FreeBSD performance tuning

AfNOG 2006

Phil Regnauld

Areas of tuning

- File descriptors and limits
- Virtual memory
- Disk I/O (placing partitions)
- Network buffers / sockets
- ...

1. Increase kernel limits

- With hundreds of processes, the kernel will eventually run out of file handles; after that, it will run out of sockets. Some important parameters to look for are:

```
# sysctl -a | less
...
kern.ipc.nmbclusters: 1024  # no. network memory buffers (2K each)
kern.maxproc: 532           # max. number of processes
kern.maxfiles: 1064         # max. number of open files
kern.maxfilesperproc: 957   # max. number of open files per process
kern.ipc.somaxconn: 128     # the size of the listen queue for
                            #   accepting new TCP connections.
kern.ipc.nmbclusters: 9024  # the number of network mbufs
```

1. Increase kernel limits

- Many of these can be changed on a running system:

```
# sysctl -w kern.maxfiles=16384
# sysctl -w kern.maxfilesperproc=2048
# sysctl -w kern.ipc.somaxconn=1024
# sysctl -w kern.ipc.nmbclusters=32768
```

1. Increase kernel limits

- For this to happen automatically upon next reboot, create "/etc/sysctl.conf" with these lines:

```
kern.maxfiles=16384
kern.maxfilesperproc=2048
kern.ipc.somaxconn=1024
kern.ipc.nmbclusters=32768
```

1. Increase kernel limits

- Others may say "read-only". Some of these can be set at kernel boot time, by adding the following lines to the **/boot/loader.conf** file then rebooting:

```
kern.maxproc="8192" # max nr. of procs
```

see:

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/configtuning-kernel-limits.html

2. Use SCSI disks

- SCSI drives perform much better under heavy use than IDE drives. Higher RPM (rpm) have a shorter latency for accessing data and perform more accesses per second.

- SCSI has another advantage: you can put many devices on a single SCSI bus (7 or 14), and they can be accessed concurrently. That is, once the controller has issued a command to a drive to fetch some data, it disconnects from the bus while it works on the request, and the bus is then free to talk to other drives

- With IDE you need a separate controller for each disk. Putting two disks on one controller does not work well, because 'master' and 'slave' cannot be accessed simultaneously.

  (SATA solves some of these problems, but not all)

3. Spread data on multiple disks

- Try to spread the data across multiple disks, separating service/data I/O from journaling/logging I/O (separate filesystems on separate disks), and separate temporary data from permanent data
- Separate system disk from data disks

4. Use mirroring, not RAID5

- If you have 6 disks, configure them as 3 mirrors of 2 disks:
  - /mail1 -- disk 1 and 2
  - /mail2 -- disk 3 and 4
  - /mail3 -- disk 5 and 6
- With RAID5, you would have more space, but RAID5 writing is *slow*.
- In RAID5, writing a single block on disk 1 requires two reads (read the old data from disk 1 and the old parity data from another disk) followed by two writes (write the new data to disk 1 and the new parity data to another disk).

5. Put in as much RAM as possible

- Any extra RAM is used as disk cache.