Performance Tuning and Debugging
Don Porter

Why is my application slow?
• No silver bullet
• Part science, part art
  – Science: Measure performance, test hypotheses
  – Art: Finding practical balances of concerns

Most common culprits
• Insufficient resources
  • Configuration error
  • Hardware problems

Digression: Throughput and Latency
• What are they?
  • Throughput: Operations over time
    – Requests per second
    – Transactions per minute
    – Higher is better
  • Latency: Time to complete one operation
    – My server can complete an HTTP GET in .01 seconds
    – Lower is better

What happens when you are overloaded?

What happens when you are overloaded?

Ideal

Realistic

Ideal

Latency

Load (%)

0 25 50 75 100 125 150 175 200

% Maximum Throughput

0 20 40 60 80 100 120

Ideal

0 25 50 75 100 125 150 175 200

Latency

Load (%)

Ideal

0 2 4 6 8

Ideal

0 25 50 75 100 125 150 175 200

% Maximum Throughput

0 20 40 60 80 100 120
What happens when you are overloaded?

Graceful Degradation

- Ideally, when a system is overloaded, by n%, operation latency would increase by n% and throughput would stay constant
- In practice, systems rarely degrade gracefully when they are overloaded
- Thus, finding the “limiting factor” is essential

Note Change in Y Axis Scale— approaches infinity

atop

Graceful Degradation

- Ideally, when a system is overloaded, by n%, operation latency would increase by n% and throughput would stay constant
- In practice, systems rarely degrade gracefully when they are overloaded
- Thus, finding the “limiting factor” is essential

Note Change in Y Axis Scale— approaches infinity

atop

Graceful Degradation

- Ideally, when a system is overloaded, by n%, operation latency would increase by n% and throughput would stay constant
- In practice, systems rarely degrade gracefully when they are overloaded
- Thus, finding the “limiting factor” is essential

Note Change in Y Axis Scale— approaches infinity

atop

Graceful Degradation

- Ideally, when a system is overloaded, by n%, operation latency would increase by n% and throughput would stay constant
- In practice, systems rarely degrade gracefully when they are overloaded
- Thus, finding the “limiting factor” is essential

Note Change in Y Axis Scale— approaches infinity

atop

Graceful Degradation

- Ideally, when a system is overloaded, by n%, operation latency would increase by n% and throughput would stay constant
- In practice, systems rarely degrade gracefully when they are overloaded
- Thus, finding the “limiting factor” is essential

Note Change in Y Axis Scale— approaches infinity

CPU

- Very rarely the bottleneck
  - Actually degrades gracefully in most cases
- Nonetheless, overloaded CPUs will seem less responsive
- Note that when another resource is scarce, CPU time is used trying to compensate

Load Average

- The average number of processes waiting for the CPU
  - Less than 1, the CPU is idle
  - Higher than 1 is ok, just means CPU is fully utilized
  - Very high values (>8) can indicate a problem
- Read from the uptime command:

  `$ uptime`

  20:18:13 up 20 days, 11:08, 5 users, load average: 0.00, 0.03, 0.05
Memory

- Often the biggest troublemaker
- Why?
  - OSes over-commit memory to applications
  - In other words, if I have 1GB RAM, I can have 5 applications that all think they have 300 MB
  - How is this possible?
  - Swapping

Swapping

- If the OS is running low on memory, it can take RAM away from applications
  - Save the contents to disk
  - Reuse the RAM
- If the application tries to read or write to this memory, the application is interrupted, OS notified
  - OS has to then find free RAM, replace contents for app

The problem with swapping

- Disk reads and writes are slow (relative to CPU)
  - You very rarely wait for them before making progress
  - Except when swapping
- Mitigation: OS makes educated guesses about unlikely-to-be-used data to swap out
  - In the best case, things slow down a bit, and then return to normal
- In the worst case, data ping-pongs between disk and RAM
  - Called thrashing

Recommendation

- If you see substantial swap usage in atop, buy more RAM
  - It is cheap, and more RAM is cheaper now than when you bought the computer
- Note: OS often uses substantial amount of RAM to cache the file system contents, so don’t be mislead if total RAM usage is near 100%
  - Look at swap to detect insufficient RAM

In a crisis...

- Linux has an out-of-memory killer
- As advertised, it just kills programs until there is enough memory

Swappiness

- Linux tries to swap some data out before there is a crisis
- Linux has a parameter that sets how aggressively to swap data. This can get out of whack
  - /proc/sys/vm/swappiness
- I’ve personally had to dial this back on an Ubuntu release that set the default too high, in order for a nearly idle system to be usable
Network

• When the network is overloaded, packets are dropped
  – But the other end usually retries
• Two biggest culprits for network overload:
  – Attack (denial of service, brute-force password guessing, spam, etc)
  – Legitimate overload (slashdotted website, peak usage time)
• Need to figure out which

Network advice

• If the overload is not legitimate, good security practice can help to reduce wasteful traffic
  – Firewall, denyhosts, spam filter, etc.
  – For DoS, there are also quality-of-service tools on many network devices to limit the share of packets delivered from any one source
• If the overload is legitimate, you may need more servers and a load-balancer
  – Like round-robin DNS

Disks

• Very rarely the bottleneck, except:
  – (Implicitly when thrashing swap)
  – Actual disk-intensive workloads (e.g., database)
  – And when disk is nearing end-of-life
• Why rarely a problem?
  – Most disk requests are asynchronous
  – Most disk-intensive applications inherently rate-limited
• Why a problem at end-of-life?
  – Heavy remapping yields poor scheduling
  – For SSDs, internal bookkeeping can take longer as the device ages

Disks

• In general, if the disk is getting old, the best advice is replace it
  – You also don’t want to lose data
• Some file systems perform worse as they age, but these are increasingly uncommon
  – Running a “defragmenter” can help

General advice

• Measure a performance baseline for your system
  – Application performance
  – Microbenchmarks (e.g., lmbench)
• If things seem slower, re-measure the component
  – Has my disk bandwidth degraded?
• This is the science of tuning

Other tools

• /proc/cpuinfo, /proc/meminfo, /proc/diskstats – useful system statistics
  – Lots of goodies in /proc
• vmstat – more details on memory usage
• nice/renice – adjust scheduling priority, giving more CPU time to important applications
• swapinfo – more details on swapping
• netstat – more details about network usage
• hdparm/sdparm – measure raw disk performance
• iostat – more details about disk I/O