



Silicon Graphics, Inc.

XFS Overview & Internals

10 - QA

November 2006

QA

- xfs-cmds/xfstests
 - See the README file in the xfstests directory for more information.
- Uses golden output pass/fail rule.
- Uses group file to contain test grouping info.
- Has ~135 tests and growing. Tests are numbered.
- Can be run over NFS and on UDF filesystems.
- Runs on IRIX and Linux.

QA Approach

- XFSQA is an aggressive test suite that:
 - Mounts/Unmounts
 - mkfses
 - If running a debug kernel will almost always oops a machine.
 - tries to corrupt the filesystem.

QA Test Results

- The test suite will cause the following failures:
 - Not Run: Something the test needed is missing.
 - Fail: Golden output mismatch.
 - Filesystem inconsistency: Run screaming.
 - Test/machine hung.
 - Machine oops.

Building QA Tools

- Linux systems require the following rpms to be installed.
 - gettext-devel libattr-devel libacl-devel e2fsprogs-devel quota indent lkcdutils
- If testing against a product kernel and user-space, you will also need to install
 - xfsprogs-devel xfsdump
- If testing against your own source, you will need to install the user space commands and headers:
 - > cd xfs-cmds
 - > make
- You can then install the rpms that are built or run:
 - > pth=\$PWD
 - > for pkg in xfsprogs dmapi xfsdump xfstests; do
 - > cd \$pth/\$pkg
 - > sudo make install
 - > sudo make install-dev
 - > done
- You can now build the tests:
 - > cd xfs-cmds/xfstests
 - > make

QA Crash Dumps

- Ensure the correct setup of lkcd (linux kernel core dump):

```
# vi /etc/sysconfig/dump
DUMPDEV=/dev/vmdump
DUMP_FLAGS=0x80000000
# chkconfig --level 35 boot.lkcd on
# /etc/init.d/boot.lkcd start
```

- In the event of a kernel panic or oops:

```
kdb# sr d
```

- The crash dump will be saved at /var/log/dump/

QA Configuration

- The host machine configuration is defined in a switch in the `common.config` file in the `xfstests` directory.
 - Edit this file and if you will be regularly using the machine for QA check the change in.
 - You can also define a `local.config` file.
- XFSQA needs at least 2 xfs partitions, refereed to as scratch and test, preferably less than 20GB
 - The test partition is a persistent partition (survives multiple runs).
 - Scratch will be mkfs'd several times each run.
- Minimum disk setup:
 - Scratch device (non persistent). `$SCRATCH_DEV`
 - Test device (persistent). `$TEST_DEV`
- Can also have external log devices, realtime and tape devices defined:
 - remote log devices. `$TEST_LOGDEV`, `$SCRATCH_LOGDEV` plus export `USE_EXTERNAL=yes`
 - realtime device `$SCRATCH_RTDEV`
 - tape device. `$TAPE_DEV`
 - remote tape device. `$RMT_TAPE_DEV`
 - remote IRIX tape device. `$RMT_IRIXTAPE_DEV`

Running QA

- Once the QA machine is setup you can run the test suite.
- Use the check command to run the tests, the most common options are:
 - -r randomize test order
 - -l log results
 - -g *group* runs the tests in *group*.
- For example:

```
# ./check -r -l -g auto
```
- There are other options see the README file for details.

Other QA tools for the kernel

- Tools:
 - KDB
 - blktrace, strace, ltrace
 - sparse
 - lockdep
 - stackcheck
 - Generic kernel debug options
 - SLAB, PAGE_ALLOC, SPINLOCKS, MUTEXES
 - PREEMPT
 - XFS debug (assert, runtime checks)
 - XFS tracing option (event history)
 - XFS error injection option (xfs_io “inject”, fsstress)

QA Large Filesystems

- We will always lack hardware to test the high end of filesystem size, inode count, etc.

- Can use large sparse files for this

- led to allocation group size changes

```
# mkfs -dfile,name=/tmp/x,size=1p)
```

```
# mount -o loop /tmp/x /mnt/x
```

- `xfstests/tools/ag-wipe`
 - uses `xfs_db` to mark some AG's as “full”
 - integrated in QA (check/repair issues)
 - `xfs_repair` undergoing surgery
 - maybe done all we can with this approach

sgi®