Old Confobd

Nathaniel Rutman

6/17/05

1 confobd on b_cray

Briefly recorded here for historical purposes.

The main reason confobd was implemented was to make running Lustre as a rootfs easier (see bz5425).

Currently confobd is a very simple obd layer that pretty much does only two things: write logs, and parse (execute) logs. lconf on b_cray was changed to use the confobd to write some very simple log files that do some of what lconf used to do directly. For example, this is the entire ost1-conf file:

```
1:LUSTRE dev:ost1
attach obdfilter ost1 ost1_UUID
2:LUSTRE dev:ost1
setup /dev/loop/0 ext3 f errors=remount-ro,asyncdel
```

There's also a similarly brief OSS-conf, mds1-conf, and MDT-conf file (actually, the mds1-conf used to also have the MDT setup in it; splitting it into two is what fixes 5287). So by putting these instructions into log files, lconf no longer had to figure this out at every run, it could just have confobd execute the log files.

Note that this had no effect on how existing logs were handled. The mds log "mds1" is written by lconf and read directly by the MDS. The client logs are written by lconf and read via a temporary MDC on the client connecting to the MDS. I'll call these "setup logs" as opposed to the confobd's "config logs".

Now enter wrinkle #1: In order for confobd to write (and read) the config logs, it needs a mounted disk. A decision was made to make that mounted disk the same as the backing device (MDT or OST), and store the log files in the same /LOGS directory as the setup logs. Two related factors result from this decision: a. the mount call must be replaced by a mount refcount, since the confobd and the mds are both using the same device, and b. the mount options must be identical (or rather, only the first mount's options (in this case confobd) are used). (b) requires that lconf now start the confobd itself with the correct mount options. A third factor resulting from this decision is that there must be one confobd instance per mds or ost.

Wrinkle #2: By writing device info (that used to be dynamically generated by lconf) into the -conf logs, we lose the portability that lconf had. For example, a loopback fs can be assigned any loop device by lconf, but the log file has hard-coded a particular loop device.