Enable file system journaling on Solaris 8

Quick and Dirty: Turning on journaling feature for UFS

To avoid lengthy fsck after an unclean shutdown or power outtage, one can turn on journaling on UFS simply by adding "logging" in the mount option:

```
# cat /etc/vfstab

#
#device         device          mount   FS    fsck  mount    mount
#to mount       to fsck         point   type  pass  at boot  options
#
/dev/md/dsk/d1 /dev/md/rdsk/d1 /RAID   ufs   2     yes      logging
```

NOTE: There were some early bugs with UFS logging, so be sure to have the latest kernel patch cluster installed on your systems before enabling logging.

Enable file system journaling on Solaris 8

Solaris 8 include a native implementation of file system journaling. This feature, known as "intent logging" or just "logging" enables FASTER file system operation and FASTER system boot.

It's trivial to implement and safe to use. The new logging feature is an option to the Unix File System (UFS), which is the standard file system for all disk partitions on SUN servers, except for partitions holding swap space. By default, the journaling option is disabled. Logging is enabled on a per file system basis, and it can even be enabled on / (root file system) and other operating system partitions.

Background

Solaris UFS logging works by allocating space from the file system's free blocks. Within that space, all metadata changes to the file system are written. Metadata includes directory and I-node information but not file data blocks, essentially everything but the actual data within the file. So, for example, a "file create" modifies the directory structure and allocates a new I-node, and those activities are written to the logging space. Once the metadata changes are made to the logging area, the system is free to perform other operations to the file system. In the background, the information in the log is flushed to the file system and updates the appropriate directory and I-node structures, completing the file system operations.

The logging data is written sequentially within the log space. It's therefore much faster for the operating system to complete metadata changes via logging and background flushing than by

directly modifying the metadata (via random I/O) spread across the disk. The size of the logging space is based on the size of the file system, and equals 1 MB per 1 GB of file system space, up to 64 MB. The space is used as a circular log: if the log space is about to fill up, new metadata change requests are paused while the log is emptied. As changes are moved from the log to the file system, that log space is made available, and new metadata changes can be written to the logging space.

Usually with UFS, if the system crashes during any file system operation, the entire system must have its consistency checked via the fsck command. That command can take several minutes per file system because it checks all metadata and file data to ensure the structures are correct, free, and used, and that the I-node block counts are correct. It also confirms that the free space available is current, repairs inconsistencies, and occasionally requires manual intervention to fix large problems. Files and even directories can be lost, depending on the operations occurring at the time of the crash.

Because metadata changes are made first to the log space rather than to the file system, the consistency check for a logged file system after a crash is a simple and fast operation. The system evaluates the logging data and determines which changes had completed against the underlying file system, which had yet to start, and which were in progress. Those completed or not yet started are removed from the log, and those partly completed are either undone or completed. If there's sufficient data in the log to complete the operation, it's completed. Otherwise, the changes made are removed from the underlying file system.

People familiar with database operation will recognize the similarity between database transaction processing and the activities here. The end result is that the underlying file system is consistent, and no thorough consistency checking is needed. That operation completes in a few seconds per file system.

Using logging

There's a new logging option to the mount command and in the `/etc/vfstab` system configuration file. Logging only appears in a couple other places within Solaris. The mount command shows which partitions are mounted and lists logging in the options fields for each partition on which logging is enabled. Finally, at system boot time, the `fsck` phase reports per partition whether each is stable, logging, or being checked. There are no other status commands available to determine the state of logging.

Column	Description
A	Device to mount
B	Device to fsck
C	Mount point
D	Filesystem Type
E	Fsck pass (unimportatnt with logging)
F	Mount at boot
G	Mount options

```
# cat /etc/vfstab

# ----------------------------------------------------------------------
# A                 B                     C       D     E   F   G
# ----------------------------------------------------------------------
fd                  -                     /dev/fd fd    -   no  -
/proc               -                     /proc   proc  -   no  -
/dev/dsk/c0t0d0s3   -                     -       swap  -   no  -
/dev/dsk/c0t0d0s0   /dev/rdsk/c0t0d0s0    /       ufs   1   no  logging
/dev/dsk/c0t0d0s6   /dev/rdsk/c0t0d0s6    /usr    ufs   2   no  logging
/dev/dsk/c0t0d0s1   /dev/rdsk/c0t0d0s1    /var    ufs   3   no  logging
/dev/dsk/c0t0d0s7   /dev/rdsk/c0t0d0s7    /home   ufs   4   yes logging
/dev/dsk/c0t0d0s5   /dev/rdsk/c0t0d0s5    /opt    ufs   5   yes logging
/dev/dsk/c0t8d0s0   /dev/rdsk/c0t8d0s0    /u01    ufs   6   yes logging
/dev/dsk/c0t9d0s0   /dev/rdsk/c0t9d0s0    /u02    ufs   7   yes logging
/dev/dsk/c0t10d0s0 /dev/rdsk/c0t10d0s0 /u03    ufs   8   yes logging
/dev/dsk/c0t11d0s0 /dev/rdsk/c0t11d0s0 /u04    ufs   9   yes logging
/dev/dsk/c0t12d0s0 /dev/rdsk/c0t12d0s0 /u05    ufs   10  yes logging
/dev/dsk/c1t13d0s0 /dev/rdsk/c1t13d0s0 /app    ufs   11  yes logging
/dev/dsk/c1t14d0s0 /dev/rdsk/c1t14d0s0 /users  ufs   12  yes logging
swap                  -                  /tmp    tmpfs  -  yes -
```

There were some early bugs with UFS logging, so be sure to have the latest kernel patch cluster installed on your systems before enabling logging.

Logging increases performance, decreases fsck time, removes the risk of a file system corruption, can be used on all UFS partitions (including root), and is free.

Another mount option - noatime

There's another relatively new and useful option you should be aware of. It's the `noatime` mount option.

Without `noatime`, every time a file is opened for read, its access time i-node value is

updated. That is useful for listing the date and time the file was last read, as displayed with the `ls -u` command.

There are many instances in which either the last access time is not interesting, keeping it up to date is too much overhead, or both. Web server contents (static text pages, for instance) and Usenet news directories are two good examples. In those instances, the overhead of performing one i-node write for every file open is quite heavy.

The noatime mount option decreases the frequency of access time update. Essentially, it tells the system to only update the access time if another update to the i-node is being done coincidently. No harm is done, especially on file systems where there's no interest in the last access time information.