OS Tuning and Configuration

Jim Harrell

Objectives

- Describe steps taken to tune an IRIX system.
- Describe some basic performance metrics.
- Describe some basic tuning objectives.
- Describe the tools and configuration options to tune IRIX.
- Answer Questions.

What is Tuning?

- Code optimization
- System use optimization
- Resource management
- Political scheduling
- Site policies
- Matching workload/system size
  - Add hardware to match workload demand
  - Reconfigure software to match workload characteristics
  - Do friendly data movement
  - Reduce workload level with NQE
- Matching site expectations to workload demands

Basic Approach

- Performance evaluation
  - Utilization of CPU, memory, I/O, and networks
  - Workload analysis by process, class, user, batch queue

- Evaluate tuning option alternatives
  - Applications and users
  - File system layout and disk space
  - Process management and CPU partitioning issues
  - I/O − user, fs buffer cache, cachefs, mbufs
  - Memory − paging, migration, replication, coalesce
  - Workload management − NQE/NQS, LSF, Codine

Tuning Methodology

- Determine system configuration
- Establish performance metrics
- Instrument the system
- Establish historical baseline performance
- Monitor system use
- Determine workload classes
- Perform a system analysis (Performance Evaluation)
- Determine your tuning objectives
- Determine your configuration parameters and options
- Test, model, simulate, experiment, and report

Determine The System Capactiy

- Hardware
  - use `hinv -v`, `fx`, `oview`
  - CPUs
  - Memory
  - Disks (SCSI, FCAL, NFS/BDS)
  - Peripherals (tape, graphics, vme devices)
- Software
  - use `systune`, `prtvtoc`, `xlv_make`, `swap`
  - Kernel size and configuration
  - File system layout and swap configuration
  - Fsbuffer cache configuration
  - Memory management configuration
  - Batch configuration (NQS or LSF)

Each of these has
- Capacity
- Speed
- Geometry

Typical Metrics

- CPU utilization
- Memory utilization
- MFLOPS, MIPS
- System time
- Expansion factor (wallclock/service)
- Turnaround ratio (includes batch queue wait time)
- Jobs per day (workload unit/day)
- Latency
- Transactions per second
- Webserver hits per second
- Benchmark metrics such as AIM, Spec, Webstone, LADDIS, STREAM, LINPACK

Instrument The System

- System activity reports
  - `/usr/lib/sa/sa1` every 10 minutes in sys crontab
  - `/usr/lib/sa/sa2` at end of day (comment out find)
- System accounting reports
  - `/etc/init.d/acct start` to turn accounting on
  - `/usr/lib/acct/runacct` to process accounting data
- Extended system accounting reports
  - I/O waits, CPU waits and other data in `/var/adm/sat`
  - `/etc/config/sat_select.options`

```
-off all -on sat_proc_acct
```

- Performance Co-Pilot (licensed)
  - `/var/pcp/config/pmlogger/config.default`
- Application analysis with SpeedShop and ProDev Workshop
- Anything else needs to be instrumented
  - webserver, Cisco router, DBMS, arrays

Establish The Baseline Performance

- Determine if site is interfaced to any analysis tools
- Start with historical sar and accounting data
  - Determine, outlier, peak, average, prime/nonprime use
- Benchmark data used in the sale is a good starting point
- Some sites have a performance test suite to run
- Determine preconceived notions and expectations

- Investigate intervals of unusual sar intervals (outliers)
- Investigate the top applications (CPU, memory, I/O)
- Investigate top users

- Prioritize the top applications

Monitor The System Use

- IRIX 6.5 application tools
  - `time, timex, par`
  - SpeedShop – `ssrun, ssusage, prof, perfex, dprof`
  - ProDev Workshop – `cvperf, cvusage`
- Irix 6.5 system tools
  - `w, who, ps, uptime`
  - `top, gr_top`
  - `osview, gr_osview, gr_nstats`
  - `gmemusage, bufview`
  - `sar, ecstats, topwait, topfunc`
  - `acctcom, runacct, acctcms, sat_interpret`
- Performance Co–Pilot – `pmlogger, oview, mpvis, pmchart`
- OPET/PESTO tools – `xsamon, xsar, sat_acctcom, xwla`
- Kernel profile tools – `prfsnap, kernprof`

Determine Your Workload Classes

- First rule of tuning is **Know Your Workload!!**
- Talk to the data center manager about workload perceptions
- Look at top applications pointed to by `acctcms`
- Is the site Interactive or Batch workload?
- A workload class is similar to ledger account names
  - compare to an accountant going into the company books
- You don't need more than 20 classes
  - Compute
  - Time share, interactive
  - Real time
  - File server
  - Web server
  - Database server
  - Video
- Each category/class is similar in function/resource use

Perform A System Analysis

- Find a typical interval
- Agree with the site that this is a fair representation
- Deal with edge effects, don't prorate

- Correlate rise in sar activity to other sar activities
- Then correlate to the workload that is/was running
- Determine all clock components of the production workload
- Determine bottlenecks by workload class
- Determine number of running processes of each workload
- Compute wallclocks, expansion factors

  Expansion Factor = Elapse/service

  Turnaround Ratio = Wallclock/service
- Determine applications that need to be instrumented

Determine Your Tuning Objectives

- Decrease application wallclock time
- Reduce system time
- Reduce I/O wait time
- Reduce cache thrash
- Increase CPU utilization
- Improve interactive response
- Balance between interactive and batch
- Increase memory utilization
- Reduce paging, swaps, page migration/replication/coalesce
- Accomplish political scheduling goals
- Ensure time critical throughput
- Decrease elapse time of a workload class
- Schedule/share system resources
- Increase total throughput
- Increase number of users, transactions

What Are Your Tuning Options?

- CPU service time
  - Let the compiler do the work
  - Use optimized libraries
  - Recode time consuming algorithms
  - Reduce system call use, stay out of the kernel
  - Reduce cache misses (threads<NCPUs)
  - Reduce tlb misses (maybe go to a larger page size)
  - `dplace` for placement in topology
  - `sn` for system NUMA migration
  - `assign` Flexible File IO layering (7.2 compiler)
  - Many environment variables on mp(3) and mpi(3) man pages

What Are Your Tuning Options?

- CPU wait time
  - `nice, npri, mpadmin`
  - `runon, dplace`
  - `miser_cpuset`
  - `miser_submit`
  - Reduce run level sar -q less than the number of CPUs (pthreads?)
  - Load level with a batch interface like NQE and LSF, this becomes qweight
  - Multi-threaded applications will have more CPU wait time, especially in Time Share
  - Use Miser, Static or Exclusive scheduling to reduce L2 thrashing

What Are Your Tuning Options?

- Page wait time, swap and sbrks
  - Reduce memory mallocs/sbrks, especially intensity (Block Transfer Engine is used)
  - Change swap configuration with swap and /var/sysgen/system/irix.sm to swap on non-root devices
  - Add lots of virtual swap can increase memory utilization, i.e., more large processes at once
  - nbuf sets the limit to kernel size memory growth
  - systune paging group
    - rsshogfrac is limited to physmem*(100-rsshogfrac) or 100MB, whichever is less (6.5)
    - gpgshi, gpgslo should be set with sar -r and sar -p as indicators
    - Kernels' shaked trims kernel down, monitor with sar -R
    - min_file_pages sets the low limit that the kernel fs buffer cache can be trimmed to

What Are Your Tuning Options?

- Page wait time, swap and sbrks (CONTINUED)
  - Increase page size with `dplace`; may reduce CPU service time, but increase page I/O load
  - Memory latency will show in User CPU service time, no performance tool is tracking NUMA latency
  - The TLB miss is charged to User CPU service time and does not vary with load
  - The swap I/O wait is accounted for in extended accounting as:
    - cached I/O and direct-to-swap I/O with `sat_interpret` or `sat_acctcom` (OPET shareware)
  - Code data structure layout on system topology with `dplace`
  - Non-DSOs or replication can reduce node memory reference conflicts
  - vhand is Miser batch critical aware and will steal pages from weightless, opportunistic Miser, time share and lastly steals from realtime

What Are Your Tuning Options?

- I/O service time
  - Device/channel speeds, i.e., get a faster disk device/channel
  - Disk Arrays – RAID, FCAL, Remote vs Direct
  - Minimum file systems per disk drive to avoid seeks from one partition to another
  - Minimum disk drives per SCSI and spread out the I/O file system across SCSIs, use sar –d
  - Split out file systems based on I/O characteristics that are similar – load segregation
  - Device firmware (write buffering is off) and disk HW cache segment configurations
  - Striping wide only helps when request is making stripe width I/O requests; xlv_mgr show stat

What Are Your Tuning Options?

- **Direct I/O wait time** (bypasses file system cache)
  - File system allocation units, `xfs_growfs`
  - Stripe factor and allocation group step size, look at `osview` XFS statistics
  - More file systems with round robin allocation among them
    - striping increases disk wait time
  - GRIO and PRIO I/O scheduling schemes

What Are Your Tuning Options?

- Cache I/O wait time
  - Don't cover up a bad file system layout with cache techniques; a cache eventually does IO
  - Filesystem buffer cache maximum size is set with `systune nbuf`
  - Use `bufview`(1) to view contents of fsbuffer cache
  - `systune bdflushr` determines how many buffers to evaluate each second, i.e., trickle sync
  - `systune min_file_pages` sets the minimum that shaked will trim the fsbuffer cache
  - `systune autoup` determines the age of dirty data to flush to disk by bdflush
  - `systune autoup * HZ / 2` is used to age clean data to inactive state
  - Bypass fsbuffer cache if not reusing data – O_DIRECT, `assign -B` on in 7.2 compilers
  - Do better I/O in the application, look at Flexible File I/O Layering; ffopen(3) or assign –F
  - Do I/O in the right filesystem, NQE has $TMPDIR and LSF has $HOME & .lsbatch

What Are Your Tuning Options?

- Batch queue wait time
  - Waiting for CPU while executing increases cache thrash
  - It's better to wait in a queue (job pool)
  - Give a price break as incentive to use batch interface
  - Change queue detail and time constraints to match load
  - Lopri and express and other political queues
  - Similar to traffic control and metering in big cities
  - Sites are writing their own workload initiation schedulers
  - NQE on IRIX 6.5 doesn't have easy access to queue wait time in the extended accounting
  - Sites have to write their own `setspinfo(2)` for batch accounting
  - Extended accounting session records need high watermark command names, not the shell
  - Distribute across array of systems with NQE's NLB, LSF, Codine
  - Need a queueing system to use Miser scheduling in multi-user environment
  - How do you track a backlog of work?

What Are Your Tuning Options?

- TCP
  - TCP network delays
  - Whether to use XFS/XLV, NFS or BDS
  - Number of TCP mbuf buffers (mbmaxpages)
  - Maximum transmission unit (MTU)
  - Routing table
  - Faster network media (HIPPI, FDDI, ATM)
  - UDP socket buffer sizes (unpxx_sendspace, unpxx _recvspace)
  - Don’t run mrouted and avoid tunnels (replicated data)

What Are Your Tuning Options?

- NFS

  NFS options (rsize wsize)
  `cachefs`
  Transfer size (nfs3_default_xfer)
  Go to BDS or faster file system

Test, Model, Experiment, Simulate

- Test
  - Site has to agree on a test suite (real world)
  - Usually done on real machine with real workload
- Model
  - 3rd party packages – SES/Workbench, TQModel
  - Determine best run limit in a complex workload
  - Bottleneck analysis
- Simulate
  - What if I ran the workload with more CPUs, memory, IO
- Experiment
  - Go dedicated such as a benchmark situation
  - Use extremes on parameters
  - Change more than one parameter at once

Prepare A Report

- Objectives
- Executive summary
- Site description
- Resource use
- Workload characteristics
- Detailed report
  - `sar`
  - accounting
  - extended accounting
  - `perfex, prof, par, ssrun` data
- Recommendations
  - What are the areas of main concern?
  - What are the main workloads and users?
  - What are the workload bottlenecks?