Proposed Scalability and Performance Roadmap

Ken Rozendal
IBM
Fall 2000
General Issues

- Performance work needs to be driven by benchmark analysis.
- There are three aspects of performance to be addressed:
  - absolute performance
  - SMP scalability
  - resource scalability
- Benchmarking work should not be duplicated between participants.
- Performance enhancement work should be coordinated between participants.
Customer Workloads

- Customer workloads come in two types:
  - workloads that can be parallelized – allowing "horizontal scalability"
  - workloads that cannot be easily parallelized – requiring large SMP or NUMA systems for larger capacities

- The goal of scalability for the first category should be linear scalability for 1 to 4–way systems.
- The goal for the second category is linear scalability for as large a system as possible.
Participants in This Work

- The following companies have expressed interest in this work:
  - IBM
  - SGI
  - Intel
  - VALinux

- The following companies/groups might be interested in this work:
  - HP
  - CITI (University of Michigan)
Performance Approach

- All work should be done on SourceForge sites.
- Separate SourceForge projects will be created for sub–projects as they become more active.
- Each piece of scalability work should be treated as a separate project and should be broken into as many independent patches as feasible.
- Each piece of work should independently be addressed as far as Linux community integration.
Choose the best of publicly available and proprietary benchmarks.
Use proprietary benchmarks for company internal validation and comparison.
Use publicly available benchmarks for Linux community (external) validation and comparison.
Create and encourage creation of good publicly available benchmarks.
Provide results of publicly available benchmarks on SourceForge site.
Classification of Benchmarks

- benchmarks that are currently set up and running in a participant’s lab
- benchmarks that are easy to set up and can be up and running quickly
- difficult to set up benchmarks that are nevertheless required
- good benchmarks that are publicly available
Currently Set Up Benchmarks

- Volanomark and Volano C (at IBM)
- FSCache (at IBM)
- Netperf (at IBM)
  - streaming
  - request – response
  - connect – request – response
- SpecWeb99 (at IBM)
Simple Setup Benchmarks

- SPEC sdet
- IOZONE
- PostMark
Important Difficult Benchmarks

- TPC–C
- TPC–D
- TPC–W
- kenbus
- SPEC SFS
Good Public Benchmarks

- AS3AP
  - ANSI SQL Standard Scalable and Portable benchmark
Minimum Benchmark Platforms

- 1–way x86 system
  - verify reference system impacts
- 4–way x86 system
  - verify "sweet spot" scalability impacts
  - verify horizontal scalability benchmark impacts
- 8–way x86 system
  - push current SMP scalability limits
- (future) 16–way and NUMA systems
  - push future SMP scalability limits
  - evaluate NUMA scalability issues
Crucial Customer Workloads

- web serving
- web application serving
- database serving
- file and print serving
- application serving
- internet service providing
Horizontal Scaling Workloads

- web serving
- web application serving
- application serving
- internet service providing
SMP Scalability Workloads

- database serving
- file and print serving
**Staging Approach**

- **short term**
  - first six months time frame
  - proceed on both benchmarking and prototyping in parallel
  - provide scalability infrastructure

- **medium term**
  - six months to one year time frame
  - start benchmark directed prototyping

- **long term**
  - one year and beyond time frame
  - continuous benchmarking and prototyping with increasing goals
**Short Term Approach**

- Get easy benchmarks in place ASAP.
- Get 8–way system benchmark results.
- Publish public benchmark results to SourceForge site.
- Identify the benchmarks for future work.
- Work on obvious scalability issues until benchmarks direct future work.
- Start setting up difficult benchmarks.
- Provide required scalability infrastructure for future work.
- Provide required performance tools.
Medium Term Approach

- Start running key customer benchmarks (even difficult ones).
- Get benchmark numbers across 1, 4, and 8-way systems.
- Compare results with other systems (AIX, Windows NT, Solaris, IRIX, etc.) to look for performance differences.
- Publish public benchmark results to SourceForge site.
- Split off larger subprojects to their own SourceForge sites (NUMA, scheduler, VM, etc.)
Staging Approach

- Get full lab with 1, 4, 8, and 16-way systems as well as NUMA systems.
- Get continuously running benchmarks to test ongoing prototypes.
- Publish public benchmark results to SourceForge sites.
Performance Requirements

- required performance analysis tools
- required performance analysis systems
- database performance analysis platforms
Required Performance Tools

- lockmeter
- user level tprof
- NMI–based kernprof
- trace facility (LTT)
- Above idle
- Mtrace
Required Scalability Infrastructure

- enhanced locking primitives
- NUMA aware allocation
- NUMA aware locality policies
Database Analysis Platforms

- **database products** –
  - DB2 (stable and scalable)
  - Oracle 8i (stable and scalable)
  - mySQL (unstable with scalability issues?)
  - PostGreSQL (unstable with scalability issues?)
  - Interbase (not full database product?)

- **database storage technologies** –
  - RAID systems
  - fiber channel interconnects
  - SCSI
Web Serving Platform

- web servers:
  - Apache – widely used
  - Zeus – currently very scalable

- web servers that are not ready:
  - tux – still unreliable under stress?
  - kHTTPd – still unreliable under stress?

- need to evaluate serving of different types of content:
  - static content only
  - dynamic content only
  - mixed static and dynamic content
Initial Performance Work

- suspected performance issues
  - code pathlength and latency issues
- suspected SMP scalability issues
  - "first principles" scalability issues
- suspected resource scalability issues
  - "compiled in" resource limits
  - data structure scalability limits
  - resource management algorithm scalability problems
Suspected Performance Issues

- gcc generated code (particularly on IA-64 platforms)
- Java
- boot time (device configuration, etc.)
- interrupt handling (latency)
Suspected SMP scalability issues

- task scheduling
- virtual memory management (VM)
- communications device drivers
- TCP/IP
- storage device drivers
- web server
- kernel locking
- kernel preemptibility (SMP only)
- buffer cache management
- IPC (semaphores, shared memory, message queues, and pipes)
Resource Scalability Issues

- number of runnable tasks
- number of threads in a "process"
- number of storage devices
- number of communications devices
- rate of storage I/O
- rate of communications I/O
- number of open files
- size of files
- size of filesystems
- size of devices
- size of physical memory
- size of swap space