Performance Monitoring in the Linux Kernel

Davidlohr Bueso
SUSE Labs

Setting Expectations

- This is not a kernel topic per-se.
- Most of this can also be applied to userland.
- Lots to cover, not enough time to go into depth.
  - Ping me after the talk.

SUSE Kernel Performance Team

- Actively monitor and fix performance regressions.
  - Including developing new features that address scalability limitations. – R&D
- Compare performance on a wide range of workloads, configurations and systems.
  - SLE products, including openSUSE. (priority)
  - Upstream.
- We rely on MMtests almost entirely nowadays. New benchmarks and monitors are constantly being added.

Agenda

- Introduction
- Performance Methodologies.
- Performance in the Kernel.
- Marvin/MMTests.
- Real Example.
- Q&A.

Introduction

- Systems performance is a primary concern for the server industry.
  - We just bought a super expensive server, why don't I get lineal scalability?
  - Why does my system actually performance worse than on my 2-socket box?
- Unfortunately performance and scalability go way beyond buying expensive toys.
- With increasingly more powerful machines and more complex software, performance becomes more and more ambiguous.

Introduction

- As with bugs, performance issues can appear in all levels of the software stack. This talk focuses on the OS kernel.
- Performance is an ever-changing, never ending story.
  - Even simple workloads can trigger unexpected results.

Performance Methodologies

Performance Methodologies

- There are dozens of Linux performance tools available for free.
  - Observability
  - Benchmarking
  - Tuning
- Methodologies are useful in that they are guidelines for choosing tools for appropriate for performance analysis.

Methodologies (B. Gregg)

- Anti-Methodologies
  - Blame-Someone-Else, Streetlight, Drunk Man, Random Change Anti-Method, etc.
- Methodologies
  - Scientific method, workload characterization, by layer, TSA, problem statement, RTFM, drill-down.
- Whatever methodology is chosen, it will at least require thinking about the process.

Methodologies: Problem Statement

- Primary starting point.
- Is this a performance problem?
- Did this ever perform well?
  - Importance of historical data.
- What, if anything, changed recently?
- What are the relevant metrics?
  - Ie, regression report stating that IO ops/sec are low may not be a performance regression if it can be shown that the throughput of the application is high and the low IOps/sec.
- QA or customers report a bug.
  - The more data, the better.

Methodologies: Bisection

- Select two points in time (good and bad) and do binary search.
- Only works when dealing with one component and a single root cause.
- Can identify problems from unknown sources.
- Does not require familiarity with the software component.
- Results are easy to verify and the process can be entirely automated.

Methodologies: Workload Characterization

- Identifies the source, reason and type of the load, even between phases of behavior
  - What is the load?
  - Who is causing the load?
  - Why is the load called?
  - How is the load changing over time?
- Can be combined with a drill-down approach
  - Iteratively drills down until a small subset of components are identified and the problem area is well bounded.
  - May also be combined with live monitoring.

Interpreting Results

- There are generally three outcomes that may occur:
  - Legitimate problem. Fix it and carry on.
  - Not a real issue.
  - Trade-off: Certain parts can be negatively affected, but the overall benefits exceed those issues.

Performance Monitoring in the Kernel

Performance in the Kernel

- The kernel can be tightly coupled with hardware or workloads.
  - As such, performance issues might be very different from one machine to another.
- The kernel is a complex system, millions of LoC, users, bugs etc. It is too much to cope with manually.
- Efforts to automate and simplify testing in the kernel are not new.
  - Should get out of the way and let users focus on the problem and not the configs and setting up the environment.
  - Less prone to errors.

Performance in the Kernel

- Developers are more focused on performance of their own patches. Which is certainly a good thing.
- But we also need a picture of the forest.
  - Enables Linux to be scalable, not just a part of it.
  - Requires infrastructure both at hardware and software levels.
- … and how kernels compare between each other.
- Approaches to performance can be considered either reactive or proactive.

Reactive Monitoring: 0-day tests

- Well known in the kernel community.
- Developed by Fengguang Wu.
- Kernel test infrastructure.
- Per-patch granularity.
- Short lived workloads.
- Incredible valuable
  - The kernel is undoubtedly better because of kbuild robot.

Reactive Monitoring: 0-day tests

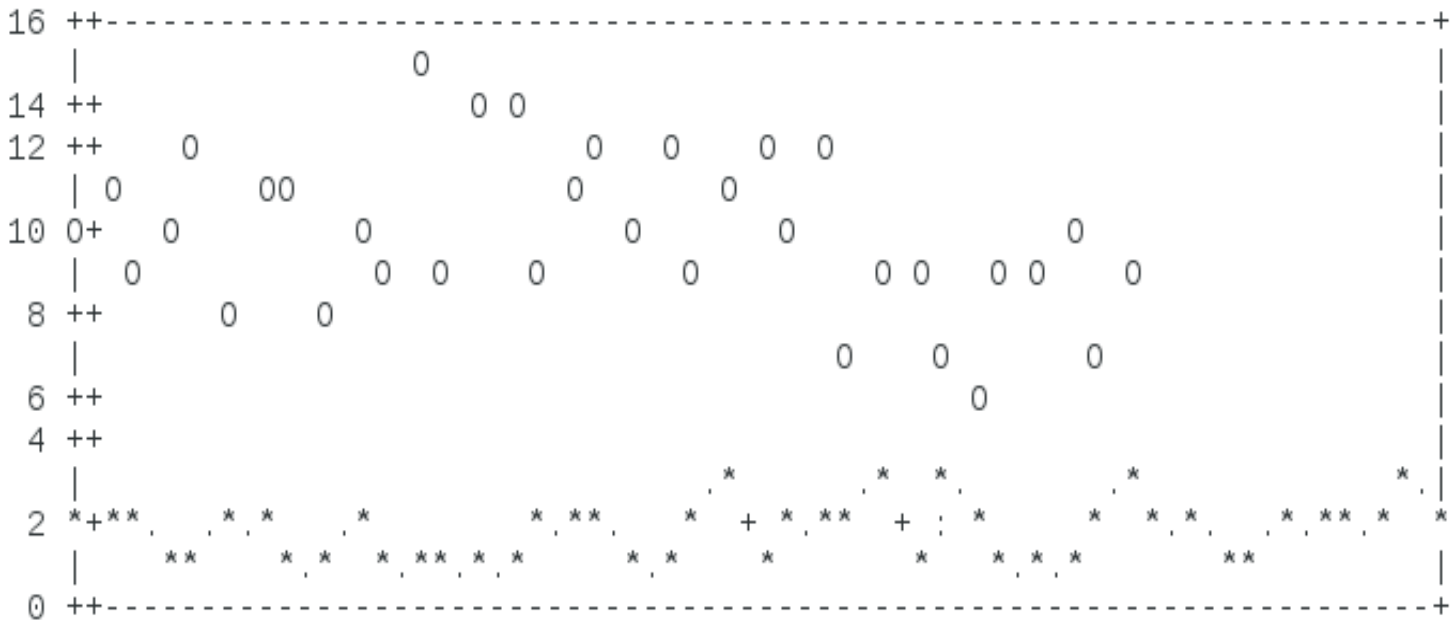
FYI, we noticed the below changes on

git://git.kernel.org/pub/scm/linux/kernel/git/next/linux-next.git master
commit b3fd4f03ca0b9952221f39ae6790e698bf4b39e7 ("locking/rwsem: Avoid deceiving lock spinners")

testbox/testcase/testparams: brickland3/vm-scalability/performance-300s-small-allocs

7a215f89a0335582 %stddev	%change	b3fd4f03ca0b9952221f39ae67 %stddev	
2.38 ± 18%	+247.4%	8.27 ± 16%	vm-scalability.stddev
24226456 ± 0%	-23.0%	18650874 ± 1%	vm-scalability.time.voluntary_context_switches
1622 ± 0%	-15.6%	1369 ± 1%	vm-scalability.time.system_time
765 ± 0%	-13.1%	665 ± 1%	vm-scalability.time.percent_of_cpu_this_job_got
1036 ± 0%	-11.9%	914 ± 1%	vm-scalability.time.user_time
7.244e+08 ± 0%	-11.1%	6.439e+08 ± 1%	vm-scalability.time.minor_page_faults
10841276 ± 0%	-11.1%	9639981 ± 1%	vm-scalability.throughput
8922883 ± 3%	-70.2%	2658543 ± 4%	cpuidle.C1-IVT-4S.usage
1.19e+10 ± 2%	-66.2%	4.016e+09 ± 4%	cpuidle.C1-IVT-4S.time
1.97 ± 4%	+106.7%	4.08 ± 6%	perf-profile.cpu-

Reactive Monitoring: 0-day tests



Phronix Test Suite (PTS)

- Industry-backed benchmark suite.
- Fully automated and extremely easy to use.
- Traditionally used to compare performance of different operating systems.
- Mostly multimedia (CPU-bound) workloads (Doom, ffmpeg) etc.
  - There is certainly things we can learn from this.
  - Goes way beyond microbenchmarks (Doom3).
- But the data needed to debug a problem is rarely included in the reports.
- Very compiler-dependent.

Performance in the Kernel

- There is no unique solution, approaches ought to be complimentary.
- Manual inspection is always required.

Marvin & MMTests

Marvin

- Created by Mel Gorman.
- System that continually runs performance-related tests.
- Mostly written in perl and bash.
- Composed of:
  - Bob: Monitors git trees looking for new kernels to test. Builds and queues these new kernels.
  - MMTests: Performance framework.
  - Inventory Manager: Configuration manager that is responsible for machine reservation, managing power, serial consoles and deploying distributions automatically

Active Monitoring: MMTests

- Design-once.
- Nowadays actively developed (mostly) by the SUSE Performance team.
- Focus more on long lived workloads.
- Complimentary to 0-day tests.
- Guided by run configurations.
- Provides multiple monitors which gives the necessary data to do performance analysis.
  - /sys/kernel/debug/*, /proc/*, perf events, ftrace, stap.
- Reports can be either standard text (ideal for changelogs) or html.

Workload Coverage

- MMTests includes most of the traditional microbenchmarks
  - Unixbench, WIS, lmbench, ebizzy, etc etc.
- Industry standard loads
  - Aim, SPEC*, etc.
- Specific subsystems
  - futex, NUMA, ipc, real-time, etc.
- Customer-like and higher level workloads
  - Compression, RDBMS, Big Data/Hadoop.

MMTests Config Files (1/3)

- Currently ~120 default files to choose from.
- Run one or more workloads, and define each workloads's input parameters.
  - Default values of these parameters are quite good in many cases, but can be modified to suit specific needs.
- Grouped by workload component and characteristics

```
dave@linux-q0g1:~/code/mmtests/configs> ls *__workload*
config-global-dhp__workload_dedup       config-global-dhp__workload_poundtime
config-global-dhp__workload_freqmine    config-global-dhp__workload_pyarray
config-global-dhp__workload_futex       config-global-dhp__workload_syscalls
config-global-dhp__workload_ipc         config-global-dhp__workload_thpscale
config-global-dhp__workload_kerndevel   config-global-dhp__workload_will-it-scale-io
config-global-dhp__workload_mailserver  config-global-dhp__workload_will-it-scale-sys
```

MMTests Config Files (2/3)

```
config-global-dhp__io-bonnie-async
config-global-dhp__io-bonnie-async-fixed
config-global-dhp__io-bonnie-fsync
config-global-dhp__io-bonnie-fsync-fixed
config-global-dhp__io-compress-tmpfs
config-global-dhp__io-dbench4-async
config-global-dhp__io-dbench4-fsync
config-global-dhp__io-ddread
config-global-dhp__io-ddwrite
config-global-dhp__io-filebench-oltp
config-global-dhp__io-filebench-oltp-directio
config-global-dhp__io-filebench-varmail-large
config-global-dhp__io-filebench-varmail-medium
config-global-dhp__io-filebench-varmail-small
config-global-dhp__io-filebench-webproxy-large
config-global-dhp__io-filebench-webproxy-medium
config-global-dhp__io-filebench-webproxy-small
config-global-dhp__io-filebench-webserver-small
config-global-dhp__io-iozone
config-global-dhp__io-iozone-dio
config-global-dhp__io-iozone-ram
config-global-dhp__io-iozone-ram-dio
config-global-dhp__io-iozone-small
config-global-dhp__io-largeread-starvation
config-global-dhp__io-metadata
config-global-dhp__io-randread-starvation
config-global-dhp__io-seeker-block-read
config-global-dhp__io-seeker-block-write
config-global-dhp__io-seeker-file-read
config-global-dhp__io-seeker-file-write
config-global-dhp__io-sync-starvation
config-global-dhp__io-threaded
config-global-dhp__io-xfsrepair
```

MMTests Config Files (3/3)

```
export MMTESTS="futexbench-hash futexbench-requeue futexbench-wake"

# Machine configuration
# Swap configuration can be one of default, partitions, swapfile, NFS
export SWAP_CONFIGURATION=default
export SWAP_PARTITIONS=
export SWAP_SWAPFILE_SIZEMB=$((MEMTOTAL_BYTES/1048576))
export SWAP_NFS_MOUNT=192.168.10.7:/exports/`hostname`-swapfile
...
export VM_TRANSPARENT_HUGEPAGES_DEFAULT=default

# Optionally use a memory control group
# export MEMCG_SIZE=$((MEMTOTAL_BYTES/2))


# List of monitors
export RUN_MONITOR=yes
export MONITORS_GZIP="proc-vmstat top numa-numastat numa-meminfo numa-convergence numa-scheduling"
export MONITORS_WITH_LATENCY="vmstat iostat"
export MONITOR_UPDATE_FREQUENCY=10

# futexbench
export FUTEXBENCH_MIN_THREADS=2
export FUTEXBENCH_MAX_THREADS=$((NUMCPUS*64))
```

The Dashboard (1/3)

- Graphical (html) layout of all tests across all systems.
- Provides a high-level view of multiple kernel subsystems.
- Provide single value analysis.
  - Performance delta. Ie: 1.02 would mean there is a 2% difference and the color indicates whether it is a performance regression or gain.
  - Automatically guess if the result is significant. Ie: 0.98 on one test might be a 2% regression or in the noise range for different one.
  - Red = bad, green = good, gray/white = neutral.
- It's fun!
  - But it *can* be wrong, or misleading.

The Dashboard (2/3)

- Designed for over-simplification.
- Might appear to be a *Traffic Light* anti-methodology.
  - Intuitive, quick, attention grabbing.
  - But we don't set thresholds by guessing.
  - Comparisons are against how *other* kernels perform.

The Dashboard (3/3)

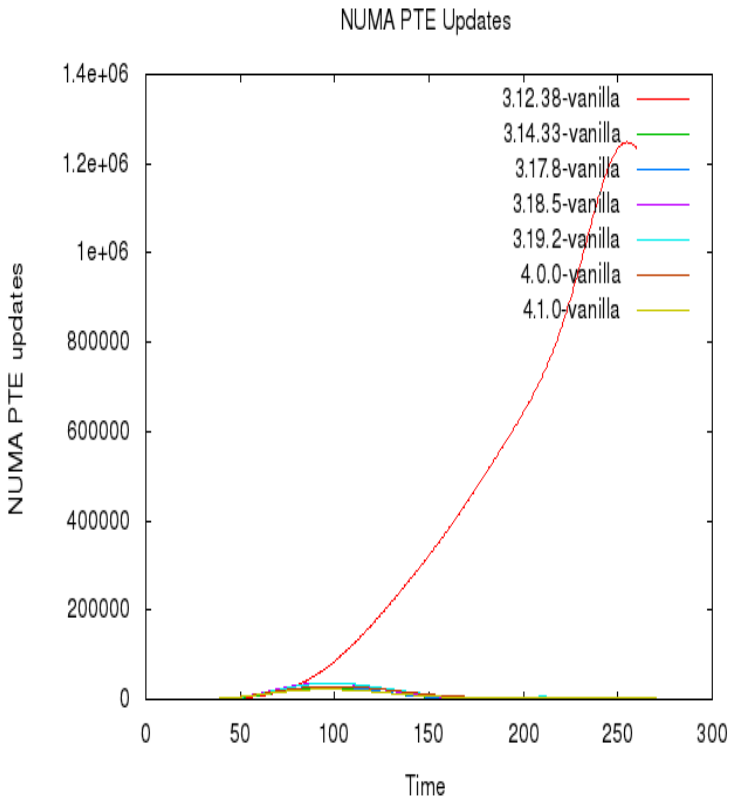
Dashboard

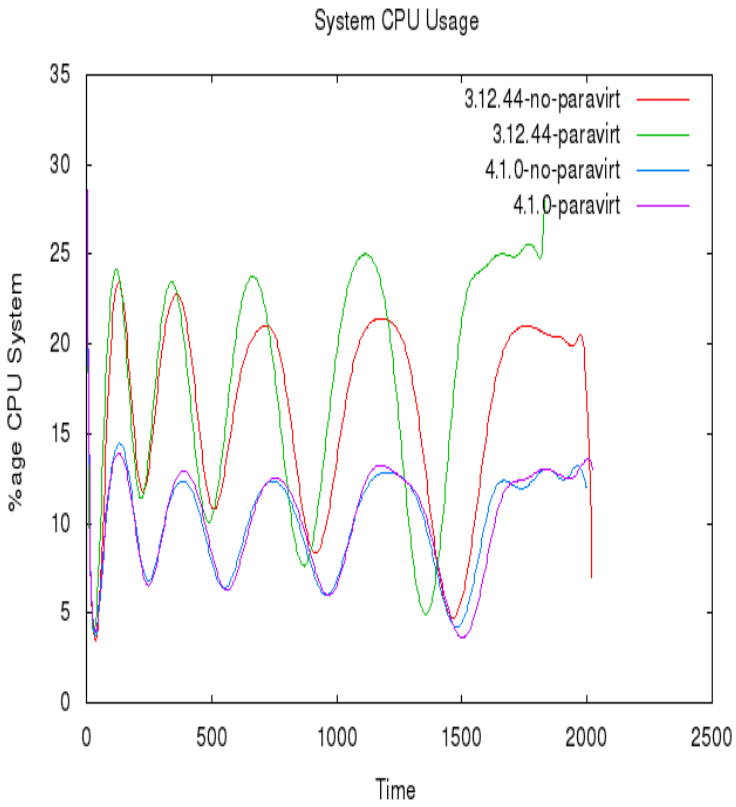
Config	alchiba	balada	bartholdy	cherubini	decker	jacobi	joplin	marvin2	mas	uv300
global-dhp__db-dbt5-smallest-ext3	dbt5-install 1.01 dbt5-bench 1.10	dbt5-install 0.90 dbt5-bench 1.38	dbt5-install 1.02 dbt5-bench 1.06	dbt5-install 0.98 dbt5-bench 1.43	dbt5-install 0.93 dbt5-bench 1.17	dbt5-install 0.93 dbt5-bench 0.95	dbt5-install 1.00 dbt5-bench 1.00	dbt5-install 0.95 dbt5-bench 1.27	dbt5-install 0.97 dbt5-bench 0.88	None
global-dhp__db-dbt5-smallest-ext4	dbt5-install 1.00 dbt5-bench 1.00	dbt5-install 0.87 dbt5-bench 0.86	dbt5-install 1.03 dbt5-bench 1.11	dbt5-install 1.06 dbt5-bench 0.76	dbt5-install 1.04 dbt5-bench 0.88	dbt5-install 0.96 dbt5-bench 0.92	dbt5-install 1.00 dbt5-bench 1.00	dbt5-install 0.96 dbt5-bench 1.29	dbt5-install 0.93 dbt5-bench 0.84	None
global-dhp__db-dbt5-smallest-xfs	dbt5-install 1.02 dbt5-bench 1.25	dbt5-install 0.92 dbt5-bench 0.98	dbt5-install 1.03 dbt5-bench 1.07	dbt5-install 1.06 dbt5-bench 0.93	dbt5-install 1.00 dbt5-bench 0.78	dbt5-install 1.00 dbt5-bench 0.92	dbt5-install 1.00 dbt5-bench 1.00	dbt5-install 1.00 dbt5-bench 1.21	dbt5-install 1.00 dbt5-bench 0.85	None
global-dhp__db-pgbench-ro-large-ext3	pgbench 0.94	pgbench 1.24 1.14	pgbench 0.85	pgbench 1.41 1.39	pgbench 1.00 1.04	pgbench 1.04 1.05	pgbench 1.00	pgbench 0.90	pgbench 0.95	None
global-dhp__db-pgbench-ro-large-ext4	pgbench 0.82	pgbench 1.33 1.38	pgbench 0.99	pgbench 1.29 1.29	pgbench 1.22 1.17	pgbench 1.06 1.06	pgbench 1.00	pgbench 0.90	pgbench 1.09	None
global-dhp__db-pgbench-ro-large-xfs	pgbench 0.95	pgbench 1.10 1.08	pgbench 0.90	pgbench 1.20 1.18	pgbench 1.17 1.17	pgbench 1.03 1.03	pgbench 1.00	pgbench 0.94	pgbench 0.86	None
global-dhp__db-pgbench-ro-medium-ext3	pgbench 0.79	pgbench 1.44 0.99	pgbench 1.66	pgbench 1.39 1.71	pgbench 0.94 0.94	pgbench 1.46 1.61	pgbench 1.00	pgbench 2.00	pgbench 0.81	None
global-dhp__db-pgbench-ro-medium-ext4	pgbench 1.66	pgbench 1.33 1.45	pgbench 1.66	pgbench 1.54 1.99	pgbench 0.94 0.96	pgbench 1.48 1.63	pgbench 1.00	pgbench 0.58	pgbench 0.79	None
global-dhp__db-pgbench-ro-medium-xfs	pgbench 1.67	pgbench 1.43 1.24	pgbench 1.69	pgbench 1.47 2.03	pgbench 1.02 0.93	pgbench 1.48 1.65	pgbench 1.00	pgbench 2.86	pgbench 0.85	None
global-dhp__db-pgbench-ro-small-btrfs	pgbench 1.86	pgbench 1.37 1.28	pgbench 1.89	pgbench 1.28 2.18	pgbench 0.92 0.97	pgbench 1.28 1.67	None	pgbench 1.00	pgbench 1.00	None
global-dhp__db-pgbench-ro-small-ext3	pgbench 1.48	pgbench 1.13 1.21	pgbench 1.92	pgbench 1.20 2.17	pgbench 0.86 0.88	pgbench 1.27 1.63	pgbench 1.00	pgbench 1.89	pgbench 0.73	None
global-dhp__db-pgbench-ro-small-ext4	pgbench 1.45	pgbench 1.16 1.24	pgbench 1.91	pgbench 1.20 2.14	pgbench 0.94 0.95	pgbench 1.27 1.66	pgbench 1.00	pgbench 2.81	pgbench 0.70	None
global-dhp__db-pgbench-ro-small-xfs	pgbench 1.46	pgbench 1.19 1.29	pgbench 1.92	pgbench 1.25 2.15	pgbench 0.96 1.03	pgbench 1.32 1.67	pgbench 1.00	pgbench 2.00	pgbench 0.69	None
global-dhp__db-pgbench-ro-xlarge-ext3	pgbench 1.00	pgbench 1.00 1.03	pgbench 1.00	pgbench 0.98 0.96	pgbench 1.00 1.05	pgbench 1.01	pgbench 1.00	None	pgbench 1.00	None
global-dhp__db-pgbench-ro-xlarge-ext4	pgbench 1.00	pgbench 1.00 0.97	pgbench 1.00	pgbench 0.99 0.98	pgbench 0.95 0.95	pgbench 1.01	pgbench 1.00	None	pgbench 1.00	None
global-dhp__db-pgbench-ro-xlarge-xfs	pgbench 1.00	pgbench 1.00 0.99	pgbench 1.00	pgbench 0.97 0.98	pgbench 1.14 1.08	pgbench 1.07	pgbench 1.00	None	pgbench 1.00	None
global-dhp__db-pgbench-rw-large-ext3	pgbench 1.20	pgbench 1.34 2.04	pgbench 0.95	pgbench 1.30 1.25	pgbench 1.20 1.22	pgbench 0.93 1.03	pgbench 1.00	pgbench 0.98	pgbench 0.94	None
global-dhp__db-pgbench-rw-large-ext4	pgbench 0.77	pgbench 0.92 0.72	pgbench 1.02	pgbench 1.08 1.15	pgbench 1.04 1.05	pgbench 1.01 0.97	pgbench 1.00	pgbench 0.98	pgbench 1.07	None
global-dhp__db-pgbench-rw-large-xfs	pgbench 0.90	pgbench 1.13 1.13	pgbench 1.01	pgbench 1.06 1.06	pgbench 1.06 1.02	pgbench 1.00 1.02	pgbench 1.00	pgbench 0.97	pgbench 0.89	None
global-dhp__db-pgbench-rw-medium-ext3	pgbench 1.53	pgbench 1.09 0.92	pgbench 1.19	pgbench 6.94 7.03	pgbench 3.13 3.22	pgbench 3.72 4.12	pgbench 1.00	pgbench 2.33	pgbench 1.03	None
global-dhp__db-pgbench-rw-medium-ext4	pgbench 1.21	pgbench 1.12 1.13	pgbench 1.16	pgbench 3.55 3.44	pgbench 1.03 1.02	pgbench 1.78 1.76	pgbench 1.00	pgbench 0.78	pgbench 1.01	None
global-dhp__db-pgbench-rw-medium-xfs	pgbench 0.96	pgbench 1.28 1.25	pgbench 1.12	pgbench 2.58 2.73	pgbench 0.99 0.95	pgbench 1.57 1.58	pgbench 1.00	pgbench 1.04	pgbench 0.98	None

Visual Aids (1/2)

- Analyzing a performance regression quickly becomes less tedious when not looking at numbers after numbers after numbers.
  - Of course, as analysis gets more involved, there is no substitution for manual inspection. Period.
- While it is expected to at least graph the benchmark results, much of the data to debug issues are in the metrics and system-wide stats.

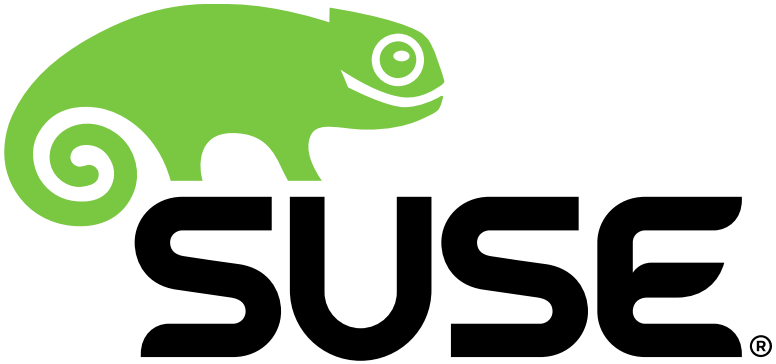
Visual Aids (2/2)





Real Example

Thanks!
Questions?


SUSE®
We adapt. You succeed.