

Re: some PRs

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-07/0122.html>

From: Garance A Drosihn (*drosih_at_rpi.edu*)

Date: 07/27/04

Date: Tue, 27 Jul 2004 13:24:00 -0400
To: Sergey Babkin <babkin@bellatlantic.net>

At 12:36 PM -0400 7/27/04, Sergey Babkin wrote:

>Giorgos Keramidas wrote:

>>

> > On 2004-07-26 19:49, Sergey Babkin <babkin@bellatlantic.net> wrote:

> > >

>> > *I'm about a week behind :-)* but here are my 2 cents: it's a VERY
>> > *useful device for testing. Not checking the error code of write(),*
>> > *printf() and such is a typical bug, so making it easy to detect*
>> > *by switching the output to /dev/full (or creating a symlink to*
>> > *it) is a very good idea. Like this:*

>> >

>> > *yourprogram >/dev/full *

>> > *&& echo "The program does not check for success of write()"*

>>

> > *If a program doesn't check the return code of write() but merrily*
> > *goes on doing other stuff or even terminates with a zero return*
> > *value, how will the redirection affect its operation? I think it*
> > *won't, as shown in the test below (run on a Linux machine):*

>

>*If you run a test in which you know the program must fail (such*
>*as writing to /dev/full) yet it does not, this means that there*
>*is a bug in the program.*

> > *I must have misunderstood something. How do you mean that we*
> > *could use /dev/full for testing?*

>

>*Well, as described above: for each file that your program can*
>*produce, try to substitute it with /dev/full and watch the*
>*program fail. If it does not fail, there is a bug. That's*
>*much easier than producing an actual full filesystem.*

The problem is that using /dev/full on a good program will only test a single write() command. If I have a program which does writes at 10 different points, and the program fails when pointed at /dev/full, I still do not know if ALL of those writes are doing the appropriate checking. Maybe nine of them do zero checking, and only one of them checks (and that one may be the

freebsd-arch: Re: some PRs

final write(), not the first one). Using /dev/full on such a program will still get the failure indication, and the developer (or user) will think the program is correct, when in fact it is 9/10ths wrong.

The problem is that a full filesystem can happen at any point in the program's execution, not just the first write(). While /dev/full might be some help in testing, I do not believe it is as useful as you seem to think it is.

I am not against the idea of /dev/full, but I am not particularly for it either...

--

Garance Alistair Droseh = gad@gilead.netel.rpi.edu
Senior Systems Programmer or gad@freebsd.org
Rensselaer Polytechnic Institute or drosih@rpi.edu

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"