

## Re: unsafe functions from signal handler

---

*Source:* <http://linux.derkeiler.com/Newsgroups/comp.os.linux.development.apps/2006-02/msg00052.html>

---

- *From:* "bill pursell" <[bill.pursell@xxxxxxxxx](mailto:bill.pursell@xxxxxxxxx)>
  - *Date:* 8 Feb 2006 09:31:54 -0800
- 

Kasper Dupont wrote:

bill pursell wrote:

I suspect it is because  
the address being written to is no longer being read from the variable,  
but is already in a register.

I suspect you are right about that.

How do I achieve the desired result?

You can siglongjmp out of the handler.

But if you don't even know, if writing the address is possible,  
how can you possibly know what damage could happen by writing to  
the address? Sounds like a design mistake.

It's not really a design at all! Just an academic question...I'm  
just trying to see how it's done. Is siglongjmp safe to call from  
the handler? It's not listed in signal(2) man page.

I'm not sure that this shouldn't start another thread, but it all  
ties into the theme of understanding the signal handling  
mechanism, so I'll keep it here....

I was having some issues today with the interaction  
of signals to a process blocked in wait(). It seems that  
wait doesn't behave as described in the man page:  
"The wait function suspends execution of the current  
process until ... a signal is delivered whose action is to ... call a  
signal handling function." But that doesn't seem to be the behavior  
I'm seeing. Process A forks and waits for child B. I attach to A via

## Re: unsafe functions from signal handler

gdb and see that it's in wait. I send A a SIGTERM. In gdb, I watch A step into the signal handler, step until the handler returns, and...find myself still sitting in wait(). Are there issues with wait I need to be aware of?