Inside look at benchmarks

Wim Coekaerts
Senior Vice President, Linux and Virtualization Engineering

Overview

- Purpose of benchmarks
- Who is involved?
- What kind of benchmarks exist out there?
- Benchmarks are useful - improve technology
- Case study : exadata

Purpose...

- Marketing tool to promote new products
  - promote new servers (hardware vendors)
  - promote new OS releases (often go hand in hand with hardware updates)
    - now also starting to include virtualization
  - promote new software product releases (SAP, SAS, Oracle Ebiz, websphere, weblogic, jboss...)
- A way to compare against competitors in numbers
  - TPC-XX, SpecYY
- Help customers to do an initial compare amongst multiple solutions

Purpose...

- Benchmarks are usually presented in 2 ways
  - performance :
    - xx transactions per minute
    - orders processed per minute
    - japp requests per second
    - IO/s
  - price / performance :
    - Cost/TPMC

Who kicks off benchmark work

- Depends on the benchmark of course
- Hardware/OS vendors (IBM, HP, Dell, Oracle...)
  - promote new models released (pricing)
  - promote new chips, new powerX, SPARC, Intel, AMD...
  - new OS release for their platform (AIX, Linux, Solaris...)
- Application vendors (SAP, Oracle, SAS, ..)
  - new database release, new clustered products

Who kicks off benchmark work

- Most of the time a team effort amongst multiple vendors
- Create a workgroup with engineers from various companies
  - this often is a predictable project
  - start out with a target benchmark number calculated beforehand
  - this is not based on ‘oh let’s see what we can get’
  - benchmark engineers are generally very very savvy people that can really calculate results based on specs on paper

Who kicks off benchmark work

- get the hardware and software set up (weeks of effort, or sometimes even more)
- start analysis, profile, modify, rerun, and again...
  - this work takes at least several months
  - make fixes for found bugs
  - make performance enhancements
  - move data layout around on storage
  - add/remove storage, disks, load generators
- audit results by external auditor
- write up and published results usually accompanied by a press release

Types of benchmarks

- microbenchmarks
  - good at measuring specific subsystem performance
    - lmbench, iozone, iperf, netperf etc
  - helps with tuning specific items but are not really good to see how it can help generic application
  - very useful for testing operating systems features and enhancements like filesystem performance or scheduler performance etc
  - non audited - more a tool to help system engineers and kernel/application developers

Types of benchmarks

- microbenchmarks
  - drawback is that changes to a subsystem for one test can negatively impact other areas
  - very little relevance to real world applications
  - convenient to run on even smaller systems - don't always need a large environment to run
    - helps developers with tools on a workstation

Types of benchmarks

- macrobenchmarks
  - use real programs to simulate user scenarios
  - run existing captured workloads
  - synthetic benchmarks
  - many published macrobenchmarks are audited
  - usually requires a very big complex environment with tons of servers, huge amounts of storage, load generator servers
  - test an end to end environment client -> server app -> server OS -> storage/networking
  - audited large benchmarks like a TPC-C or TPC-H are incredibly expensive (millions of $) to complete

Why should you care?

- Developers often shrug off benchmarks as useless
  - just a marketing tool
  - not real world real user application
  - only high-end - doesn't matter on my desktop
- For the most part for developers and regular end-users that's true. however ...
- Look at F1 racing, or space travel
  - often an industry or technology needs extreme driven competition to help improve day to day technology

Why should you care?

- The complexity of these environments discovers very interesting side effects in software and finds tons of bugs
- Sometimes optimize for corner cases but there are in fact many very large companies out there doing extreme things day in, day out - in every sector
- It's a huge investment in time and $ but it pays off - many product improvements, bugfixes are found during these long running extreme workloads tests
- Many of the new large benchmarks are on Linux. yes it scales, yes it can handle it well

Why should you care?

- Given the high hardware cost of these setups, it's a great opportunity to get access to large systems

# of processors / cores	108 / 1728
# of clients	81
# of users	24,300,000
# of disks	720
# of flash arrays	138
# of DRAM	13.8TB

Case study : Exadata

- Why : Introduction of a new database server appliance
- What :
  - 2 x 8 socket servers with 1 TB ram
  - infiniband interconnect
  - 168 disks across 14 storage servers with 168 cores
  - 5 TB flash
- Starting point : OL5/2.6.18 kernel

Case study : Exadata

- 4 x 2 socket machines got 690k iops
- 1 x 8 socket machine got 197k iops

- after 6 months, 300+ bugfixes, new irqbalance

- 1 x 8 socket machine got 1M iops

Case study : Exadata

- RDS code changes (performance, hangs, lockups)
- ipoib memory corruption
  - ipoib module unload crashes
- scsi reset slow / hangs system
- idle bottlenecks (powerstate related, idle=mwait)
- tsc syncing (tsc vs hpet -> tsc much faster but hard to trust) for gettimeofday()
- RDMA performance
- IPC semaphore tuning
- RPS tuning (backport from .35)

Case study : Exadata

- aio regressions
- driver /scsi stack lock contention
- database tuning

- similar system setup benchmark (non-exadata)
- starting point 2.6.18 1.9M tpmC
- end result with kernel updates (2.6.32) 2.8M tpmC

Case study : Exadata

Benchmark	2.6.18 starting point	2.6.32 end result	Gain
8kb flash cache reads (IOPS)	197 thousand	1 million	**400%**
Solid State Disk access	4GB/second	9.5GB/second	**137%**
Infiniband RDS messages, single card (IOPS)	89 thousand	273 thousand	**200%**
8 socket database OLTP (transactions per minute)	**1.8 million**	**3.2 million**	**75%**

http://oss.oracle.com/git/?p=linux-2.6-unbreakable.git;a=summary

Take away

- 2.6.39 is next (for us) in many benchmark efforts we ‘re doing
- we try to stick close to mainline kernels, ala gregkh model, should help us use latest features, should help Linux at large as we do massive testing
- benchmarks are useful - don’t dismiss them - it helps the product and it helps those ‘big’ customers

Cool stuff !