Re: has anyone installed 5.1 from a SCSI CD?

Source: http://unix.derkeiler.com/Mailing−Lists/FreeBSD/hackers/2003−09/0485.html

---

From: Terry Lambert (*tlambert2_at_mindspring.com*)
Date: 09/30/03

```
Date: Tue, 30 Sep 2003 01:05:12 −0700
To: Peter Jeremy <peterjeremy@optushome.com.au>
```

Peter Jeremy wrote:
> On Sun, Sep 28, 2003 at 06:14:25PM −0400, Sergey Babkin wrote:
> >BTW, I have another related issue too: since at least 4.7
> >all the disk device nodes have charcater device entries in /dev.
>
> 'block' vs 'character' has nothing to do with random or sequential
> access and any application that thinks it does is broken. Any
> application that directly accesses devices must understand all the
> various quirks − ability to seek, block size(s) supported, side−
> effects of operations etc.

As opposed to the kernel understanding them, and representing the classes of devices uniformly to the user level software.

> Yes, block devices must be random access,
> but character devices can be either random or sequential−only
> depending on the physical device.

But character devices can't be "random−only". Therefore, you can assume the ability to perform random access on block devices, and you can assume character devices require sequential access, and your software will "just work", without a lot of buffer copying around in user space.

> The only purpose for block devices was to provide a cache for disk
> devices. It makes far more sense for this caching to be tightly
> coupled into the filesystem code where the cache characteristics
> can be better controlled.

Actually, there are a number of other uses for this. The number one use is for devices whose physical block size is not an even power of two less than or equal to the page size. The primary place you see this is in reading audio tracks directly off CD's.

Another place this is useful is in the UDF file system that Apple was prepared to donate to the FreeBSD project several years ago. DVD's are recorded in two discrete areas, one of which is an

index section, recorded as ISO9660, and one of which is recorded as UDF. By providing two distinct devices to access the drive, it was possible to mount the character device as ISO9660, and then access the UDF data via the block device. Again, we are talking about physical block sizes of which the page size is not an even power of 2 multiple.

Another use for these devices is to avoid the need for some form of intermediary blocking program (e.g. "dd", etc.) for accessing historical archives on tape devices. Traditional blocking on tape devices is 20K, and by enforcing this at the block device layer, it's possible to deal with streaming of tape devices without adding an unacceptable CPU overhead.

Another issue is Linux emulation; Linux itself has only block devices, not character, and when things are the right size and alignment, the block devices "pass through" and act like character devices. However... this means that Linux software which depends on this behaviour will not run on FreeBSD under emulation.

Finally, block devices serve the function of double−buffering a device for unaligned and/or non−physical block sized accesses. The benefit to this is that you do not need to replicate code in *every single user space program* in order deal with buffering issues. There has historically been a lot of pain involved in maintaining disk utilities, and in porting new FS's to FreeBSD, as a result of the lack of block devices to deal with issues like this.

I'll agree that the change has been "mostly harmless" −− additional pain, rather than actually obstructing code from being written (except that Apple didn't donate the UDF code and it took years to reproduce it, of course, FreeBSD doesn't appear to have suffered anything other than a migration of FS developers to other platforms).

On the other hand, a lot of the promised benefits of this change never really materialized; for example, even though it's "more efficient" in theory, Linux performance still surpasses FreeBSD performance when it comes to raw device I/O (and Linux has only *block* devices). We still have to use a hack ("foot shooting") to allow us to edit disklabels, rather than using an ioctl() to retrive thm or rewrite them as necessary, etc., and thus use user space utilities to do the work that belongs below an abstract layer in the kernel.

I'm not saying that FreeBSD should switch to the Linux model −− though doing so would benefit Linux emulation, and, as Linux demonstrates, it does not have to mean a loss of performance −− but to paint it as something "everyone agreed upon" or even something "everyone has grown to accept" is also not a fair characterization.

−− Terry