NFS performance tuning with HP StorageWorks Enterprise File Services (EFS) Clustered File System white paper

This paper applies to the HP StorageWorks Enterprise File Services (EFS) Clustered File System (CFS) on the SLES9 Linux operating system.

Summary

NFS file system performance using the default settings can be quite low. This white paper describes how to tune NFS performance to an optimal level.

Most of the information in this white paper is taken directly from the sections "Optimizing NFS Performance" and "Mount Options" in the "nfs-howto" document located at http://nfs.sourceforge.net/nfs-howto/.

For additional information on NFS performance tuning, visit: http://www.tldp.org/HOWTO/NFS-HOWTO/performance.html

Soft versus hard mounts

Soft and hard mounts can be characterized by how they handle errors.

	Hard	**Soft**
How it works	When an NFS client with a hard-mounted share requests a file, it keeps trying until it succeeds or someone interrupts its attempts. You can specify whether you want users to be able to interrupt an NFS request with the `intr` and `nointr` options.	When an NFS client with a soft-mounted share requests a file, it tries to receive the file a specified number of times (`retrans`); each try waits for a specified amount of time (`timeo`). After this, if the file has not been received, NFS will return an I/O error to the process on the client machine requesting the file.
Important parameters	`intr:` Specify `intr` if users are not likely to damage critical data by manually interrupting an NFS request. If a hard mount is interruptible, a user may press **CTRL-C** or issue the kill command to interrupt an NFS mount that is hanging indefinitely because a server is down. `nointr:` Specify `nointr` if users might damage critical data by manually interrupting an NFS request and you would rather have the system hang while the server is down than risk losing data between the client and the server.	`timeo:` Timeout in tenths of a second for NFS requests. The max value for `timeo` is 30 (3 seconds). Default value is 7 = 0.7 seconds. Try doubling the `timeo` value if you see several "server not responding" messages within a few minutes. This can happen because you are mounting directories across a gateway, your server is slow, or your network is busy with heavy traffic. `retrans:` The number of times an NFS request is retransmitted after it times out. If the request does not succeed after the specified number of retransmissions, an error is returned. Default value is 4. Increase the `retrans` value for a directory that is soft-mounted from a server that has frequent, short periods of downtime. This gives the server sufficient time to recover, so the soft mount does not return an error.
Sample /etc/fstab	10.0.0.1:/share /mnt/share nfs hard,intr 0 0	10.0.0.1:/share /mnt/share nfs soft,timeo=7,retrans=4 0 0

Source: Configuring and Administering an NFS Client, 2004–2005, 6/21/05, http://docs.hp.com/en/B1031-90043/ch02s03.html.

If you are receiving I/O errors with a soft mount, you might want to consider switching to a hard mount or raising your `timeo` and/or `retrans` parameters to compensate. Consider that the maximum acceptable time delay for an nfs mount to respond before receiving an I/O error is (`retrans*timeo`), so for the preceding example 4*0.7=2.8 seconds.

Other changes that can improve NFS performance include:

- Use Gigabit Ethernet.
- Change the read and write size to 32 K.
- Using a blocksize of 32 K causes a great deal of fragmentation and can actually reduce performance on a heavily loaded network. To avoid this problem, set the MTU for the NICs to 9 KB instead of the default 1500 bytes. This requires use of a switch that supports the so-called "jumbo frames."
- Increase the number of nfsd processes. Depending on the load and the power of the server, 16 or 32 are good numbers to try.
- Increase the size of the socket input queue:

  ```
  # echo 262144 > /proc/sys/net/core/rmem_default

  # echo 262144 > /proc/sys/net/core/rmem_max
  ```

These values can be assigned permanently by way of */etc/sysctl.conf*. For example:

```
net.core.rmem_default = 262144

net.core.rmem_max = 262144
```

Other considerations

Because of the nature of the NFS protocol, single-stream write performance is entirely bounded by the size of the I/Os submitted to the client. Since NFS is "stateless" (disregarding file locking), the only way to avoid data loss is for writes to actually be committed to storage. By default, NFS v3 mounts are "synchronous." If a client is submitting 4-K writes, each write must be transmitted, received by the server, submitted to disk, written, and then a response generated. The latency causes very low throughput. If the client submits a 1-MB write, it will be broken down into "wsize" writes (32 K if the preceding tuning is performed). All but the last are immediately acked by the server, and only the final requires a commit/write. This allows for much higher stream performance.

This can be an issue when using commands such as `cp`, which use very small buffer sizes. Use of `cp` to copy a file to an nfs-mounted file system will yield poor I/O rates. Using `dd` with a large blocksize to perform the same operation will give vastly improved results.

It is strongly recommended that Matrix Server PSFS file systems mounted with the `DBOPTIMIZE` option not be exported by way of NFS. This option causes all writes to be synchronous. Write performance by way of NFS to such file systems will be very low regardless of client I/O block size.

For more information

For more information on HP StorageWorks Enterprise File Services Clustered File System, visit:

www.hp.com/go/efs

4AA0-0922ENW, 06/2005