Issue with signal handling, interrupt handling

While the process is executing a blocking system call, say read or write, signal has arrived. Does the system call is terminated with the error EINTR? Does the system call is restarted after handling the system call?

Suppose, system call is terminated with the error EINTR, Kernel handles the signal before returing to user space.

Does the signal handle is executed in user mode/kernel mode? If its in user mode, does there'll be return to the instruction after the system call(read/write) during which signal arrived or again it goes to kernel mode after handling the signal and returns to the user from ret_from_syscall. How the execution is resumed at the instruction next to the system call during which signal arrived?

Is it possible to restart the system by passing SA_RESTART flag in sigaction?

linux linux-kernel

edited Apr 18 '13 at 9:23
Amit Tomar
1,861 12 37

asked Sep 3 '10 at 7:40
Ganesh Kundapur
210 2 11

1 Answer

Signal is executed in user *mode*, but with a different user *context*, then return to kernel, which return to user_mode with ret_from_syscall. The behaviour of system call when signal handler are installed with SA_RESTART depends on the system call.

A description of which system call are restarted is available in recent version of the signal overview manpage :

```
man 7 signal
```

If the SA_RESTART flag is not used, system call is not restarted.

answered Sep 3 '10 at 14:05
shodanex
7,639 5 28 64

1 Now i got what you mean by signal is executed in user mode with a different user context. In case of process receives a signal, process switches to kernel mode to handle the exception where in bitmask of the signal array of the current process is set. Right before returning to user mode, kernel checks the signal pending and calls the do_signal function to handle the signal which in turn calls handle_signal which copies the kernel hardware context, and modifies the user mode stack by invoking setup_frame. – Ganesh Kundapur Sep 21 '10 at 19:50

The new stack frame contains signum, PC value pointing to user mode signal handler, return address field containing the address of the system call sigreturn. On process returning to user mode, it starts executing the signal handler and on termination executes sigreturn which makes process to switch to kernel mode where in hardware context from the user mode stack is copied back to kernel stack by restore_sigcontext(restoring the user mode stack to original state). When the sigreturn system call terminates process switches back to user mode and continue from where it is left before the signal. – Ganesh Kundapur Sep 21 '10 at 19:54

It's important to notice that SA_RESTART does not always apply. Some interfaces are never restarted after being interrupted by a signal handler, regardless of the use of SA_RESTART; they always fail with the error EINTR when interrupted by a signal handler. Check Signal man page for details. – kikeenrique Sep 20 '13 at 8:29