LINUX KERNEL INTERNALS

Saturday, February 8, 2014

Kernel Space and User Space

Understanding of Kernel space and User space in detail is very important if you wish to have a strong base of Linux Kernel.

- Here Kernel Space and User Space corresponds to their Virtual address space.
- Every process in linux utilizes its own separate virtual space.
- In a linux system based on 32 bit Architecture, user space address space corresponds to lower 3GB of virtual space and kernel space the upper 1GB.(general way)
- The kernel space virtual address space is shared between all the processes.
- When a process is active, it can either be running in "user mode" or "kernel mode".
- In a process is running in User mode it means that the CPU is running the user space side of code.
- A process running in the user mode has limited capability and is controlled by a flag in the CPU.
- Even though the kernel memory is present in the process's memory map the user space code is not allowed to access the kernel space code.(can do in some special way).
- When a process wants to do something other than move data around in its own (userspace) virtual memory, like opening a file for example, it must make a syscall to communicate with the kernel space.
- Each CPU architecture has it's unique way of making a system call but the basic remains the same i.e.
- A magic instruction is executed, the CPU turns on the "privileged mode" flag, and jumps to a special address in kernel space, the "syscall entry point".(read another post to understand what syscall is)
- Now when the syscall has reached the kernel space then the process is running in kernel mode and executing instructions from the kernel space memory.
- Taking the same example of open system call, to find the requested file, the kernel may consult with filesystem drivers (to figure out where the file is) and block device drivers (to load the necessary blocks from disk) or network device drivers and protocols (to load the file from a remote source).
- These drivers can be either built in or can be loaded as module but the key point that remains it that they are the part of kernel space.
- Loading a module is done with a syscall that asks the kernel to copy the module's code and data into kernel space and run its initialization code in kernel mode.
- If the kernel can't process the request then the process is made to sleep by the kernel and when the request is complete then the syscall returns back to the user space.
- Returning back to user mode means restoring the CPU registers to what they were before coming to Kernel Mode and changing the CPU privilege level to non-privilege .
- Apart from syscalls there are some other things that take CPU to kernel mode eg.

1. Page faults- If the process tries to access a virtual memory address that doesn't have a physical address assigned to it then the CPU enters the Kernel mode and jumps to page fault handler and the kernel sees whether the virtual addresss is valid or not and depending upon this it either tries to create a physical page for the given virtual address or if it can't then sends a segmentation fault signal (SIGSEGV).

Learn Linux Kernel from Android Perspective

Android is powered by Linux Kernel

Study Topics

- Process Management
- Kernel Space and User Space
- Linux Addressing
- Stack and Heap
- System Calls
- Paging and Segmentation
- Memory Management
- Interrupts
- Bottom Halves
- Timers and Time Management
- Android Boot Sequence
- Browse Linux Source Code
- Download Latest Linux Kernel
- Good Linux Kernel Questions.

Linux Kernel Questions and Answers

- Linux Kernel In short
- What is a Linux Device Driver Model ?
- How to Pass Command Line Arguments to a Kernel Module?
- What are the Possible Task States ?
- What is a Loadable Kernel Module?
- Monolithic and Micro Kernel
- Why do we need two bootloaders viz. primary and secondary?
- Given a pid, how will you distinguish if it is a process or a thread ?
- What is the difference between kill-6 and kill -9?
- What is memory leak?
- http://learnlinuxconcepts.blogspot.in/2014/03/what-is-bus-error-what-are-common.html
- What is a device tree in Linux?
- What is a linux kernel ? isit a process or thread?
- How function pointers are shared across different processes? using which IPCs?
- What are the Synchronization techniques used in Linux Kernel?
- On x86-32 Linux, at which address the code segment of the program and stack starts?
- Why Kernel Code running in interrupt

2. Interrupts- When the CPU receives some interrupt from the hardware then it jumps to the kernel mode and executes the interrupt handler and when the kernel is finished handing the interrupt the the code return to the user space where it was executing.

Posted by pk007 at 7:19 AM

g+1 Recommend this on Google

Labels: drivers, interrupts, kernel mode, Kernel space, linus kernel internals, LINUX, LINUX addressing, mapping, page fault, physical memory, segmentation fault, SIGSEGV, syscall, user space, virtual memory

1 comment:

SHIVAM MEHTA February 25, 2014 at 10:44 PM

Awesome yaar....keep writing man.....:)

Reply

Enter your comment...

Comment as: Google Accour ▾

Publish Preview

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

context cannot sleep?
- How does User Space memory Layout look like?
- How will the User Space mapping look like when our RAM is less than 896 MB?
- How will the Kernel Space mapping look like when our RAM is less than 896 MB?

Total Pageviews

11471

Is this blog useful to you?

□ Yes, very much

□ Ya, to some extent

□ It needs improvement

You may select multiple answers.

Vote Show results

Votes so far: 0
Days left to vote: 258

Follow by Email

Email address... Submit

Share It

Share this on Facebook

Tweet this

View stats

(NEW) Appointment gadget >>

Google+ Followers

Search This Blog

Search

Real Time Traffic

Live Traffic Feed

A visitor from India viewed "LINUX KERNEL INTERNALS: Kernel Space and User Space" 7 secs ago

A visitor from Seoul, Seoul-tukpyolsi viewed "LINUX KERNEL INTERNALS: Kernel Space and User Space" 38 mins ago

A visitor from Seoul, Seoul-tukpyolsi viewed "LINUX KERNEL INTERNALS" 1 hr 58 mins ago

A visitor from Pune, Maharashtra viewed "LINUX KERNEL INTERNALS" 2 hrs 21 mins ago

A visitor from Bangalore, Karnataka viewed "LINUX KERNEL INTERNALS: What is a Linux Device Driver Model A" 2 hrs 38 mins ago

A visitor from Seoul, Seoul-tukpyolsi viewed "LINUX KERNEL INTERNALS: How will the Kernel Space mapping look like when our RAM is less than 896 MB?" 2 hrs 38 mins ago

A visitor from Vijayawada, Andhra Pradesh viewed "LINUX KERNEL INTERNALS" 3 hrs 35 mins ago

A visitor from Bangalore, Karnataka viewed "LINUX KERNEL INTERNALS: Process Management" 4 hrs 6 mins ago

A visitor from Bangalore, Karnataka viewed "LINUX KERNEL INTERNALS: Explain the basics of Linux Kernel" 6 hrs 35 mins ago

A visitor from Seoul, Seoul-tukpyolsi viewed "LINUX KERNEL INTERNALS: Linux Addressing" 6 hrs 48 mins ago

Real-time view · Menu

g+1 2

Blog Archive

- ▼ 2014 (34)
  - ► March (23)
  - ▼ February (9)
    - Linux Kernel Questions
    - Interrupts
    - System Calls
    - Linux Memory Management
    - Android Boot Sequence
    - Kernel Space and User Space
    - Linux Addressing
    - STACK AND HEAP
    - Paging and Segmentation
  - ► January (2)

About Me

pk007

View my complete profile