Re: usb2 port gets very slow on 2−gig Flash Drive.

Source: http://linux.derkeiler.com/Mailing−Lists/Debian/2008−07/msg02280.html

- *From*: "elijah r." <elijahr@xxxxxxxxx>
- *Date*: Thu, 24 Jul 2008 09:21:35 −0500

Correction:

mount −o loop −t msds zenstone2g.img /mnt/zenstoneimg
should read:
mount −o loop −t msdos zenstone2g.img /mnt/zenstoneimg

Material Safety Data Sheets do not play a part here. :−) You might
also try −t vfat, one of them should work.

On Thu, Jul 24, 2008 at 9:20 AM, elijah r. <elijahr@xxxxxxxxx> wrote:

> Say your flash disk is /dev/sdc and has 1 FAT partition where the
> music is stored:
>
> cd ~
> dd if=/dev/sdc1 of=zenstone2g.img bs=64M
> mkdir /mnt/zenstoneimg
> mount −o loop −t msds zenstone2g.img /mnt/zenstoneimg
> −−−delete anything in /mnt/zenstoneimg/music you don't want −−−−−
> cp −R /path/to/my/music/* /mnt/zenstoneimg/music/
> umount /mnt/zenstoneimg
> dd if=zenstone2g.img of=/dev/sdc1 bs=64M
>
> You'd only need to run the commands before "mount" once, then keep the
> image around for the future.
> I don't actually know if this will speed things up, but the idea is
> that you are bypassing the filesystem layer over the USB bottleneck
> for disk writes.
> You make all the filesystem changes to a disk image on your hard disk,
> then you stream the bits in that image to the flash disk.
> The "bs=64M" is the block size and can be increased or decreased
> depending on your situation. AFAIK, it basically acts like a buffer
> in situations like this.
>
> Be very careful with the syntax of dd; one misplaced character and you
> could wipe your hard disk or flash drive.
>
> Just an idea. Let me know if it works out.

−Elijah

On Thu, Jul 24, 2008 at 8:27 AM, Martin McCormick
<martin@xxxxxxxxxxxxxxxxxx> wrote:

> I needed to reload the flash drive on a Zenstone 2−gig mp3
> player. I have noticed that when the drive is full, operations
> all still work, but take much longer than one might expect which
> probably has something to do with the fat32 file system and size
> of the FAT table.
>
> I started by doing rm −r −f music which is the one and
> only main directory. That took about 2 hours to complete and
> then it was time to reload it from the Linux computer.
>
> The Linux computer is a 500−MHZ Pentium running a 2.6.5
> kernel and the usb2 port is, by definition 13 Mb so it is slow,
> about 10−meg Ethernet slow under normal conditions.
>
> What actually happens is much worse. After doing the rm
> −r −f, I started with an empty drive but the tar program started
> running very slowly. It takes at times, 10 minutes to load one
> tune that would normally play in 2 or 3 minutes.
>
> Just for fun, I let it plod slowly along and it has been
> running now for about 36 hours and is 75% complete but what is
> happening?
>
> If I look at the health of the system, it isn't that bad
> all be it busy.
>
> $ uptime
>
> 07:21:58 up 2 days, 23:37, 2 users, load average: 4.00, 4.13, 4.19
>
> It is July 24 and ps ax −Olstart shows me that tar has
> been running a very long time:
>
> 22986 Tue Jul 22 23:35:01 2008 D pts/13 00:00:40 tar xf
> /home/martin/musicmain.tar
>
> I tried renic−ing tar and the usb mass storage daemon
> which has had no effect, either good or bad.
>
> I do remember once that if there is a time lapse of
> several minutes between the rm −r −f command to wipe the FAT
> table clean and the tar command that initial performance is much
> better so there must be a cleanup routine going on somewhere.
>
> This is also true if you unmount the drive and remount it empty.
> I seem to remember that it only took a couple of hours to reload

the entire drive when I did things that way.

I did a top command and I am either missing something or nothing much is wrong:

Tasks: 42 total, 2 running, 40 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.3% us, 0.0% sy, 0.0% ni, 0.0% id, 99.7% wa, 0.0% hi, 0.0% si
Mem: 126424k total, 124572k used, 1852k free, 1604k buffers
Swap: 377488k total, 0k used, 377488k free, 102308k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
23560 martin 16 0 2160 1084 1880 R 0.3 0.9 0:00.03 top
1 root 16 0 1492 528 1336 S 0.0 0.4 0:07.26 init
2 root 34 19 0 0 0 S 0.0 0.0 0:12.13 ksoftirqd/0
3 root 5 −10 0 0 0 S 0.0 0.0 0:00.00 events/0
4 root 5 −10 0 0 0 S 0.0 0.0 0:04.02 kblockd/0
10 root 7 −10 0 0 0 S 0.0 0.0 0:00.00 aio/0
9 root 15 0 0 0 0 S 0.0 0.0 0:22.06 kswapd0
11 root 25 0 0 0 0 S 0.0 0.0 0:00.00 jfsIO

I plan to let this tar command run its natural course to see what happens, but is there anything I can do with the existing system to optimise the performance?

Thanks.

Martin McCormick WB5AGZ Stillwater, OK
Systems Engineer
OSU Information Technology Department Network Operations Group

−−
http://elijahr.blogspot.com/

−−
To UNSUBSCRIBE, email to debian−user−REQUEST@xxxxxxxxxxxxxxxx
with a subject of "unsubscribe". Trouble? Contact listmaster@xxxxxxxxxxxxxxxx