

Re: Kernel stack

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-10/3194.html>

From: suthambhara nagaraj (suthambhara_at_gmail.com)

Date: 10/12/04

Date: Tue, 12 Oct 2004 12:21:37 +0530
To: "Dhiman, Gaurav" <gaurav.dhiman@ca.com>

Hi,

The problem is each process does not have a TSS of its own. Only one TSS per processor is present and the process dependant features (Like esp) are stored in another structure (struct thread_struct). A kernel stack of size 8k (By default) is actually shared by processes running on a processor. There is a func named load_tss (or something similar) which loads the TSS from the thread_struct structure during task switch .

A Process does not have an SS entry in its thread_struct but only an esp (and esp0) entry. This made me believe that the stack base is the same.

Correct me

Regards

On Tue, 12 Oct 2004 11:55:24 +0530, Dhiman, Gaurav <gaurav.dhiman@ca.com> wrote:

>
>> *I have not understood how the common kernel stack in the*
>> *init_thread_union(2.6 ,init_task_union in case of 2.4) works for all*
>> *the processes which run on the same processor*
>
> *As far as I know, Kernel do not have any common stack for all the*
> *processes running over it. Whenever we enter the kernel mode thru system*
> *calls, we go thru system gate or descriptor (0x80 entry) in IDT. This*
> *entry contains the index of the descriptor in GDT (normally it points to*
> *Kernel CS Segment Descriptor in GDT) and the offset (pointer) to the*
> *code to be executed in kernel mode (which is system_call() function in*
> *Kernel).*
>
> *Now the descriptor entry in GDT pointed out by the system gate entry in*
> *IDT, contains 2 bit field known as DPL (Desired Privelege Level). If*
> *this DPL is less than the CPL (Current Prevelege Level) of CPU then CPU*
> *switches to the process specific kernel stack segement by refferring the*
> *TSS of current running process. This stack switch is automatic by CPU and*
> *there is no assembly intruccion required for it.*

Linux-Kernel: Re: Kernel stack

>
> *This stack switch is done at the time when we enter from user space to
> the kernel space, this is done because we can not trust and share the
> user process stack (stack used by user process in user mode). That is
> why every process has atleast two and can even have four stacks. In each
> process, stack for every CPU level (ring level) is defined. So whenever
> the process runs in user mode (ring 3), its user mode stack is used, but
> when it enters the kernel mode (ring 0) its stack is switched to the
> kernel stack of that process. All the stacks of a process for different
> levels of CPU are tracked thru TSS defined for that process.*
>
> *To read more on IDT, GDT, TSS and System Calls invocation, refer to the
> Intels System Programmer's Guide. Her is the Link:
> <ftp://download.intel.com/design/PentiumII/manuals/24319202.pdf>*
>
> *Correct me if I am wrong somewhere.*
>
> *Cheers !!*
> *Gaurav*
>
>
>
>
>
> -----Original Message-----
> *From: kernelnewbies-bounce@nl.linux.org*
> *[mailto:kernelnewbies-bounce@nl.linux.org] On Behalf Of suthambhara*
> *nagaraj*
> *Sent: Tuesday, October 12, 2004 10:31 AM*
> *To: kernel*
> *Subject: Kernel stack*
>
> *Hi all,*
>
> *I have not understood how the common kernel stack in the*
> *init_thread_union(2.6 ,init_task_union in case of 2.4) works for all*
> *the processes which run on the same processor. The scheduling is round*
> *robin and yet the things on the stack (saved during SAVE_ALL) have to*
> *be maintained after a switch without them getting erased. I am*
> *familiar with only the i386 arch implementation.*
>
> *Please help*
>
> *regards,*
> *Suthambhara*
>
> --
> *Kernelnewbies: Help each other learn about the Linux kernel.*
> *Archive: <http://mail.nl.linux.org/kernelnewbies/>*
> *FAQ: <http://kernelnewbies.org/faq/>*
>
>

Re: Kernel stack

Linux-Kernel: Re: Kernel stack

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>