

# Re: Linux community software–update–anarchy polemic

**Source:** <http://linux.derkeiler.com/Newsgroups/comp.os.linux.misc/2004-01/2201.html>

---

**From:** Anonymous Coward (*acoward\_at\_mail.ru*)

**Date:** 01/21/04

Date: 21 Jan 2004 04:28:35 -0800

Wow, you're willing to talk to a monkey!

ptb@oboe.it.uc3m.es (P.T. Breuer) wrote in message news:<e2ceub.5tr.ln@news.it.uc3m.es>...

> *GNU Parted – a partition manipulation program*

>

> *You are free to look it up :-).*

Thanks, I did. :)

> *Assuming that somebody else knows better is always a mistake.*

Have you personally read the source code of all software that you use?

Do you personally read all patches before building new versions of the software that you use?

I don't have the brains, time, or will to second-guess all authority.

Remember, I'm just a monkey. :)

> *One does not know of a bug until one discovers it. If you have found*

> *one, tell the author. I don't know of any bugs in parted (I know*

> *of some horrible misfeatures). The man page says:*

>

> **REPORTING BUGS**

> *Report bugs to <bug-parted@gnu.org>*

Yes, I know. parted itself reported that email address along with its

request that I submit a bug report. Why didn't I submit a bug report?

Why am I whining about it here instead? Read my original post.

> > *main point of my original post anyway. (Simply stamping the entire*

> > *distribution with "use at your own risk" is not the correct level of*

>

> *Of course it is. Your responsibility entirely.*

Sorry, "use at your own risk" there implies an issue of liability, which wasn't what I was trying to address. "Warning, all this software is perpetually beta, have a nice day" would be a better phrase.

Distributions currently categorized as beta or release depending on quantity/severity of known bugs, but there's never any zero-known-bugs version or a version with all known-buggy software quarantined or even easily distinguishable from the zero-known-bugs software.

[partitioning tool discussion snipped]

Ok, I'll take your word for it (not sarcastic), remember I'm just a monkey :) and partitioning tools aren't my primary concern anyway.

> > > *Choose your utility according to the match between its bugs and your bugs.*

>

> > *In general, without expending excessive time on research, I don't know what the bugs are, or whether they even exist. That's what I was*

>

> *Then nobody can help you, because bugs are ubiquitous.*

But they can be managed! That's my point.

> > *complaining about. A standard, universally used mechanism for marking, after release, released software as buggy could prevent problems such*

>

> *Buggy often is in the eyes of the beholder.*

Behaviors classified as buggy are often universally agreed to be buggy. Declaring complete surrender in the attempt to identify bugs just because the classification of some behaviors is disputed isn't a reasonable approach to take. The parted bug was really a bug, and nobody whom I'm aware of disputes that.

> > *as Knoppix shipping with a known–buggy parted 1.6.4, as I discussed in my original post.*

>

> *What's really buggy about it? Not working for you might not be considered a real big bug!*

It was a bug. And I'm not the only one who was bitten.

> > > > *In any case, it's not a good idea to have One Humungous*

> > > > *Partition, despite what the Micros\*\*t fans tell you.*

> > > > *Even if it's presently good practical advice to have multiple smaller*

> > > > *partitions, this is due to the way the software is currently written;*

> > >

> > > *Nonsense. Are you completely unaware of disks which break,*

> > *Having multiple disks addresses this problem.*

>

> *It doesn't, unless you have one disk per "partition".*

That implies that if you have one disk per partition, it does.

> *Which disk will*

> *be your readonly disk? Which disk will be your variable data high*

> *frequency change disk? Which disk will be your low frequency change*

> *disk? Which disk will bear your log files? Which disk will bear your*

> *mail spool? Which disk will be formatted with a jouralling file*

> *system? Which disk will be running sync? Which disk will be running*

> *noatime?*

You can have such distinctions delimited by boundaries other than partition boundaries, and therefore by boundaries other than disks boundaries in the case that you have one disk per partition, with

probably the practical exception of journaling and syncing, in which case the (again, practical) solution is simply to put a battery in the computer and obviate the need for journaling and syncing (at least in the case that the disk is local, with the RAM<–>disk connection being axiomatically reliable). Actually I considered mentioning in my previous post the battery option as another alternative to journaling, but chose to omit it for the sake of brevity. With a battery, journaling (and syncing) is an unnecessary waste of time unless the disk is remote (remote meaning that the RAM<–>disk connection is considered a potential point of failure, for example if it's a removable drive or a network–accessed drive).

> > *My comment was in the*  
> > *context of multiple partitions on a single disk. (As a side note, with*  
> > *regards to multiple disks, although presently popular filesystems do*  
> > *require multiple partitions if multiple disks are used, since those*  
> > *filesystems' partitions can't span disks,*  
>  
> *Partitions can span disks perfectly easily – this is called "logical*  
> *volume management", and linux has had it forever. The idea is that you*  
> *use a kernel logical layer above the real disks which presents a single*  
> *logical area to be split up into "partitions" which may themselves span*  
> *several real disks (but you don't know).*

Give me a break, I'm just a monkey.

> *The horror of this idea is that when you lose a real disk, you lose an*  
> *unknown part of your file system, possibly several file systems. It*  
> *makes your system more vulnerable, not less! And you never know just*  
> *where the thing was physically, so you try fixing it!*

I didn't actually say that one partition spanning multiple disks could be just as reliable as a partition contained entirely on a single disk.

> *Eccccccch. Arrrrrrrrgh..*  
You sound like an unfed monkey.

> *I \*hate\* lvm.*  
Hmm, I suppose I do too.

> *A broken lvm is a nightmare*  
> *Several of them simultaneously.*  
Sure sounds like it.

> > *this also is due merely to*  
> > *the present design of the filesystems, and isn't a fundamental rule.*  
>  
> *That's lack of experience talking and sounding big! :-).*

Well I was obviously right that it isn't a fundamental rule; lvm implements partitions spanning disks. :)

> > *The kernel can hose anything it wants, and partition barriers can't*

>

> *But it doesn't.*

[snip]

> *You might, if you*

> *were lucky, get a bit changed in the partition table image in memory,*

> *so that the partition offset was registered wrong. THAT would produce*

> *the effect you want.*

>

> *But that bit would have to be flipped quite precisely. It's a difficult*

> *shot – it's much more likely that the cosmic ray will flip a bit in*

> *some applicatiion instead!*

Trust the kernel or don't trust the kernel, but either way, both partition and filesystem boundaries are just bits on the disk, and only become barriers when the kernel interprets them (as you agree below). The kernel that enforces partition barriers is the same one that enforces filesystem barriers.

> > *Outside the kernel, intrapartition barriers (filesystem*

> > *security attributes) are the standard means of protection.*

>

> *I have no idea .. are you suggesting making a directory hierarchy*

> *readonly?*

Of course you mean "making the disk blocks referenced by a directory hierarchy readonly", as the hierarchy itself can of course be made readonly on present systems (chmod –R 444 dirname). In order to prevent shuffling of the blocks on disk, if that's actually what you need... ok I'm just a monkey, you need a separate partition. But you only need to prevent shuffling of blocks on disk when you don't have a backup power source.

> *How would you then mark it noatime*

Per directory.

> *and sync? Or journalled?*

Don't. Buy a battery.

> *The reason some things are partition rather than file attributes is*

> *because it is unreasonable to engineer it any other way!*

Block–level readonly, you're right. Buy a battery. Sync and journaled, you're right. Buy a battery.

> *It would mean*

> *coalescing the ideas of file system and file to a much greater extent.*

In other words: no–can–do with present software. I didn't claim otherwise.

> *The result would be phenomenally inefficient. Imagine having to look up*

> *the target of every write before you could decide if you should journal*

> *it or not!*

Ok ok I'm just a monkey, enough with the journaling example already,

buy a battery!

> *How would you deal with races between mkdir and write? Etc.*  
Having multiple partitions is the solution to this problem? That's news to me.

> > > *users who have no social responsibility,*

>

> > *Again, intra–, not inter–partition barriers address this.*

>

> *I'm afraid it's more subtle. If /tmp is on the same file system as root,*  
> *then even if every user is quotaed, they can still get together and*  
> *together fill /tmp, thus filling the root file system. If /tmp*  
> *(/var/tmp) is on the same file system as /var/spool, then they can stop*  
> *mail by the same mechanism. And if they are so quotaed they can't even*  
> *do it as a group, they can mail each other stuff continuously just to*  
> *stop mail, AND to fill /var, maybe stopping logging. Etc etc etc.*  
That's because you're effectively quotaing the size of /tmp by putting it in a separate partition. Add the ability to quota directories, not just quota users, and the necessity of putting /tmp in a separate partition goes away.

> > > *and the*

> > > *myriad of other damages that make partitioning a simple matter of*

> > > *common sense?*

> > *Examples? If you're referring to filesystem corruption due to kernel*

>

> *There are generally two or three random corruptions on the file system*  
> *every day in any set of 20 or so machines. If that system is the root*  
> *file system, either the machine breaks and throws a hissy fit at next*  
> *fsck, possibly deciding to clear the root inode, or the corruption goes*  
> *unnoticed until things get even more interesting, or it goes blooie*  
> *with "odd effects" immediately.*

ECC your memory, your disk's interface cable, the disk itself, your system bus, the processor's memory cache, etc. Yes I know the disk already uses ECC, but the manufacturers' specs of around  $1/10^{13}$  nonrecoverable error rate, although impressively low, is still too high; sacrifice a little disk capacity and increase reliability. To the best of my knowledge the processor's L1 cache typically has ECC, haven't really paid any attention to other parts of the processor or other parts of the system. But if data integrity matters, don't have non-ECC memory or datapaths anywhere in your system.

> *I have learned by long experience to keep the root files system small*  
> *and duplicate it elsewhere on disk. That way you can always boot*  
> *and log in easily when /var/run and /var/adm decide to disappear,*  
> *thanks to gamma ray absorption, because the error will not be on the*  
> *root file system.*

First, compensate for gamma ray absorption bit errors at the bit level with ECC. But I didn't say that backups were bad. But if your backups are on the same disk, they're subject to being hosed by a

bit–error–induced kernel psychosis regardless of whether they're in some other partition.

- > *I have also learned through experience to keep /usr readonly. Things*
- > *get so much less corrupted when there is nothing being written anywhere*
- > *near them on the disk.*

Nearness is a meaningless concept to a psychotic kernel.

- > *And then there are the bouncing screaming disk head crashes, that tear*
- > *of strips at the same area every cylinder for about 60MB worth (says he*
- > *remembering the last time he had to quarantine a disk zone, by hand,*
- > *after recovering what he could through 8 hours of manually attended*
- > *fsck, with the heads painfully retrying every unreadable sector 10*
- > *times while I noted down the sector number for later reference).*

Use backups.

- > > *malfunction, remember that the kernel can hose anything it wants on*
- >
- > *It can, but it doesn't. I've explained why.*

The best I understood your explanation was that the kernel has a lower probability than an application of deciding to hose things. Agreed.

But when the kernel does decide to hose something, it can hose anything it wants.

- >
- > > *any partition. If you're referring to filesystem corruption due to*
- > > *power loss, partition barriers can't provide any protection (or a*
- >
- > *They do provide it. For one thing, if you are not writing there, then a*
- > *missed write cannot be a problem. Once written – always readable.*
- > *Read–only partitions are a godsend. Aaaaaaaah. Bliss.*

You split my sentence. The clause "partition barriers can't provide any protection" wasn't absolute, it was qualified by a "that can't be provided by..." clause (below) which your response here doesn't account for.

- > > *level of performance at a given level of protection for that matter)*
- > > *that can't be provided by a single partition with a filesystem which*
- > > *uses sufficiently deterministic write ordering (e.g. a journaling*
- > > *filesystem).*
- >
- > *Journalled filesystems are among the most fragile in practice to*
- > *hardware (not software) corruption. I have learned never to journal*
- > */var, because the high i/o at the journal seems to wear a hole in the*
- > *disk under it! I have learned to always have the journal external to*
- > *the partition so that when it breaks, I can move it to another 8MB spot*
- > *on the disk and nobody is hurt. You try quarrantining an 8MB unwritable*
- > *unreadable hole in the middle of a partition!*

Flash disk systems seem to handle write–wearing problems by shuffling physical write areas without needing multiple partitions.

> *And then there is the issue of creeping corruption, which goes  
> unchecked.*

Read-only partitions can lull you into a false sense of security; you can't avoid regularly fscking them too, because a psychotic kernel can hose them nonobviously just like it can hose everything else nonobviously.

> > *Redundancy and ECC codes protect against small random errors.*  
>  
> *But they don't – ecc memory is not generally used and is expensive  
> (therefore not bought).*

And you're the one who's worried about creeping corruption?  
Besides, ECC memory is just memory with with different addressing dimensions (a wider than usual word size); if you have to buy so much memory that cost is an issue, buy standard memory instead and reconfigure the ECC controller to use every ninth word (or every ninth page or whatever would be most efficient with current DRAM addressing modes, I really haven't paid attention to the state of the art) for ECC bits, which simply results in a 1/9 decrease in memory capacity.

> *Redundancy requires a separate partition.*  
Please think about what you just said.

> *And then you get the issue of copying and replicating the errors, if  
> you are talking about raid!*

Yes, if you hose you data and then copy it, the copy will be hosed too. So make sure your data isn't hosed before you copy it. ECC is your friend.

> > *Inter-partition redundancy provides no protection that can't be  
> > provided intra-partition.*  
>  
> *Oh yes it does. Try turning of metadata read time updates per  
> directory.*

Ok.

> *Yes – even a readonly directory on a stanadard writable file  
> system will have the time it was last read continuously written into it.*  
Set noatime per directory.

> *And that's only one example. See above for more and more and more ..*  
See above.

> > *See above.*  
>  
> *You "see above"!*  
See above. :)

> > > *maladminsitration, human error,  
> > As root, rm -rf/ can cross partition barriers.*  
>

> *Nope. Partitions are mounted readonly, so the rm will fail at the  
> boundary.*

Ok, play with mount before you play with rm. Or play with fdisk by itself.

> > *and a host of other things that happen  
> > with probability one over a surprisingly short interval of time.*

>

> > *More examples?*

>

> *Don't you have enough already? People adding service and forgetting to  
> add a stanza to logrotate their logs every day, so that the log fills  
> up the partition after a week, and one thousand users start  
> telephoning your office to say they can't log into X windows. Umm ...  
> warez trying to fill ~ftp/upload to bursting, umm ... the monitor  
> software running unattended for two years as root finally gets to the  
> point at which its history files are so large that the real users on the  
> system have no room to do anything, and start complaining that they  
> are "losing mail" (yo! well read the message that says "mail file  
> corrupted, mail not written, do not quit editor until saved" then).*

Directory quotas (as opposed to user quotas) could fix this; see my response to your example above about /tmp.

> *Etc. etc. etc. etc. etc. etc. etc. etc.*

More examples?

> > *It's relevant to us [a]ssessing your expertise,  
> > True, but this is irrelevant to the issues which I reported.*

>

> *It is. What to you seems a big deal may not be a big deal to someone  
> else. I pointed out that you can use something else to make partitions  
> (I'd never trust any installer to do such a thing for me!). You may say  
> that you are just acting like a dumb monkey and reporting what things  
> look like to a dumb monkey. And I'd say that's true. Yes – it's great  
> that you can speak up about it. Most dumb monkeys can't. That's a real  
> service you are performing, and I mean it! It's useful to know what  
> things look like to a dumb monkey.*

Even people who are capable of being more than dumb monkeys often don't want to spend the time or effort necessary to operate at more than monkey level on systems with which they're not already familiar (and sometimes even on systems with which they are). People use each other's software and rely on community consensus about its reliability, security, etc without even bothering to examine the code themselves, rather than always writing their own software or even reading the source they download before compiling, and there's a reason for this. The makers of apt–get were competent to manage their packages without it, yet they made it, and they use it themselves; they didn't just write it as a crutch for dumb monkeys. Etc.

> > *False. If you were to discover one day that your random number  
> > generator had just output a polynomial–time factoring algorithm, would*

> > *you ignore the algorithm due to the source?*

>

> *I would ignore the report, especially if the source proved in its  
> introduction or discussion that it were not cognizant of the issues, nor  
> of other peoples perceptions of them. Thanks – but I receive enough  
> such missives as it is. I can distinguish "crank" perfectly easily.  
> A random number generator would provide no convincing argument at  
> all, and hence would be ignored directly.*

Notice that I said "If you were to discover one day that your random number generator had just output...", not "If your random number generator had just output...". Meaning that for some reason, you were actually sitting around bored one day, reading /dev/random, and you discovered a polynomial–time factoring algorithm in /dev/random's output.

> > *Your sense of my expertise*

> > *can reasonably determine whether you bother to read what I write, but*

> > *after you read what I write, my expertise and your sense of it become*

> > *irrelevant, with only that which was written remaining relevant.*

>

> *This is not quite true.*

It is completely true! Reread what I wrote here.

> *Issues are being discussed, and you must show*

> *that you understand the background to them, and other peoples*

> *arguments, before asserting your own opinions and arguments.*

I didn't say otherwise.

> > *Anyway, please remember that my comment was only that a single*

> > *partition can be just as reliable as multiple ones if the software is*

> > *so written, not that contemporary software actually is so written. And*

>

> *Unfortunately, what you mean is "if all software is so written", not "if*

> *software is so written". Partitions are only a virtuality and it is*

> *precisely software (in the form of the kernel) that forces their*

> *observance. Indeed, a "partition" is precisely a region of the disk*

> *that the kernel treats in a particular uniform way, imposing the*

> *boundaries on applications.*

Which places partition barriers in the same category as filesystem barriers, as discussed above.

> *So what you say is tautologous if by*

> *"software" you include the kernel.*

So you're finally acknowledging the equivalence of partition and filesystem barriers?

> *But if you do not include it, then*

> *it becomes false, because "ALL" software must then be written to observe*

> *the boundaries that the kernel no longer imposes.*

I think we can agree to drop this case, as I am including the kernel.

> > *please remember that it's completely irrelevant to the issues raised*

> > *in my original post, issues which no one has yet addressed.*

>

> *Because I don't think they deserve the name "issues". You had a problem*

> *with parted? Well, tell the author. That's the end of it.*

There's no need for me to write to the author. I explained why in my original post. Please read it.

> *If you want to raise an issue, an appropriate one might be how one can*

> *deliver feedback to authors automatically,*

Agreed. And that's the other half of the issue I'm talking about.

> *and spread an awareness of*

> *the development state of the software to potential users.*

That's exactly the issue I'm talking about. (Users being not only end users, but other programmers and distribution makers too).