesso ao dispositivos (**sysfs**)

Kernel Linux

Versões:

Estáveis:

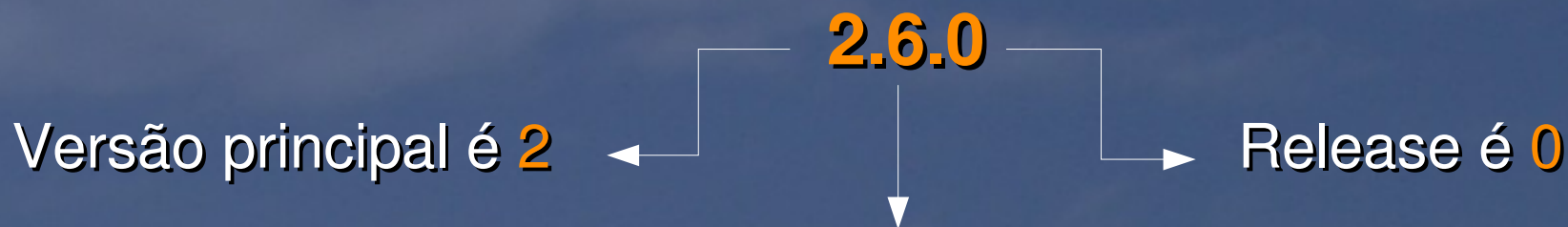
- Correções de bugs
- Novos drivers

Desenvolvimento:

- Alterações frequentes e drásticas
- Desenvolvedores testam novas soluções

Kernel Linux

Versões:



Números pares = versões estáveis

Ex.: (2.0, 2.4, 2.6)

Números ímpares = versões de desenvolvimento

Ex.: (1.3, 2.3, 2.5)

Ex.: `linux-2.6.20.3.tar.bz2`

Versões, e hoje como funciona?

- Hoje **só temos** a versão 2.6.X
- Decisão tomada no Linux Kernel Developers Summit (**2004**)
- Todas novas implementações são enviadas para o **Andrew Morton** (**Árvore -mm**)
- Após **tornar-se estável** o patch entra na versão principal.
- Versão 2.7 \sim **Árvore -mm**

Diretórios:

<code>arch/</code>	Código específico de arquitetura
<code>crypto/</code>	API de criptografia
<code>Documentation/</code>	Documentação
<code>drivers/</code>	Device Drivers
<code>fs/</code>	File systems
<code>include/</code>	Headers
<code>init/</code>	Kernel boot
<code>ipc/</code>	Interprocess communication
<code>kernel/</code>	Core do Kernel

Diretórios:

<code>lib/</code>	Bibliotecas
<code>mm/</code>	Gerenciamento de Memória
<code>net/</code>	Sistema de Rede
<code>scripts/</code>	Scripts em geral
<code>security/</code>	Subsistema de segurança
<code>sound/</code>	Susbsistema de som
<code>usr/</code>	initramfs

CodingStyle:

Identação 1 TAB = 8 caracteres (!= 8 espaços)

Colunas 80

```
if(teste) {  
    blah();  
} else {  
    bleh();  
}  
if(foo)  
    bar();
```

CodingStyle:

Funções:

1, 2 páginas?

< 10 variáveis locais ?

Comentários:

```
/*
```

```
* Olá, Eu sou um comentário!
```

```
*/
```

Outros: typedef, structs, etc.

CodingStyle:

Ferramenta indent:

```
$ indent -kr -i8 -ts8 -sob -180 -ss -bs -ps1 <arquivo>
```

ou

```
$ scripts/Lindent
```

Compilando/Carregando/Listando/Descarregando:

Atenção aos WARNINGS

```
$ linux/drivers/net> vi hello.c
```

```
$ linux/drivers/net> vi Makefile
```

```
        obj-m += hello.o
```

```
$ linux/drivers/net> make -C /usr/src/linux SUBDIRS=$PWD modules
```

```
$ insmod ./modulo.ko
```

```
$ lsmod
```

```
$ modprobe modulo.ko (procura por dependências)
```

```
$ rmmod modulo.ko
```

Kernel Linux

hello.c

```
#include <linux/init.h>
#include <linux/module.h>
MODULE_LICENSE("Dual BSD/GPL");

static int hello_init(void)
{
    printk(KERN_ALERT "hello!\n");
    return 0;
}
```

Kernel Linux

hello.c

```
static void hello_exit(void)
{
    printk(KERN_ALERT "Goodbye\n");
}
```

```
module_init(hello_init);
module_exit(hello_exit);
```

Depuração com printfk():

printfk() ~= printf()

```
printfk(KERN_WARNING "mensagem de warning!\n");
```

```
printfk(KERN_DEBUG "mensagem de debug!\n");
```

```
printfk(<1> "mensagem de alerta!!\n");
```

<linux/kernel.h> valores das MACROS {0,1,2,3, ... 7}

Prioridade: 0 -> 7

Ferramentas:

<code>diff</code>	Ferramenta para comparar arquivos
<code>patch</code>	Ferramenta para aplicar patches
<code>quilt</code>	Scripts para manutenção de patches
<code>vimdiff</code>	Ferramenta para comparar arquivos
<code>qemu</code>	Emulador
<code>git</code>	Controle de fontes/versões
<code>ctags</code>	Tags no código fonte
<code>cscope</code>	Navega no código fonte
<code>ketchup</code>	Ferramenta para atualização do kernel

Kernel Linux

Ctags:

```
$ make tags
```

```
$ vi .vimrc
```

```
    set tags=/usr/src/linux/tags
```

```
:ta printk
```

```
$ vim -t printk
```

```
CTRL + ]
```

```
CTRL + t
```

<http://ctags.sourceforge.net>.

Ferramentas Diff e Patch:

```
$ diff -ruN linux-x.y.z/ linux/ > meu-patch.diff
```

- r Recursivo
- u Formato compreensivo
- N Incluir arquivos novos

```
$ patch -p1 < ../meu-patch.diff (diretório abaixo)
```

- p1 Indica qual ponto da árvore ele vai aplicar o patch
linux/drivers/net/arquivo.c

Ferramentas Diff e Patch (exemplo):

```
--- linux-2.6.20.3.orig/drivers/net/ni65.c
+++ linux-2.6.20.3/drivers/net/ni65.c
@@ -295,7 +295,7 @@ static void ni65_set_performance(struct
 */
static int ni65_open(struct net_device *dev)
{
-   struct priv *p = (struct priv *) dev->priv;
+   struct priv *p = dev->priv;
```

Ferramenta Quilt:

```
$ mkdir patches
```

```
$ quilt new nome-do-patch.diff
```

```
$ quilt add nome_do_arquivo
```

```
$ quilt refresh
```

```
$ quilt top
```

```
$ quilt diff
```

```
$ quilt pop [-f] [-a]
```

```
$ quilt push [-f] [-a]
```

```
$ quilt remove
```

Enviando um patch:

SEM anexos, patches **INLINE**

Mensagem em **TEXTO PURO** (**SEM HTML**)

Você testou ?

Escolheu a **lista certa?**

O patch esta conforme o **CodingStyle?**

Mensagem: O que o patch faz com detalhes

Assunto: [PATCH] arquivo.c O que ele faz

Assinatura: Signed-off-by: Autor <email>

No **máximo um patch por email** (Depende de outro patch?)

Enviando um patch (exemplo):

To: kerneljanitors@....

Subject: [PATCH] ni65.c: cleanup not needed casts

Mensagem: Removed all unnecessary casts.

Signed-off-by: Douglas Schilling Landgraf <dougslan@gmail.com>

--- linux-2.6.20.3.orig/drivers/net/ni65.c

+++ linux-2.6.20.3/drivers/net/ni65.c

@@ -295,7 +295,7 @@ static void ni65_set_performance(struct
*/

static int ni65_open(struct net_device *dev)

Kernel Linux

Instalando o código fonte:

Diretório padrão:

`/usr/src/linux` (Devemos usar esse path ?)

Descompactando:

```
$ tar xvjf linux-x-y-z.tar.bz2
```

```
linux-x.y.z/Documentation/device-mapper/linear.txt
```

```
linux-x.y.z/Documentation/device-mapper/snapshot.txt
```

```
linux-x.y.z/Documentation/device-mapper/striped.txt
```

Compilando:

\$ make help	ajuda
\$ make mrproper	Remover todos os arquivos + .config + backup
\$ make config	modo texto
\$ make menuconfig	modo texto (ncurses)
\$ make xconfig	modo gráfico (Xwindows)
\$ make gconfig	modo gráfico (GTK+)

Kernel Linux

Compilando:

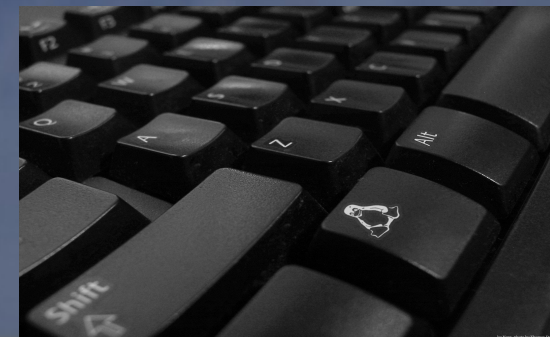
Opções:

[*] - Habilitado (built-in) [] - Não esta habilitado

[M] - Habilitado (Módulo)

\$ **make** Compilando o Kernel

\$ **make modules_install** Instalando os módulos



Compilando:

```
# cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-2.6.X
```

vmlinuz = Kernel Linux Compactado

```
# cp /usr/src/linux/System.map /boot/System.map-2.6.X
```

System.map = Tabela de Símbolos

```
# cp /usr/src/linux/.config /boot/config-2.6.X
```

Copiar o .config para /boot (backup)

Compilando:

Será necessário criar uma imagem inicial para que o kernel carregue alguns módulos básicos (IDE, SCSI, RAID) antes de acessar o filesystem.

```
# mkinitrd -k /boot/vmlinuz-2.6.X -i /boot/initrd-2.6.X
```

ou

```
# mkinitrd /boot/initrd-2.6.X.img 2.6.X
```

Compilando (Grub boot loader):

```
$ vi /boot/grub/menu.list
```

```
title Kernel-2.6.X-default
```

```
root (hd0,5)
```

```
kernel /boot/vmlinuz-2.6.X-default root=/dev/hda6
```

```
vga=0x314 resume=/dev/hda5 splash=silent showopts
```

```
initrd /boot/initrd-2.6.X-default
```

Kernel Linux

Compilando (LILO boot loader):

```
$ vi /etc/lilo/lilo.conf
```

```
image=/boot/vmlinuz-2.6.X-default
```

```
label=2.6.X
```

```
root=/dev/hda3
```

```
read-only
```

```
$ /sbin/lilo
```

Grava as configurações

```
$ reboot
```

Reiniciando o sistema

```
$ uname -a
```

Exibe o kernel atual

Documentação:

linux-2.6.X/Documentation

Linux Kernel Development 2rd Edition (Robert Love)

ISBN: 0-672327201

Linux Device Drivers 3rd Edition (Cobert, Rubini, Kroah-Hartman)

ISBN: 0-596-00590-3

Versão Online (free): <http://kroah.com/lkn/>

Linux Weekly News

<http://www.lwn.net>

Google

<http://www.google.com>



Dúvidas ? Sugestões?

<http://tchelinix.org>

<http://dougsland.livejournal.com>

Palestrante: Douglas Schilling Landgraf

Email: dougsland@gmail.com