



Introdução ao Kernel Linux

Carlos Eduardo Maiolino
Software Maintenance Engineer, Red Hat

Agenda

- Conceitos básicos de SO
- Características do kernel Linux
- Boot sequence
- System Calls
- Depurando o kernel
- Como se tornar um desenvolvedor do kernel
- Onde obter mais informações
- Bibliografia
- Contato



Conceitos Básicos de SO

- O que é Kernel ?
 - “Núcleo” de um sistema operacional
 - Interface de comunicação entre o usuário e o hardware
 - Camada mais baixa de abstração de software



Conceitos Básicos de SO

- Mas para que serve ?
 - Gerenciar recursos do sistema (hardware e software)
 - Prover uma interface simples entre o usuário e o hardware
 - Segurança (principalmente contra usuários :)



Conceitos Básicos de SO

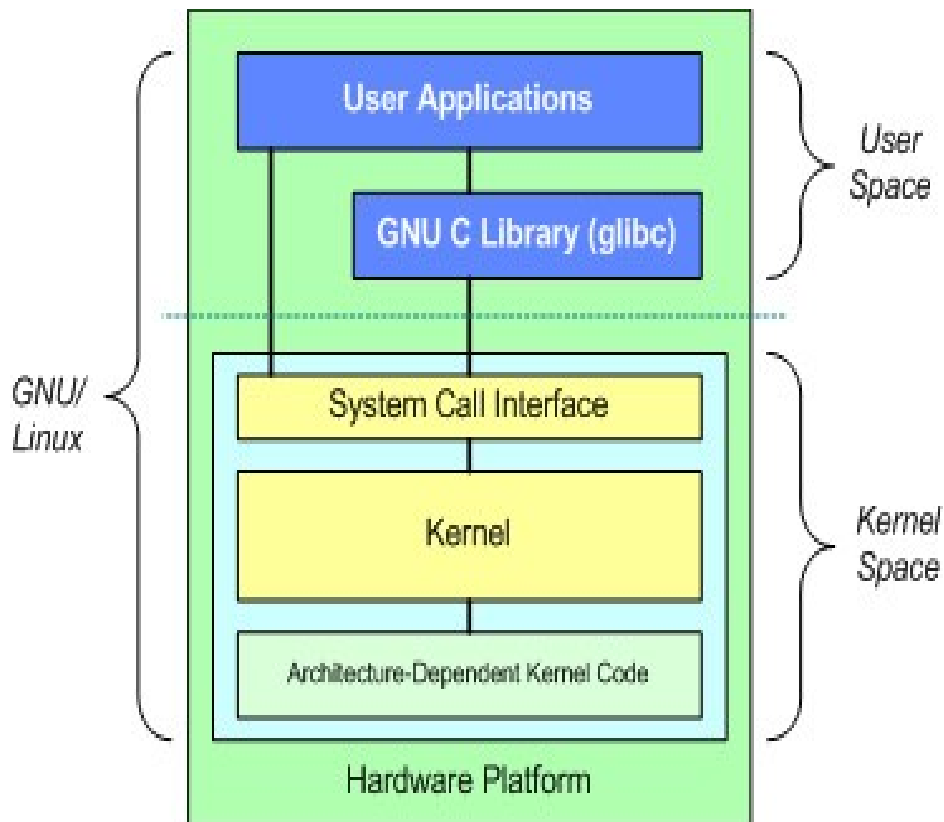
- Segurança
 - Limita e/ou restringe o acesso a determinados recursos
 - Programas de usuário não tem acesso direto ao hardware
 - Utiliza algumas características de hardware para forçar estas limitações
 - “Isolamento” das aplicações dos usuários
 - User space Vs. Kernel space – User mode Vs. Supervisor mode



Conceitos Básicos de SO

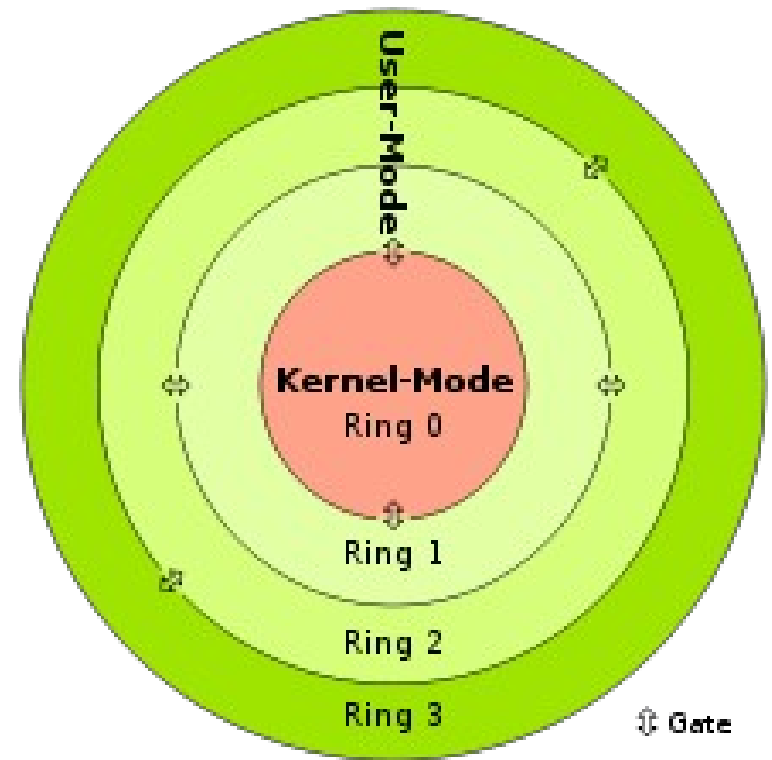
User space X Kernel space

(Visão do kernel)



User mode X Supervisor mode

(Visão da CPU)



Conceitos Básicos de SO

“As pessoas não utilizam um sistema operacional, estas, utilizam programas.”

-Linus Torvalds



Características do kernel Linux

- Multiusuário / Multitarefa
 - Executar diversas tarefas de diferentes usuários “ao mesmo tempo” de forma independente.
 - “ao mesmo tempo” significa que as aplicações utilizam de forma concorrente o tempo da CPU, porém, esta sempre estará executando apenas uma ÚNICA tarefa por vez.



Características do kernel Linux

- Kernel Monolítico
 - Um grande e único executável contendo (quase) todas as camadas necessárias para gerenciamento do hardware
 - Permite a inserção de módulos
 - A maior parte dos drivers de dispositivos são implementados como módulos



Características do kernel Linux

- Reentrante
 - Diversos processos podem estar executando em kernel mode ao mesmo tempo
 - Em sistemas com apenas uma CPU, diversos processos podem estar “bloqueados” em kernel mode aguardando pela CPU ou a conclusão de alguma tarefa de I/O.



Características do kernel Linux

- Preemptivo
 - Permite suspender a execução de processos para iniciar a execução de outro, quando em kernel space
 - Preempções podem ser desabilitadas
 - Conceito exclusivo de kernel space



Características do kernel Linux

- Multiprocessamento (SMP)
 - Possibilita a utilização de diversas CPUs
 - Possibilita a execução de diversas tarefas ao mesmo tempo
 - “True parallelism”
 - Sincronismo de execução é um dos maiores desafios
 - spin_locks, mutexes, semaforos, desabilitar preempções e interrupções e o big kernel lock (ugh!), são formas de controlar o sincronismo do kernel entre os diversos processadores.

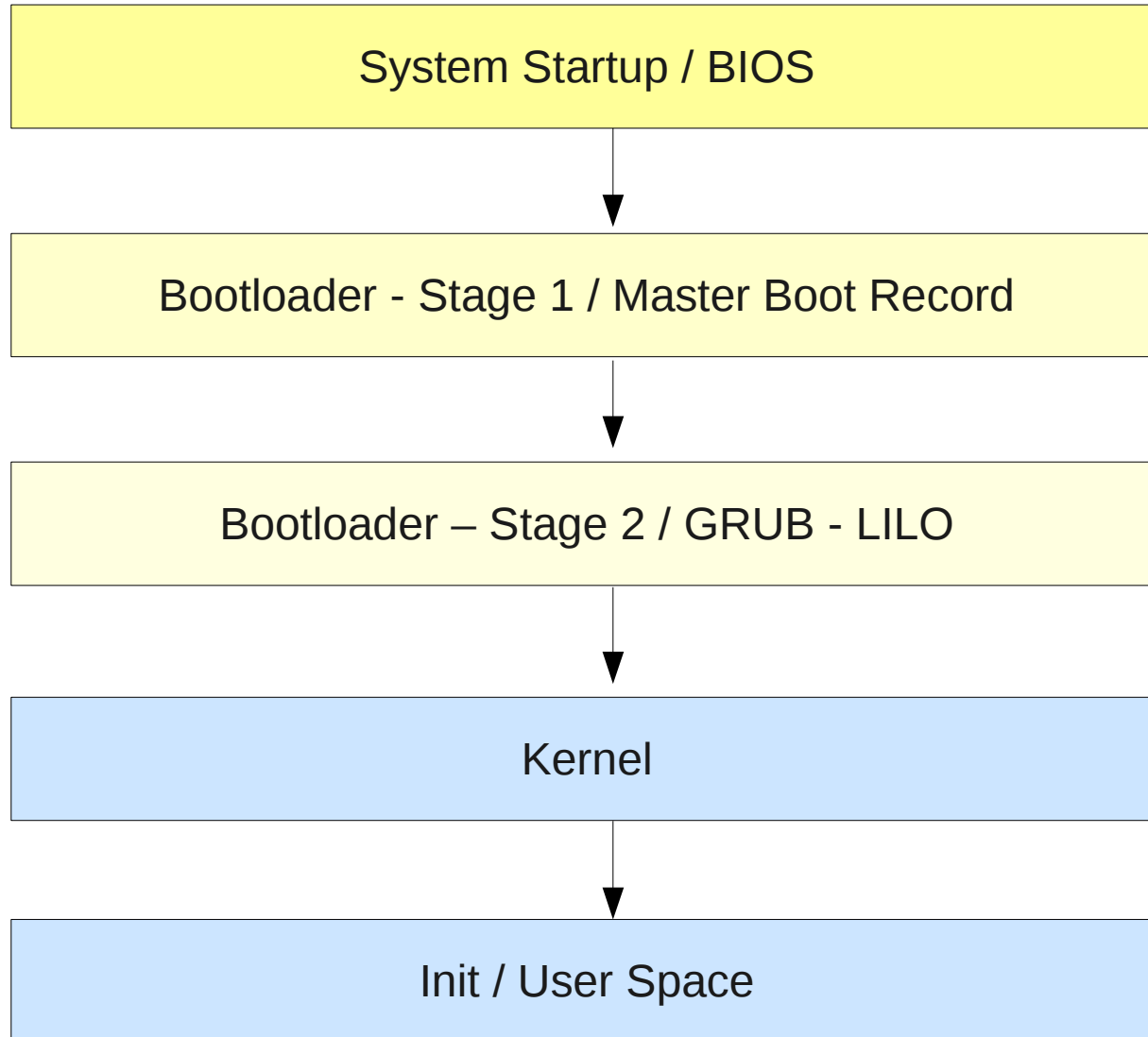


Características do kernel Linux

- Outras características:
 - OpenSource (leia, modifique, estude, aprenda)
 - Free (você não precisa comprar para utilizar)
 - Totalmente customizável
 - Não necessita de hardwares super potentes
 - Pode ser muito compacto (após customizado)
 - Compatível com muitos outros SOs
 - Excelente fonte de aprendizado

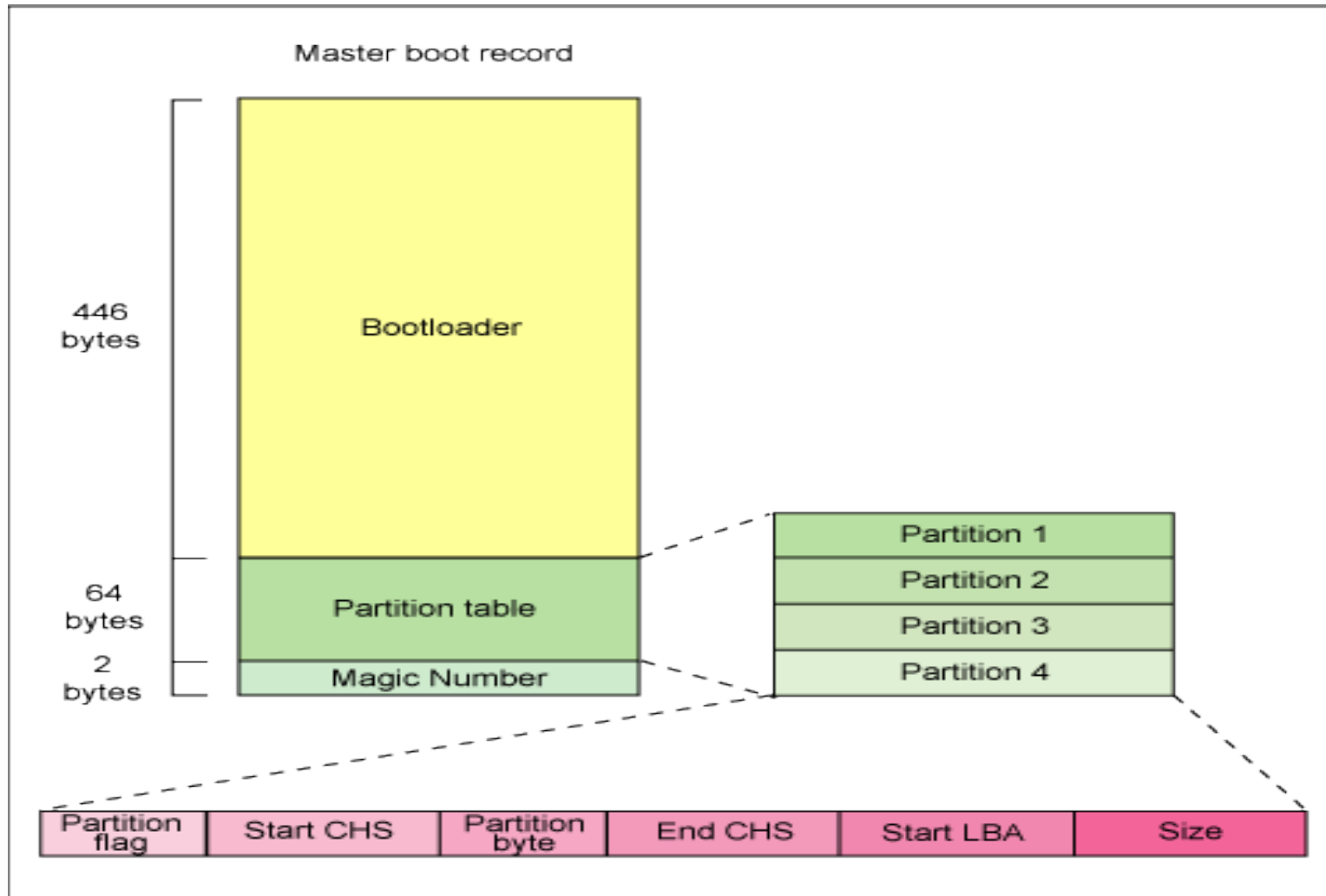


Boot Sequence



Boot Sequence

- Stage 1 – Anatomia da MBR



System Calls

- Permite aplicações de usuário de acessar dispositivos de hardware de maneira “segura”
- Executar instruções em kernel-space
- Facilita o desenvolvimento de aplicações em User space
- Portabilidade
- O Linux possui 238 system calls na versão atual (`linux2.6/arch/x86/kernel/syscall_table_32.S`)



Depurando o kernel

- Systemtap
 - Permite através de 'probes', obter informações do kernel em tempo real, através de scripts desenvolvidos na própria linguagem do systemtap
- Ftrace
 - Inicialmente desenvolvido para monitorar funções do kernel, porém, atualmente é possível efetuar diversos traces em interrupções, scheduler, e outros
- Perf
 - Mais utilizado para tratar questões relacionadas a performance



Depurando o Kernel

- Para utilizar o systemtap são necessários:
 - A instalação da ferramenta em si
 - Necessário que o kernel seja compilado com os símbolos de depuração
 - Algumas distribuições disponibilizam estes símbolos em pacotes separados, como exemplo o pacote kernel-debuginfo nas distribuições baseadas em empacotamento RPM
 - Conhecimento da linguagem systemtap



Depurando o Kernel

- Systemtap
 - Exemplos:

```
probe begin{  
    printf("hello world\n");  
}  
  
probe end{  
    printf("Good bye, cruel world\n");  
}
```



Depurando o Kernel

- Systemtap
 - Exemplos úteis :)

```
probe kernel.function("sys_open") {  
    printf("function sys_open called by %s  
    (%d) \n"execname(), pid());  
}
```



Como se tornar um desenvolvedor

- Proficiência em linguagem C
- Conhecimento no funcionamento do hardware
- Testar código dos outros é um bom começo
- Documentando código já escrito
- Programar para kernel é diferente de programar para user space
- Faça porque goste, não por obrigação
- Paciência (muita paciência :)



Obtendo mais informações

- <http://www.kernel.org>
- LKML (Linux kernel Mail List)
- <http://www.kernelnewbies.org>
- <http://www.lwn.net>
- IRC (Sim, ainda existe e é o principal “ponto de encontro” de kernel hackers)
 - irc.freenode.net / irc.oftc.net
- Livros
- <http://www.google.com/linux>



Contato

- E-mail
 - cmaiolino@maiolino.org
 - cmaiolino@redhat.com
- IRC
 - cmaiolino AT irc.freenode.net
 - #linuxfs / #ext4 / ##kernel / #fedora-br
- Gtalk
 - cybersonic0@gmail.com



Bibliografia

- <http://www.kernel.org>
- <http://www.kernelnewbies.org>
- Kernel Documentation (linux2.6/Documentation)
- Google :)
- Understanding the Linux Kernel (Daniel Bovet, Marco Cesati)
- Linux Device Drivers (Corbert, Rubini & Kroah-Hartman)
- Linux Kernel Development (Robert Love)



Dúvidas

Questions ?

Dúvidas ?

Fragen ?

