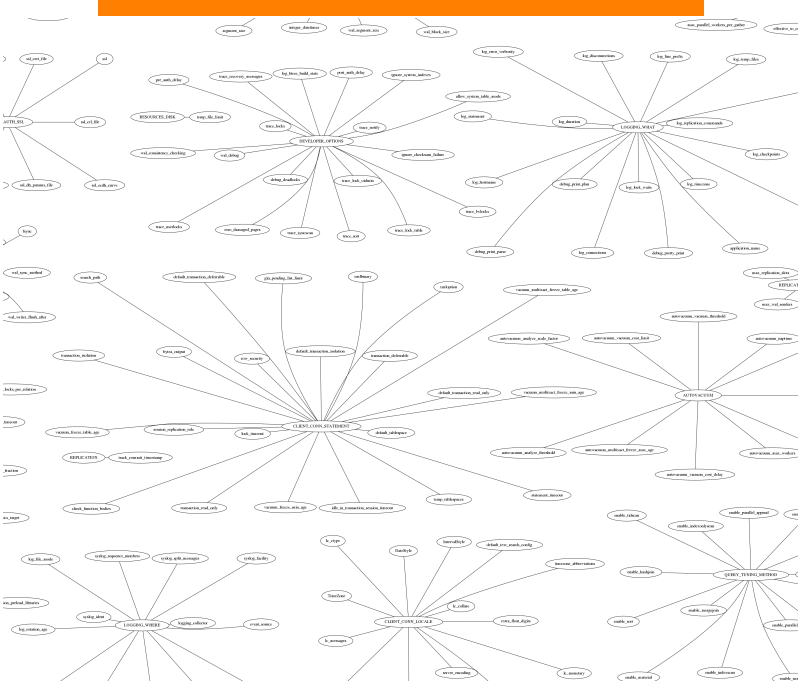
zalando

POSTGRESQL LINUX KERNEL

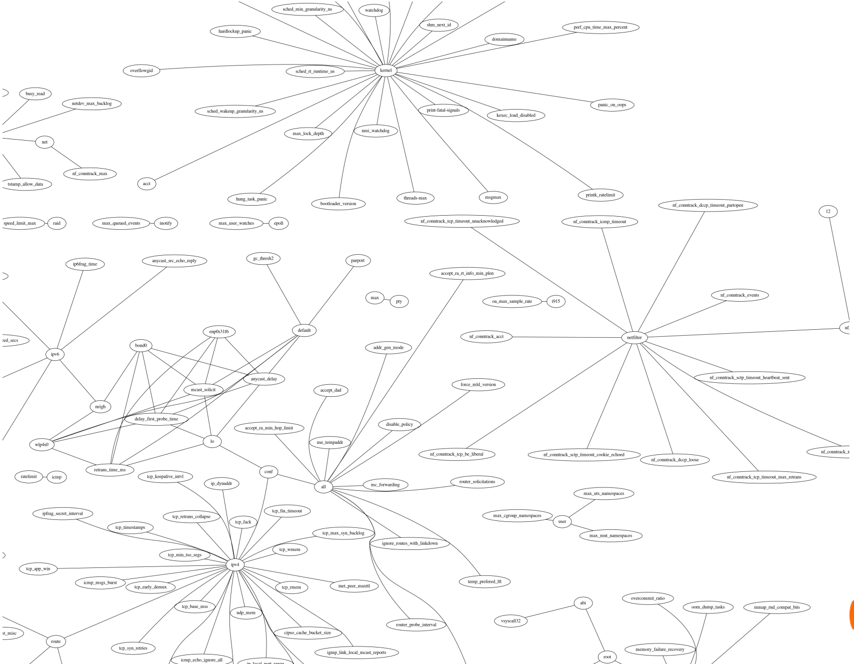
FRIENDSHIP

DMITRY DOLGOV

22-10-2018


DEVELOPER_OPTIONS
LOGGING_WHAT
CLIENT_CONN_STATEMENTS
AUTOVACUUM
QUERY_TUNING_METHOD
CLIENT_CONN_LOCALE
LOGGING_WHERE
AUTH_SSL

zalando


sched_min_granularity_ns
watchdog
shm_next_id
hardlockup_panic
domainname
perf_cpu_time_max_percent
overflowgid
sched_rt_runtime_us
kernel
busy_read
netdev_max_backlog
sched_wakeup_granularity_ns
print-fatal-signals
kexec_load_disabled
panic_on_oops
max_lock_depth
nmi_watchdog
net
nf_conntrack_max
tstamp_allow_data
acct
hung_task_panic
bootloader_version
threads-max
msgmax
printk_ratelimit
nf_conntrack_dccp_timeout_partopen
12
speed_limit_max
raid
max_queued_events
inotify
max_user_watches
epoll
nf_conntrack_tcp_timeout_unacknowledged
nf_conntrack_icmp_timeout
ip6frag_time
anycast_src_echo_reply
gc_thresh2
parport
accept_ra_rt_info_min_plen
max
pty
oa_max_sample_rate
i915
nf_conntrack_events
default
enp0s31f6
bond0
ipv6
addr_gen_mode
nf_conntrack_acct
netfilter
anycast_delay
mcast_solicit
accept_dad
force_mld_version
nf_conntrack_sctp_timeout_heartbeat_sent
neigh
delay_first_probe_time
disable_policy
accept_ra_min_hop_limit
use_tempaddr
wlp4s0
lo
nf_conntrack_tcp_be_liberal
nf_conntrack_sctp_timeout_cookie_echoed
nf_conntrack_dccp_loose
retrans_time_ms
conf
ratelimit
icmp
tcp_keepalive_intvl
ip_dynaddr
all
mc_forwarding
router_solicitations
nf_conntrack_tcp_timeout_max_retrans
max_uts_namespaces
ipfrag_secret_interval
tcp_retrans_collapse
tcp_fin_timeout
max_cgroup_namespaces
user
tcp_timestamps
tcp_fack
max_mnt_namespaces
tcp_max_syn_backlog
ignore_routes_with_linkdown
tcp_min_tso_segs
tcp_wmem
tcp_app_win
ipv4
icmp_msgs_burst
tcp_early_demux
tcp_rmem
inet_peer_maxttl
temp_prefered_lft
overcommit_ratio
abi
oom_dump_tasks
mmap_rnd_compat_bits
tcp_base_mss
udp_mem
vsyscall32
cipso_cache_bucket_size
router_probe_interval
route
tcp_syn_retries
igmp_link_local_mcast_reports
memory_failure_recovery
icmp_echo_ignore_all
root

Plan

Overview

Plan

Overview

Resources

- ➔ Execution scheduling
- ➔ Memory management
- ➔ Storage IO

Plan

Overview

Resources

- ➔ Execution scheduling
- ➔ Memory management
- ➔ Storage IO

Abstracted layers

- ➔ Virtualization
- ➔ Containerization

Plan

Overview

Resources

- ➔ Execution scheduling
- ➔ Memory management
- ➔ Storage IO

Abstracted layers

- ➔ Virtualization
- ➔ Containerization

How to look inside

Overview

Bidirectional resource management

IPC mechanisms

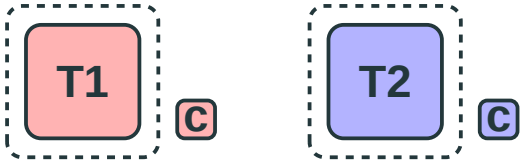
New hardware support

…

Execution scheduling

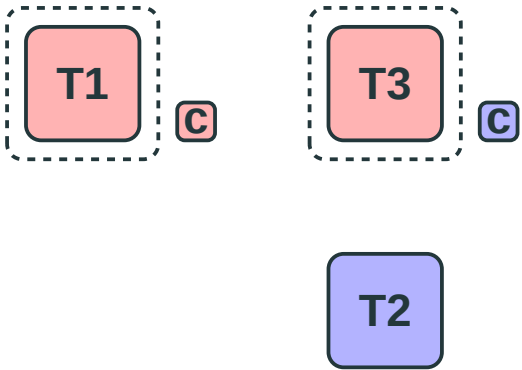
```
# Experiment 1
transaction type: pg_long.sql
latency average = 1312.903 ms

# Experiment 2
SQL script 1: pg_long.sql
 - weight: 1 (targets 50.0% of total)
 - latency average = 1426.928 ms

SQL script 2: pg_short.sql
 - weight: 1 (targets 50.0% of total)
 - latency average = 303.092 ms
```

Scheduling



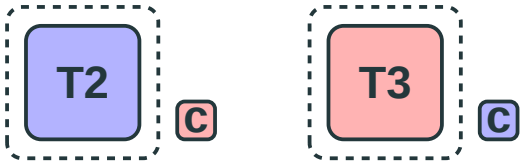
zalando

Scheduling



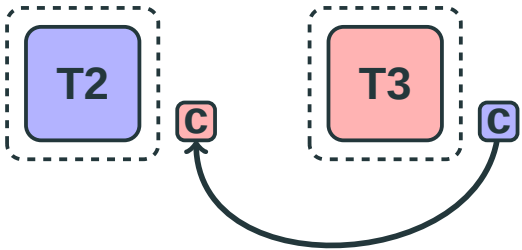
zalando

Scheduling



zalando

Scheduling



zalando

```
# Experiment 1
12,396,382,649      cache-misses # 28.562%
 2,750              cpu-migrations

# Experiment 2
20,665,817,234      cache-misses # 28.533%
10,460              cpu-migrations
```


zalando

```
         time    cpu  01234  task name                    wait time  sch delay   run time
                             [tid/pid]                       (msec)     (msec)     (msec)
--------------- ------  -----  ---------------------------  ---------  ---------  ---------
   4227.834476 [0003]     s   postgres[12935]                 0.000      0.000      0.000
   4227.834895 [0003]     s   postgres[12935]                 0.000      0.000      0.418
   4227.835478 [0003]     s   postgres[25080]                 0.000      0.040      0.583
   4227.836485 [0003]     s   postgres[25080]                 0.000      0.000      1.007
   4227.837482 [0003]     s   postgres[25080]                 0.000      0.000      0.996
   4227.837784 [0003]     s   postgres[25080]                 0.000      0.000      0.302
   4227.837989 [0003]      m    postgres[25080]                                              migrated: :25077 cpu 1 => 3
   4227.837993 [0003]     s   postgres[25080]                 0.000      0.000      0.208
   4227.848487 [0003]     s   postgres[25080]                 0.000      0.000     10.493
   4227.848991 [0003]     s   postgres[25080]                 0.000      0.000      0.504
   4227.849487 [0003]     s   postgres[25080]                 0.000      0.000      0.495
   4227.849748 [0003]     s   postgres[25080]                 0.000      0.000      0.260
   4227.849912 [0003]     s   postgres[25080]                 0.000      0.000      0.164
   4227.851477 [0001]    s    postgres[25082]                 0.000      0.000      0.000
   4227.851481 [0002]     s   postgres[25080]                 0.000      0.000      0.000
   4227.851778 [0003]     s   postgres[12935]                15.017      0.000      1.866
   4227.852259 [0003]      m    postgres[12935]                                              migrated: postgres[25083] cpu 1 => 3
   4227.852263 [0003]     s   postgres[12935]                 0.000      0.000      0.484
   4227.852477 [0003]     s   postgres[25083]                 0.000      0.058      0.214
   4227.852478 [0001]    s    postgres[25082]                 0.000      0.000      1.001
   4227.852614 [0002]     s   postgres[12935]                 0.000      0.000      1.133
```


zalando

Tunables

Tunables

➔ /proc/sys/kernel/sched_migration_cost_ns

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000
- /proc/sys/kernel/sched_wakeup_granularity_ns

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000
- /proc/sys/kernel/sched_wakeup_granularity_ns
- 2000000

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000
- /proc/sys/kernel/sched_wakeup_granularity_ns
- 2000000
- /proc/sys/kernel/sched_min_granularity_ns

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000
- /proc/sys/kernel/sched_wakeup_granularity_ns
- 2000000
- /proc/sys/kernel/sched_min_granularity_ns
- 1500000

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000
- /proc/sys/kernel/sched_wakeup_granularity_ns
- 2000000
- /proc/sys/kernel/sched_min_granularity_ns
- 1500000
- /proc/sys/kernel/sched_latency_ns

Tunables

- /proc/sys/kernel/sched_migration_cost_ns
- 500000
- /proc/sys/kernel/sched_wakeup_granularity_ns
- 2000000
- /proc/sys/kernel/sched_min_granularity_ns
- 1500000
- /proc/sys/kernel/sched_latency_ns
- 12000000

pgbench and pg_dump

```
real    1m38.990s
user    1m9.127s
sys     0m2.066s
```

```
     usecs               : count     distribution
         0 -> 1          : 16       |                                        |
         2 -> 3          : 4604     |**                                      |
         4 -> 7          : 6812     |****                                    |
         8 -> 15         : 14888    |*********                               |
        16 -> 31         : 19267    |***********                             |
        32 -> 63         : 65795    |****************************************|
        64 -> 127        : 50454    |******************************          |
       128 -> 255        : 16393    |*********                               |
       256 -> 511        : 5981     |***                                     |
       512 -> 1023       : 12300    |*******                                 |
      1024 -> 2047       : 48       |                                        |
      2048 -> 4095       : 0        |                                        |
```


zalando

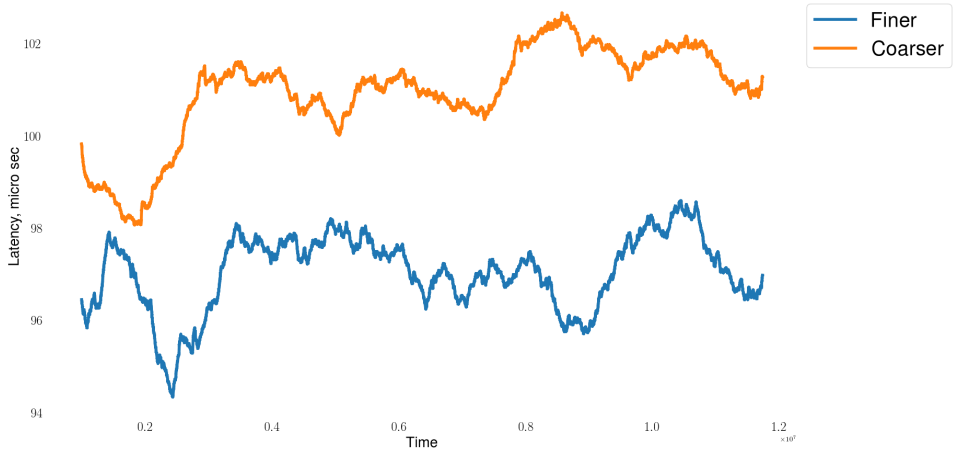
pgbench and pg_dump

```
real    1m32.030s
user    1m8.559s
sys     0m1.641s
```

```
     usecs               : count     distribution
         0 -> 1          : 1       |                                        |
         2 -> 3          : 8       |                                        |
         4 -> 7          : 25      |                                        |
         8 -> 15         : 46      |*                                       |
        16 -> 31         : 189     |*******                                 |
        32 -> 63         : 119     |****                                    |
        64 -> 127        : 96      |***                                     |
       128 -> 255        : 93      |***                                     |
       256 -> 511        : 238     |*********                               |
       512 -> 1023       : 323     |************                            |
      1024 -> 2047       : 1012    |****************************************|
      2048 -> 4095       : 47      |*                                       |
```


zalando

Wakeup granularity, microsec



zalando

CPU hotplug and HyperThreading

Intel® 64 and IA-32 Architectures Optimization Reference Manual

CPU hotplug and HyperThreading

➔ Share execution state and cache

Intel® 64 and IA-32 Architectures Optimization Reference Manual

CPU hotplug and HyperThreading

- ➔ Share execution state and cache
- ➔ Spin locks have significant impact

Intel® 64 and IA-32 Architectures Optimization Reference Manual

CPU hotplug and HyperThreading

- ➔ Share execution state and cache
- ➔ Spin locks have significant impact
- ➔ PAUSE instruction (skylake latency 140 cycles)

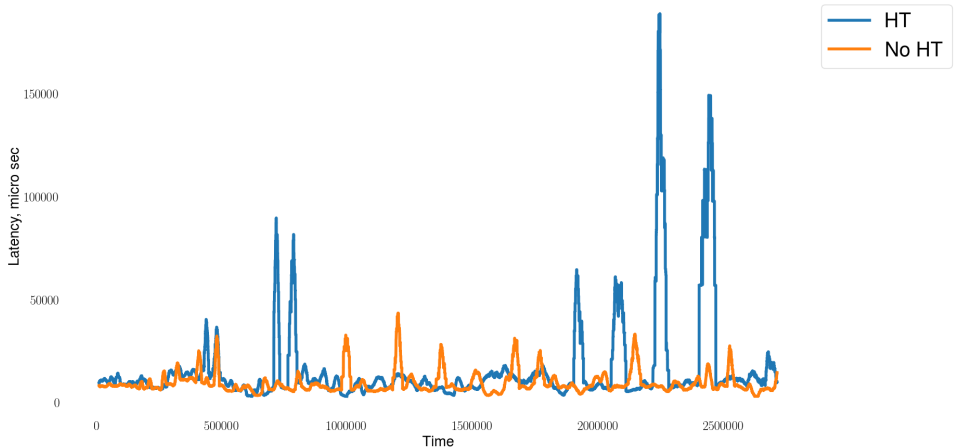
Intel® 64 and IA-32 Architectures Optimization Reference Manual

CPU hotplug and HyperThreading

- ➔ Share execution state and cache
- ➔ Spin locks have significant impact
- ➔ PAUSE instruction (skylake latency 140 cycles)
- ➔ More deviation for latency

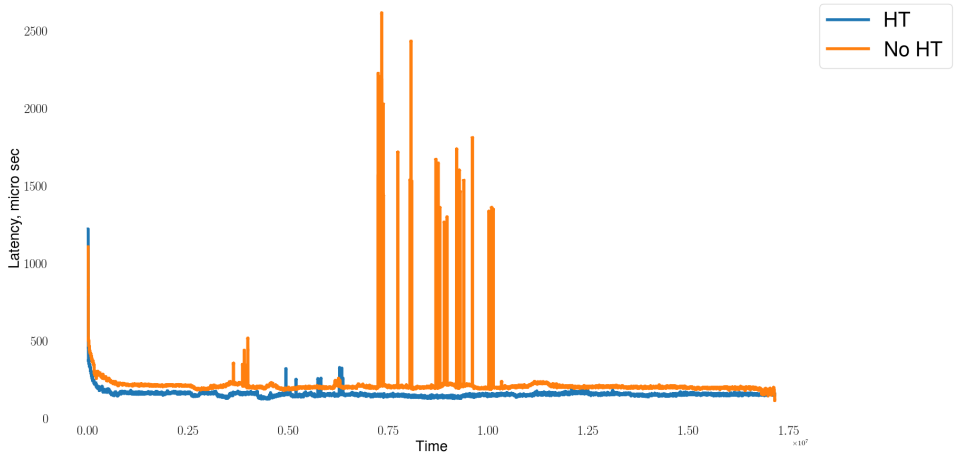
Intel® 64 and IA-32 Architectures Optimization Reference Manual

Latency rolling standard deviation, r/w



zalando

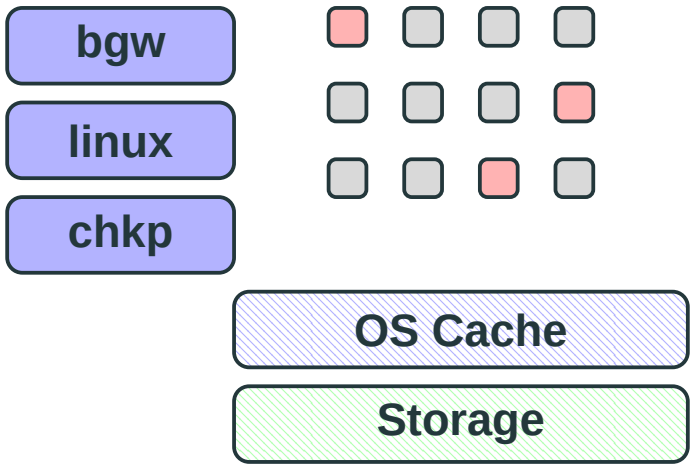
Latency rolling standard deviation, readonly



zalando

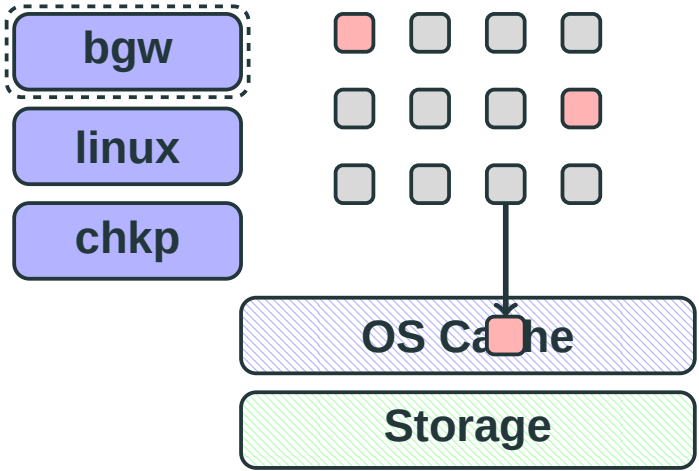
Memory management

Dirty pages



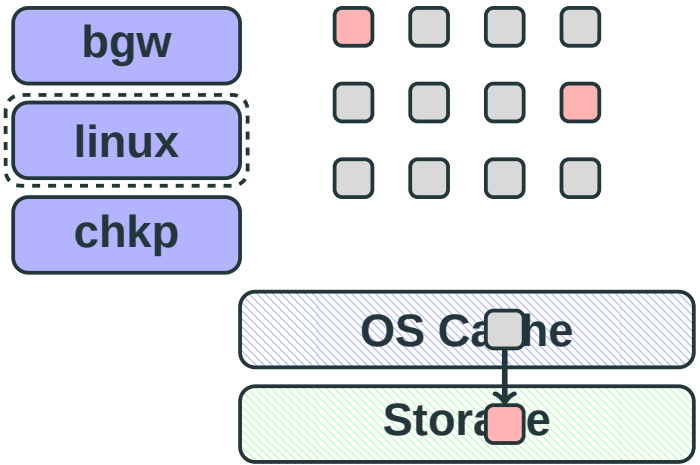
zalando

Dirty pages



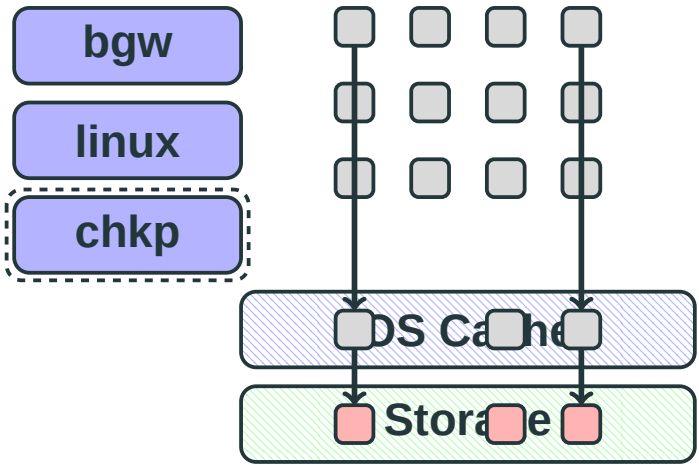
zalando

Dirty pages



zalando

Dirty pages

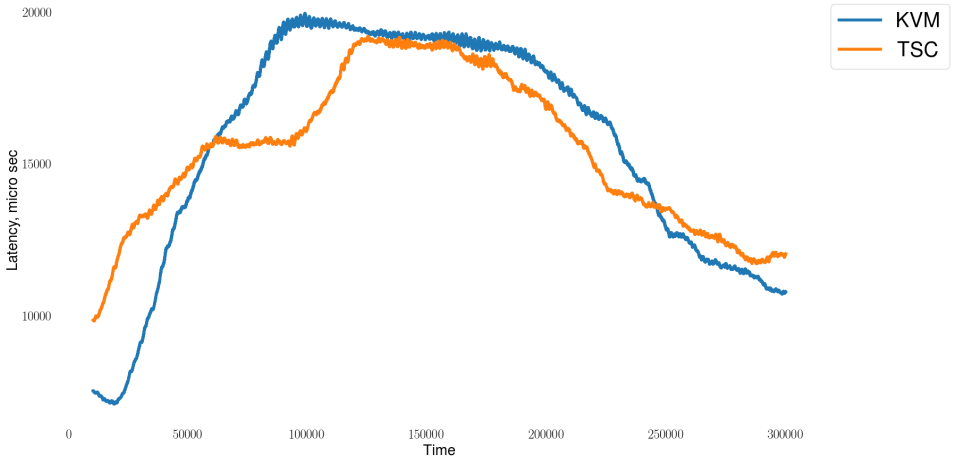


zalando

Dirty pages, r/w

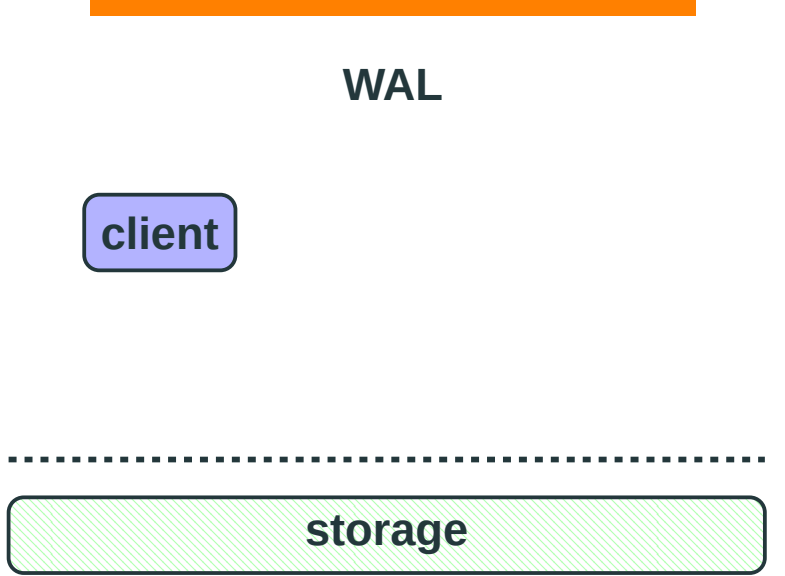
- ➔ vm.dirty_ratio 20
- ➔ vm.dirty_background_ratio 10
- ➔ vm.dirty_bytes 0
- ➔ vm.dirty_backround_bytes 0

Dirty pages



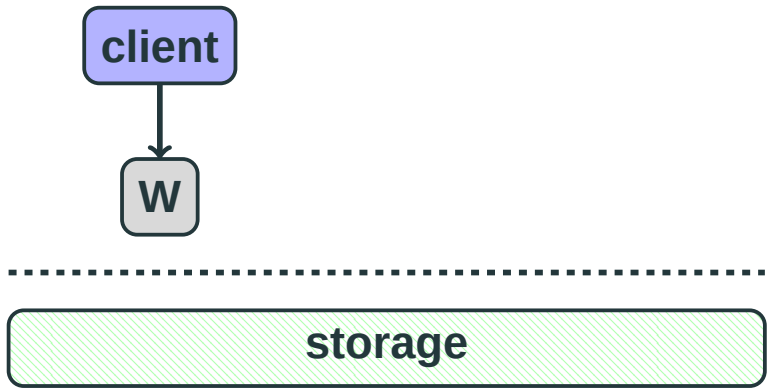
zalando

Storage IO



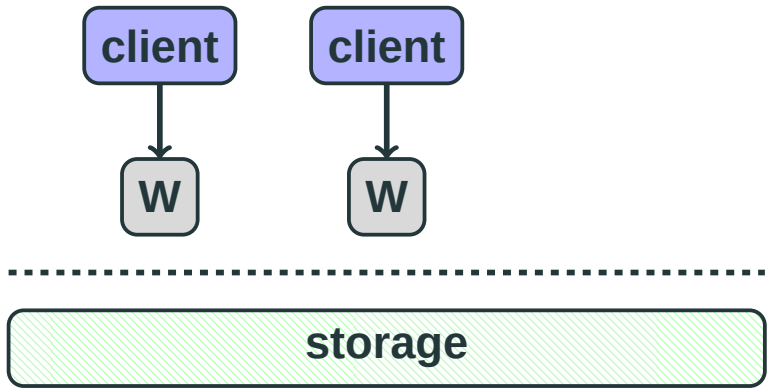
zalando

WAL



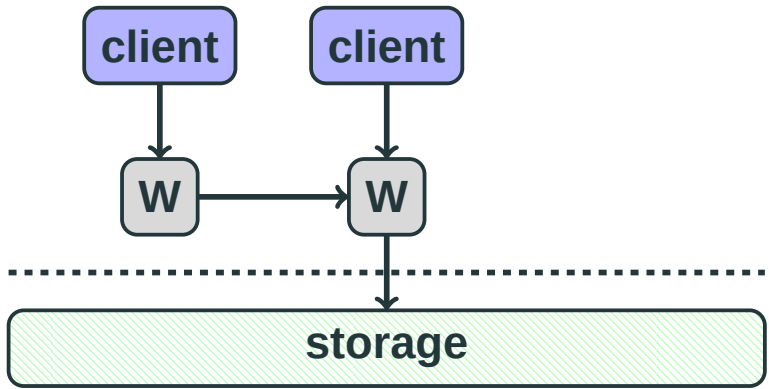
zalando

WAL



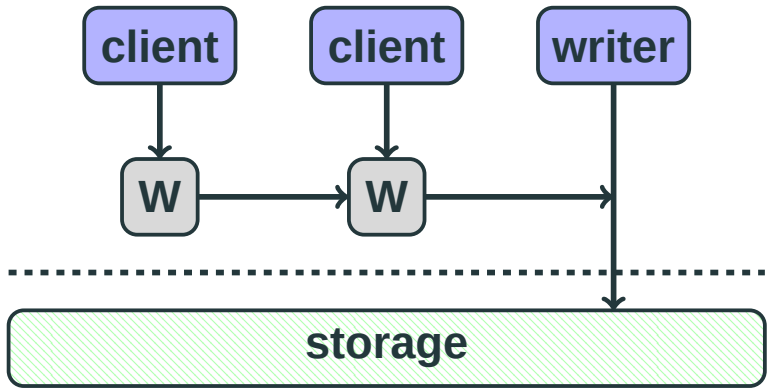
zalando

WAL



zalando

WAL



zalando

WAL

- ➔ bgwriter write to WAl
  (transaction snaphots for replication)
- ➔ with logical decoding turned on
  more data is going to WAL

NVMe

- ➔ better for resourse sharing (PCI express) under the virtualization
- ➔ /sys/block/sda/queue/scheduler [noop|none]
- ➔ DSM operations

NVMe DSM

- ➔ Expected lifetime
- ➔ Prepare for some workload (read/write)
- ➔ Access frequency

DSM support

- ➔ Command DWORD 11 in ioctl
- ➔ fcntl SET_FILE_RW_HINT
- ➔ nvme-cli (ioctl)
- ➔ Specify a start block and a range length

NVM Express Revision 1.3c May 24, 2018

```
# get a start block
hdparm --fibmap data_file
data_file:
 filesystem blocksize 4096, begins at LBA 0;
 assuming 512 byte sectors.
 byte_offset  begin_LBA    end_LBA    sectors
           0   55041560   55041567          8

# set dsm for sequential read optimized
nvme dsm /dev/nvme1n01 --slbs=55041560 --blocks=1 --idr
```

Virtualization

Timekeeping

Timekeeping in VMware Virtual: Information Guide

zalando

Timekeeping

- ➔ Statistical sampling
  (occasional incorrect charging)

Timekeeping in VMware Virtual: Information Guide

Timekeeping

- ➔ Statistical sampling
  (occasional incorrect charging)
- ➔ Exact measurement (TSC time drift)

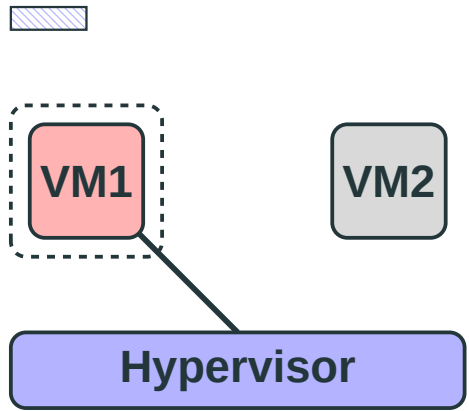
Timekeeping in VMware Virtual: Information Guide

Timekeeping

- ➔ Statistical sampling
  (occasional incorrect charging)
- ➔ Exact measurement (TSC time drift)
- ➔ /sys/devices/system/clocksource/clocksource0/

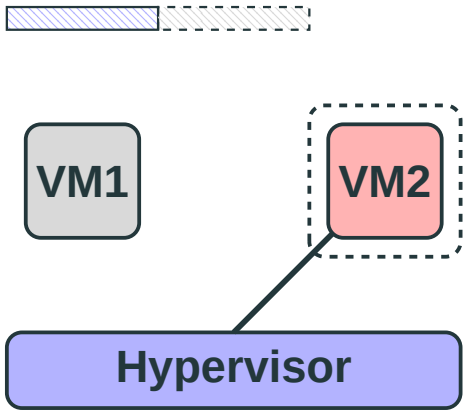
Timekeeping in VMware Virtual: Information Guide

Scheduling



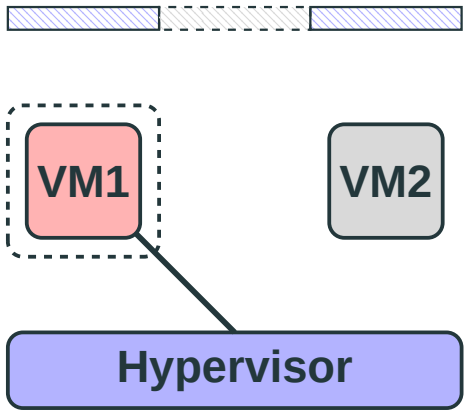
zalando

Scheduling



zalando

Scheduling



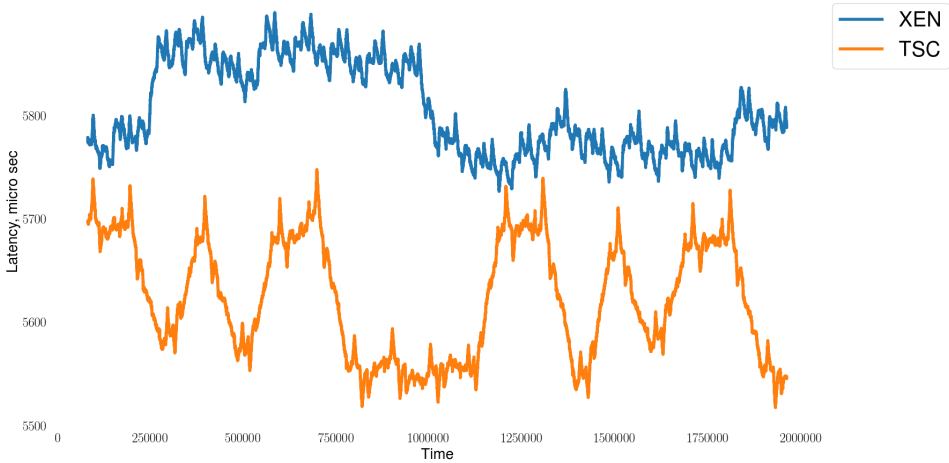
zalando

vDSO

- ➔ gettimeofday
- ➔ clock_gettime
- ➔ XEN doesn’t support vDSO for them
- ➔ unnecessary context switches to a kernel

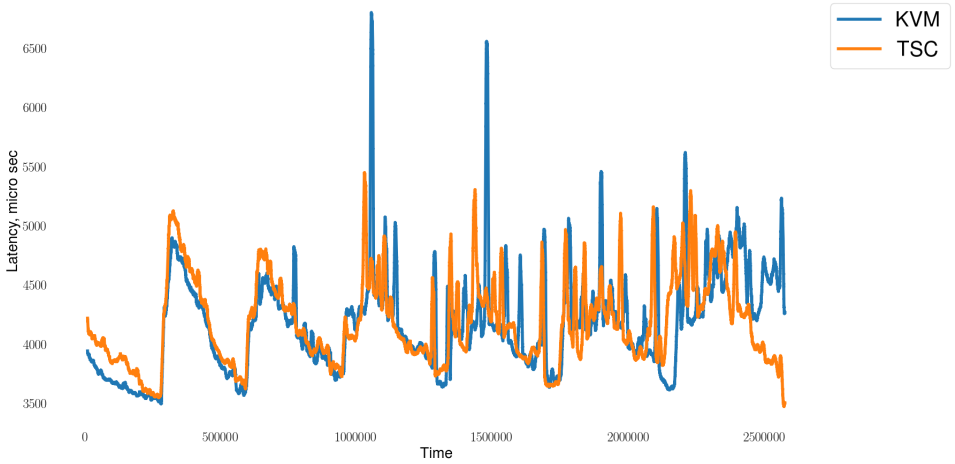
Two frequently used system calls are 77% slower on AWS EC2

Latency m4.xlarge XEN/TSC, r/w



zalando

Latency m5.xlarge KVM/TSC, r/w



zalando

Locks

Intel® 64 and IA-32 Architectures Software Developer’s Manual, Vol. 3

Locks

➔ Lock holder preemption problem

Intel® 64 and IA-32 Architectures Software Developer’s Manual, Vol. 3

Locks

- ➔ Lock holder preemption problem
- ➔ Lock waiter preemption problem

Intel® 64 and IA-32 Architectures Software Developer’s Manual, Vol. 3

Locks

- ➔ Lock holder preemption problem
- ➔ Lock waiter preemption problem
- ➔ Intel PLE (pause loop exiting)

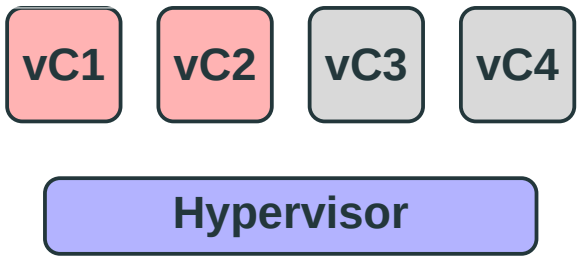
Intel® 64 and IA-32 Architectures Software Developer’s Manual, Vol. 3

Locks

- ➔ Lock holder preemption problem
- ➔ Lock waiter preemption problem
- ➔ Intel PLE (pause loop exiting)
- ➔ PLE_Gap, PLE_Window

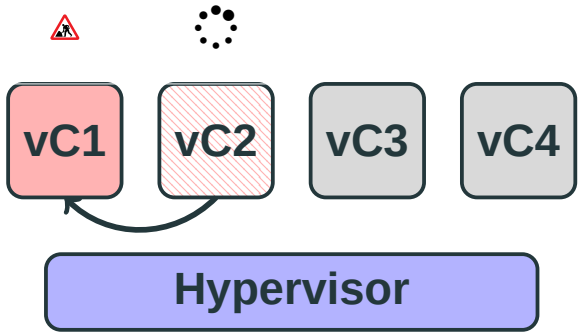
Intel® 64 and IA-32 Architectures Software Developer's Manual, Vol. 3

vCPU



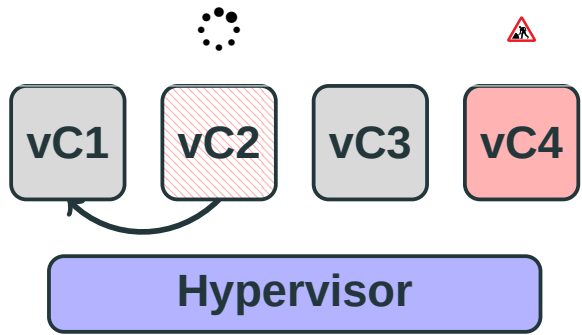
zalando

vCPU



zalando

vCPU



zalando

Containerization

cgroups controllers

- ➔ cpu,cpuacct
- ➔ cpuset
- ➔ memory
- ➔ devices
- ➔ freezer
- ➔ net_cls
- ➔ rdma

- ➔ blkio
- ➔ perf_event
- ➔ net_prio
- ➔ hugetlb
- ➔ pids
- ➔ rdma

```
8388   8388   postgres     blk_throtl_bio
       blk_throtl_bio+0x1        [kernel]
       dm_make_request+0x80      [kernel]
       generic_make_request+0xf6 [kernel]
       submit_bio+0x7d           [kernel]
       blkdev_issue_flush+0x68   [kernel]
       ext4_sync_file+0x310      [kernel]
       vfs_fsync_range+0x4b      [kernel]
       do_fsync+0x3d             [kernel]
       sys_fdatasync+0x13        [kernel]
       fdatasync+0x10            [libc-2.24.so]
       XLogBackgroundFlush+0x17e [postgres]
       WalWriterMain+0x1cb       [postgres]
       PostmasterMain+0xfea      [postgres]
```

bklio controller

- ➔ CFQ & throttling policy (generic block layer)
- ➔ No weight related options will work without CFQ
- ➔ Advisable io scheduler for SSD is noop/none
- ➔ Block layer do sampling to enforce throttling

throttle_sample_time

This is the time window that blk-throttle samples data, in millisecond. blk-throttle makes decision based on the samplings. Lower time means cgroups have more smooth throughput, but higher CPU overhead. This exists only when CONFIG_BLK_DEV_THROTTLING_LOW is enabled.

blkio

On traditional cgroup hierarchies, relationships between different controllers cannot be established making it impossible for writeback to operate accounting for cgroup resource restrictions and all writeback IOs are attributed to the root cgroup.

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git

CONFIG_MEMCG_KMEM

- ➔ Enabled in modern versions
- ➔ PostgreSQL requires contiguous memory for shared buffers
- ➔ It’s being allocated using slab
- ➔ memory.kmem.limit_in_bytes is too high

```
# Host, normal
 Zone: Normal
 Free KiB in zone: 807232.00
    Fragment size        Free fragments
    4096                 29612
    8192                 23308
    16384                13495

# Host with a container
 Zone: Normal
 Free KiB in zone: 109700.00
    Fragment size        Free fragments
    4096                 3405
    8192                 7082
    16384                1954
```


zalando

Bad neighbour

- ➔ buddy allocator can fail
  to find a page of proper size
- ➔ kernel will start a compaction process
- ➔ compaction implementation
  knows nothing about cgroups

Bad neighbour

- ➔ PGSemaphore* funcions make use of futex
- ➔ Per-cpu hash table for futex with hash buckets
- ➔ WAL segment/heap file creation
- ➔ inode lock contention

Understanding Manycore Scalability of File Systems

Questions?

- github.com/erthalion
- github.com/erthalion/ansible-ycsb
- @erthalion
- dmitrii.dolgov at zalando dot de
- 9erthalion6 at gmail dot com

zalando