LinuxDays 10. 10. 2015

PostgreSQL na EXT3/4, XFS, BTRFS a ZFS

srovnání (Linuxových) souborových systémů

Tomáš Vondra <tomas@2ndquadrant.com>

2ndQuadrant
Professional PostgreSQL

22.10. PostgreSQL Meetup @ FIT

27-30.10. pgconf.eu @ Vídeň

cca půlka února P2D2 @ Praha

http://www.slideshare.net/fuzzycz/postgresql-na-ext34-xfs-btrfs-a-zfs

filesystem engineer

~~filesystem engineer~~

database engineer

Který souborový systém mám použít pro produkční server na kterém poběží PostgreSQL?

Podle našich benchmarků z roku 2003 je nejlepší ...

Co vlastně znamená že souborový systém je “stable” nebo “production ready”?

SSD

Souborové systémy

- EXT3/4, XFS, … (spousta dalších)
  - cíli vlastnostmi (features), možnostmi ladění
  - vyzrálostí, spolehlivostí
  - tradiční design, design víceméně z 90. let
  - postupné zlepšování, rozumně “moderní”
- BTRFS, ZFS
  - next-gen souborové systémy, nová architektura / design
- další (nezahrnuté do přednášky)
  - log-organized, distributed, clustered, ...

EXT3, EXT4, XFS, ...

EXT3, EXT4, XFS

- EXT3 (2001) / EXT4 (2008)
  - evoluce původního Linuxového systému (ext, ext2, ...)
  - průběžné zlepšování, opravování chyb, ...
- XFS (2002)
  - původně ze SGI Irix 5.3 (1994)
  - 2000 - uvolněno pod GPL
  - 2002 – zamergováno do 2.5.36
- EXT4 i XFS jsou
  - spolehlivé souborové systémy se žurnálem
  - prověřené časem a provozem na mnoha systémech

EXT3, EXT4, XFS

- tradiční design se žurnálem
- neřeší
  - provoz na více fyzickýc discích
  - logical volume managemnt
  - snapshoty
  - ...
- vyžadují další komponenty
  - hardware RAID
  - software RAID (dm)
  - LVM / LVM2

EXT3, EXT4, XFS

- počaty v dobách rotačních disků
  - víceméně fungují se SSD
  - stop-gap pro budoucí úložné systémy (NVRAM, ...)
- víceméně evoluce, nikoliv revoluce
  - doplňování vlastností (e.g. TRIM, write barriers, ...)
  - vylepšování škálování (metadata, ...)
  - opravování chyb
- nutná značná opatrnost kvůli
  - zastaralým výsledkům benchmarků a povídačkám
  - zavádějícím syntetickým benchmarkům

EXT3, EXT4, XFS

- Linux Filesystems: Where did they come from?
  (Dave Chinner @ linux.conf.au 2014)
  https://www.youtube.com/watch?v=SMcVdZk7wV8
- Ted Ts'o on the ext4 Filesystem
  (Ted Ts'o, NYLUG, 2013)
  https://www.youtube.com/watch?v=2mYDFr5T4tY
- XFS: There and Back … and There Again?
  (Dave Chinner @ Vault 2015)
  https://lwn.net/Articles/638546/
- XFS: Recent and Future Adventures in Filesystem Scalability
  (Dave Chinner, linux.conf.au 2012)
  https://www.youtube.com/watch?v=FegjLbCnoBw
- XFS: the filesystem of the future?
  (Jonathan Corbet, Dave Chinner, LWN, 2012)
  http://lwn.net/Articles/476263/

BTRFS, ZFS

BTRFS, ZFS

- základní myšlenky
  - integrujme vrstvy (LVM + dm + ...)
  - návrh zaměřený na běžný hardware (chyby jsou běžné)
  - návrh pro velké datové objemy, flexibilitu
- čímž jednodušeji získáme …
  - flexibilnější management
  - zabudovaný snapshotting
  - kompresi, deduplikaci
  - checksums

BTRFS, ZFS

- BTRFS
  - zamergováno 2009, ale nadále “experimental”
  - on-disk format “stable” (1.0)
  - někteří tvrdí že “stable” nebo “production ready” ...
  - některé distribuce ho dnes používají jako výchozí volbu
- ZFS
  - původně ze Solarisu, ale “got Oracled” :-(
  - dnes poněkud fragmentovaný vývoj
  - dostupné i na dalších BSD systémech (FreeBSD)
  - “ZFS on Linux” projekt (ale CDDL vs. GPL)

Možnosti ladění

Obecné možnosti

- TRIM (discard)
  - zapnutí / vypnutí TRIM na SSDs
  - ovlivňuje interní “garbage collection” / wear leveling
  - není nutný, ale může pomoci SSD disku při “garbage collection”
- write barriers
  - zabraňují disku v optimalizaci práce změnou pořadí zápisů
  - nebrání ztrátě dat ale poškození konzistence (metadata vs. data)
  - write cache + battery => write barriers lze vypnout
- SSD alignment

BTRFS

- nodatacow
  - vypne “copy on write” (CoW)
  - snapshoty CoW dočasně zapnou
  - vypne checksums (vyžaduje “plný” CoW)
- ssd
  - zapne SSD optimalizace
  - dokumentace neříká jaké
- compress=lzo/zlib
  - spekulativní komprese dat

ZFS

- recordsize=8kB
  - standardní velikost stránky ZFS je 128kB
  - PostgreSQL má datové stránky o velikosti 8kB
  - problémy při cachování v ARC (menší počet “slotů”)
- logbias=throughput
  - ovlivňuje práci se ZIL (latence vs. throughput)
- zfs_arc_max
  - omezení velikosti ARC cache
  - měla by se uvolňovat automaticky, ale ...
- primarycache=metadata
  - zabránění double bufferingu (shared buffers vs. ARC)

ZFS

- recordsize=8kB
  - standardní velikost stránky ZFS je 128kB
  - PostgreSQL má datové stránky o velikosti 8kB
  - problémy při cachování v ARC (menší počet “slotů”)
- logbias=throughput
  - ovlivňuje práci se ZIL (latence vs. throughput)
- zfs_arc_max
  - omezení velikosti ARC cache
  - měla by se uvolňovat automaticky, ale ...
- ~~primarycache=metadata~~
  - ~~zabránění double bufferingu (shared buffers vs. ARC)~~

Benchmark

Použitý systém

- CPU: Intel i5-2500k
  - 4 cores @ 3.3 GHz (3.7GHz)
  - 6MB cache
  - 2011-2013
- 8GB RAM (DDR3 1333)
- SSD Intel S3700 100GB (SATA3)
- Gentoo + kernel 4.0.4
- PostgreSQL 9.4

pgbench (TPC-B)

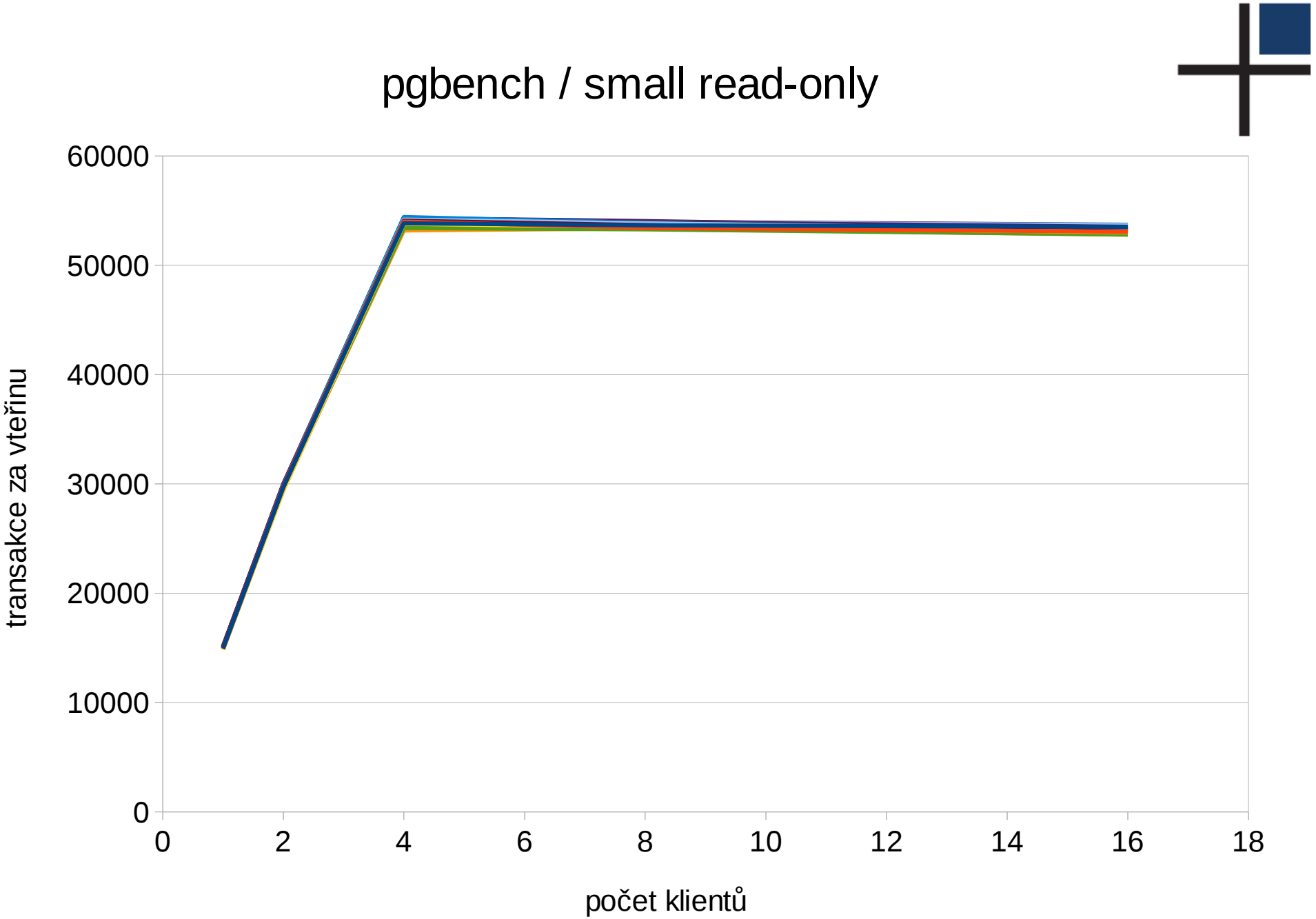
- transakční benchmark / stress-test
  - malé dotazy (přístup přes PK, ...)
  - mix různých typů I/O (reads/writes, náhodný/sekvenční)
- varianty
  - read-write (SELECT + INSERT + UPDATE)
  - read-only (SELECT)
- objemy dat
  - malý (~200MB)
  - střední (~50% RAM)
  - velký (~200% RAM)

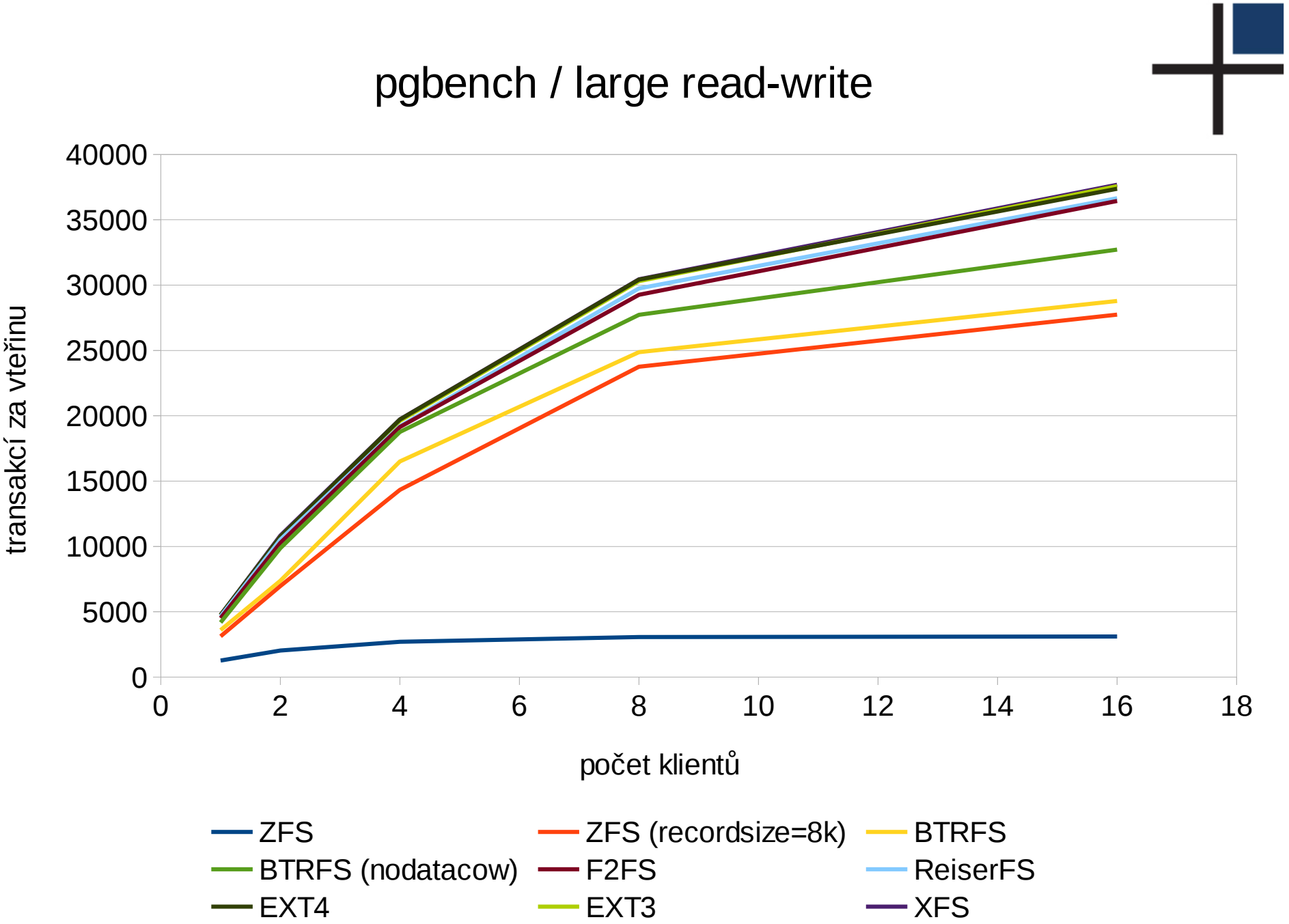
Výsledky

- zatím otestováno přes 40 kombinací
- každý test běží ~4 dny

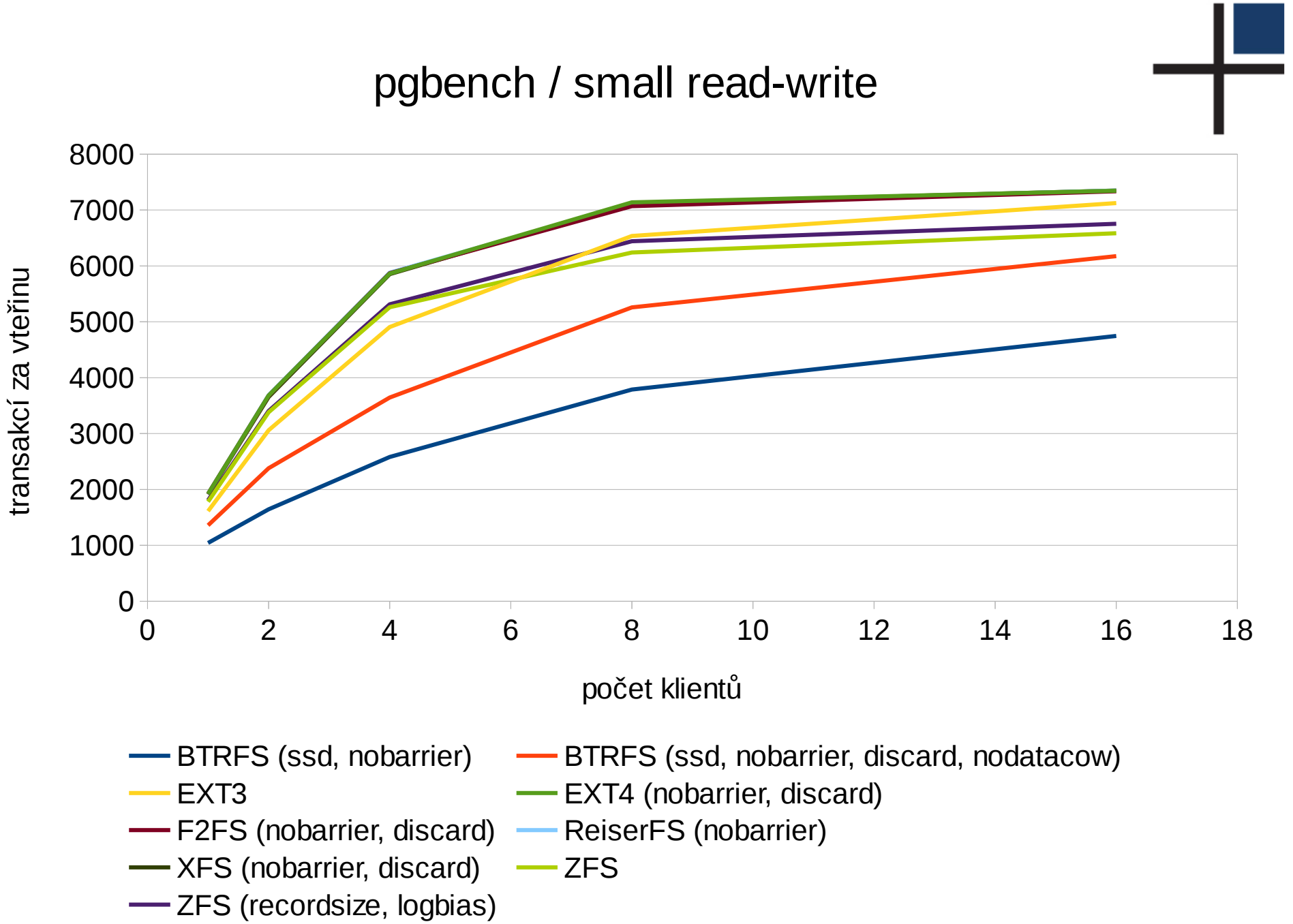
https://bitbucket.org/tvondra/fsbench-i5

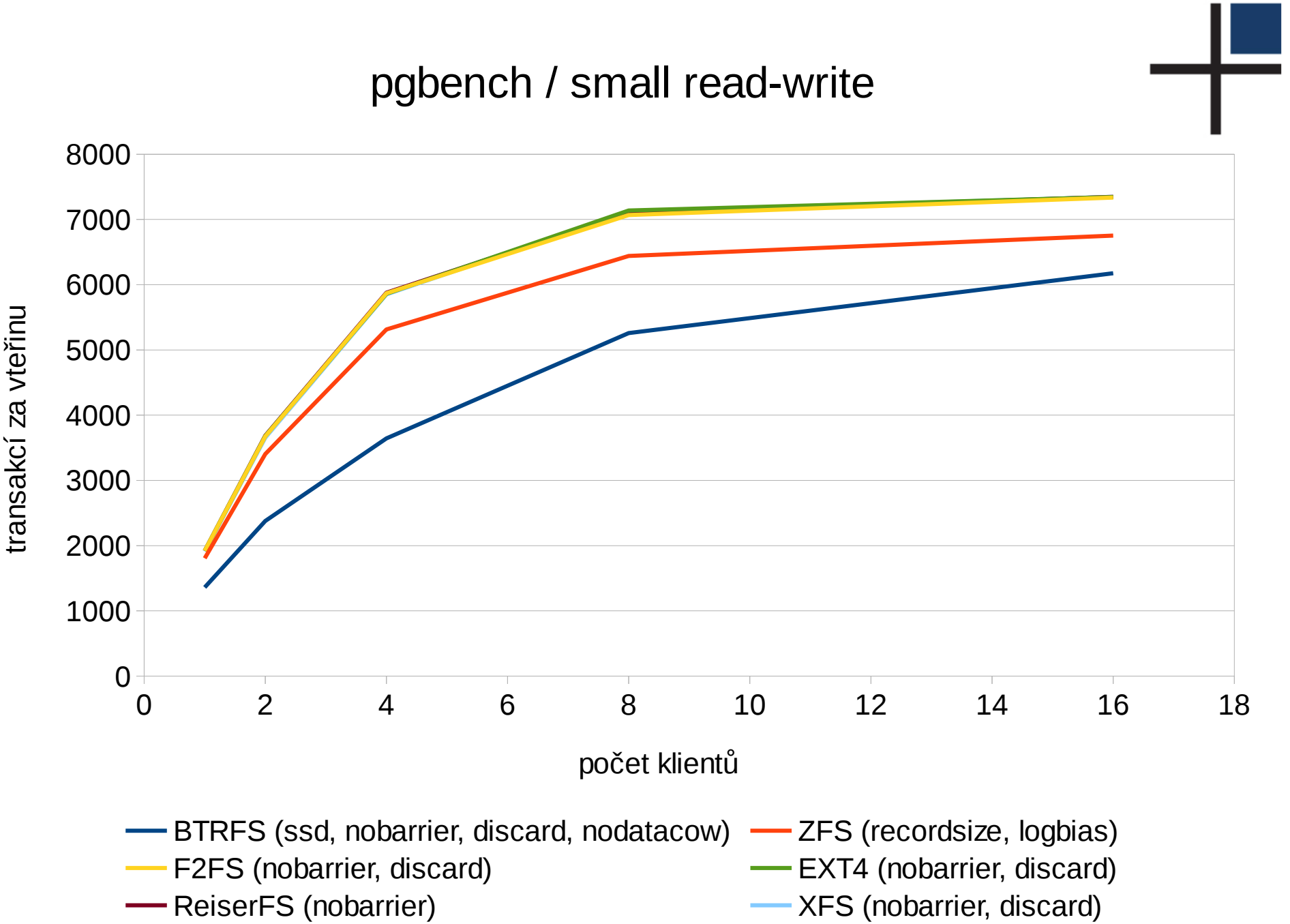
pgbench read-only

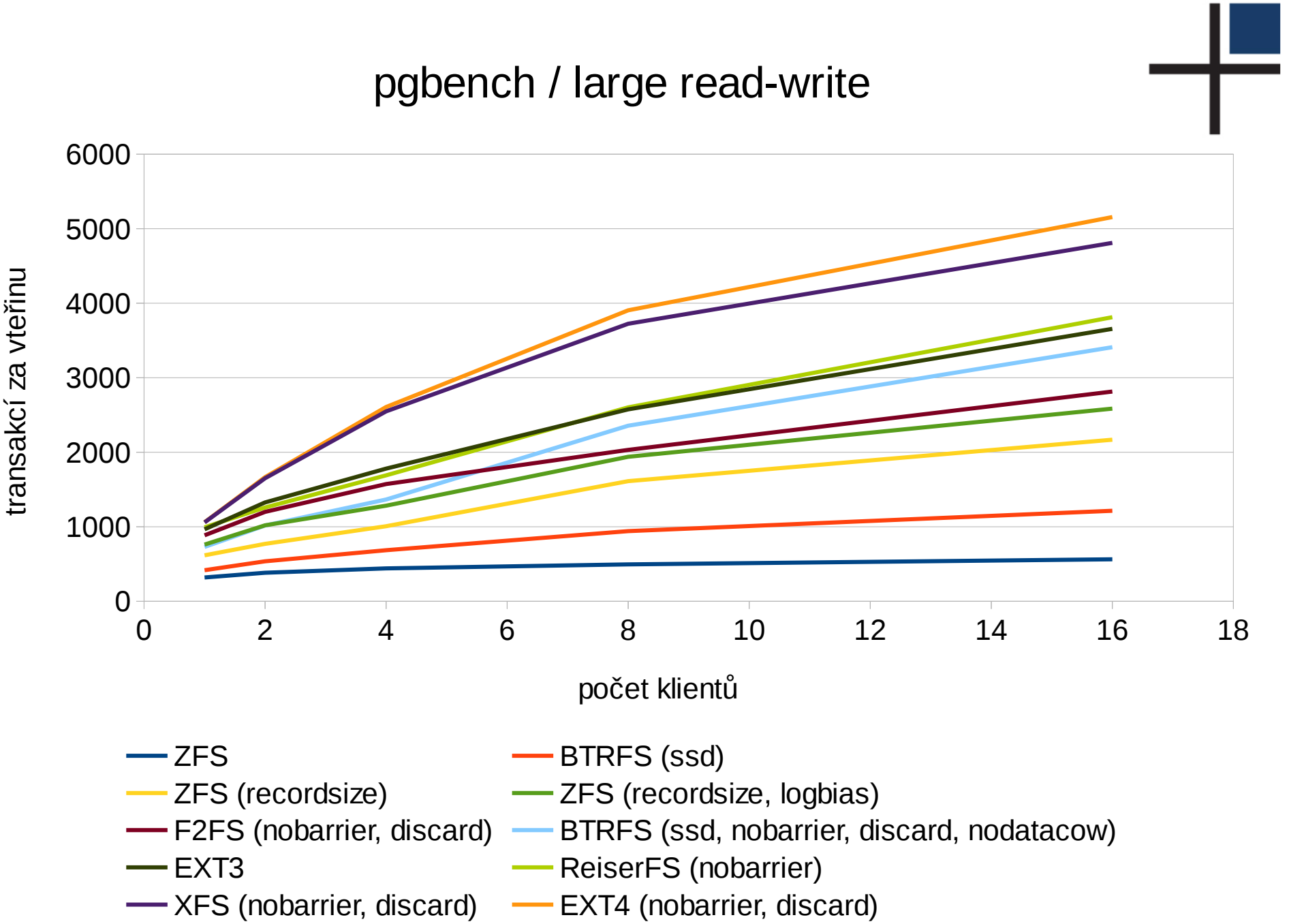

pgbench / small read-only
transakce za vteřinu
60000
50000
40000
30000
20000
10000
0
0
2
4
6
8
10
12
14
16
18
počet klientů

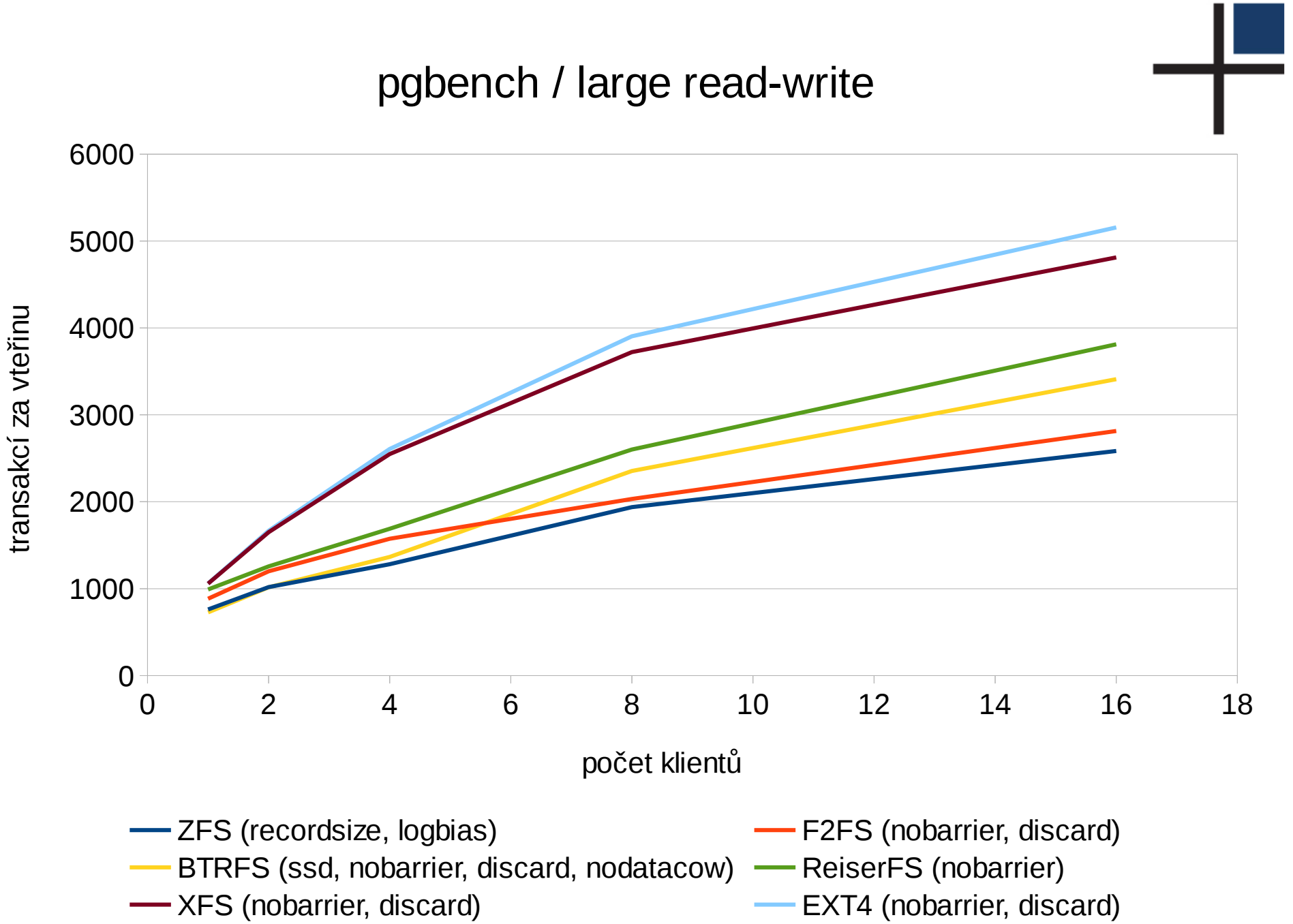

pgbench / large read-write
transakcí za vteřinu
40000
35000
30000
25000
20000
15000
10000
5000
0
0
2
4
6
8
10
12
14
16
18
počet klientů
ZFS
ZFS (recordsize=8k)
BTRFS
BTRFS (nodatacow)
F2FS
ReiserFS
EXT4
EXT3
XFS

pgbench read-write


pgbench / small read-write
transakcí za vteřinu
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
18
počet klientů
BTRFS (ssd, nobarrier)
BTRFS (ssd, nobarrier, discard, nodatacow)
EXT3
EXT4 (nobarrier, discard)
F2FS (nobarrier, discard)
ReiserFS (nobarrier)
XFS (nobarrier, discard)
ZFS
ZFS (recordsize, logbias)

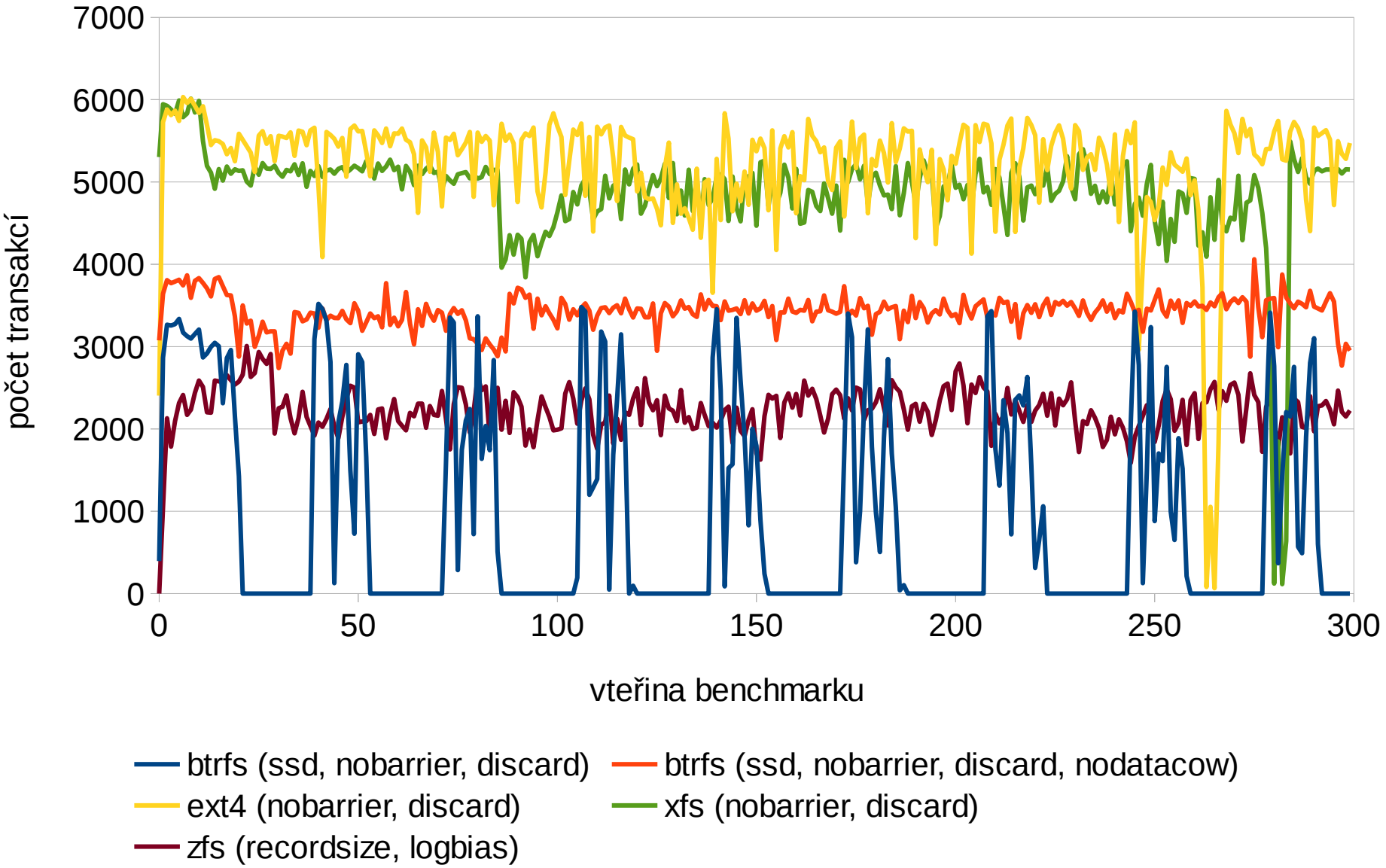

pgbench / small read-write
transakcí za vteřinu
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
18
počet klientů
BTRFS (ssd, nobarrier, discard, nodatacow)
ZFS (recordsize, logbias)
F2FS (nobarrier, discard)
EXT4 (nobarrier, discard)
ReiserFS (nobarrier)
XFS (nobarrier, discard)


pgbench / large read-write
transakcí za vteřinu
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
18
počet klientů
ZFS
BTRFS (ssd)
ZFS (recordsize)
ZFS (recordsize, logbias)
F2FS (nobarrier, discard)
BTRFS (ssd, nobarrier, discard, nodatacow)
EXT3
ReiserFS (nobarrier)
XFS (nobarrier, discard)
EXT4 (nobarrier, discard)


pgbench / large read-write
transakcí za vteřinu
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
18
počet klientů
ZFS (recordsize, logbias)
F2FS (nobarrier, discard)
BTRFS (ssd, nobarrier, discard, nodatacow)
ReiserFS (nobarrier)
XFS (nobarrier, discard)
EXT4 (nobarrier, discard)

variabilita

pgbench po vteřinách



EXT / XFS

- víceméně stejné výsledky
  - EXT4 – vyšší propustnost, vyšší rozptyl latencí
  - XFS – mírně nižší propustnost, nižší rozptyl
- zásadní vliv “write barriers”
  - nutností jsou dobré disky / RAID řadiče
- menší vliv TRIM
  - záleží na typu SSD (over-provisioning)
  - záleží na zaplnění disku

BTRFS, ZFS

- cena za CoW je značná
  - cca 50% redukce výkonu (cena za funkce)
- ZFS
  - trochu vetřelec ve světě Linuxu
  - značně vyzrálejší než BTRFS, lepší chování
- BTRFS
  - všechny problémy během testování byly s BTRFS
  - značně nestabilní a nekonzistentní chování
  - žádné data corruption bugy (good)

BTRFS, ZFS

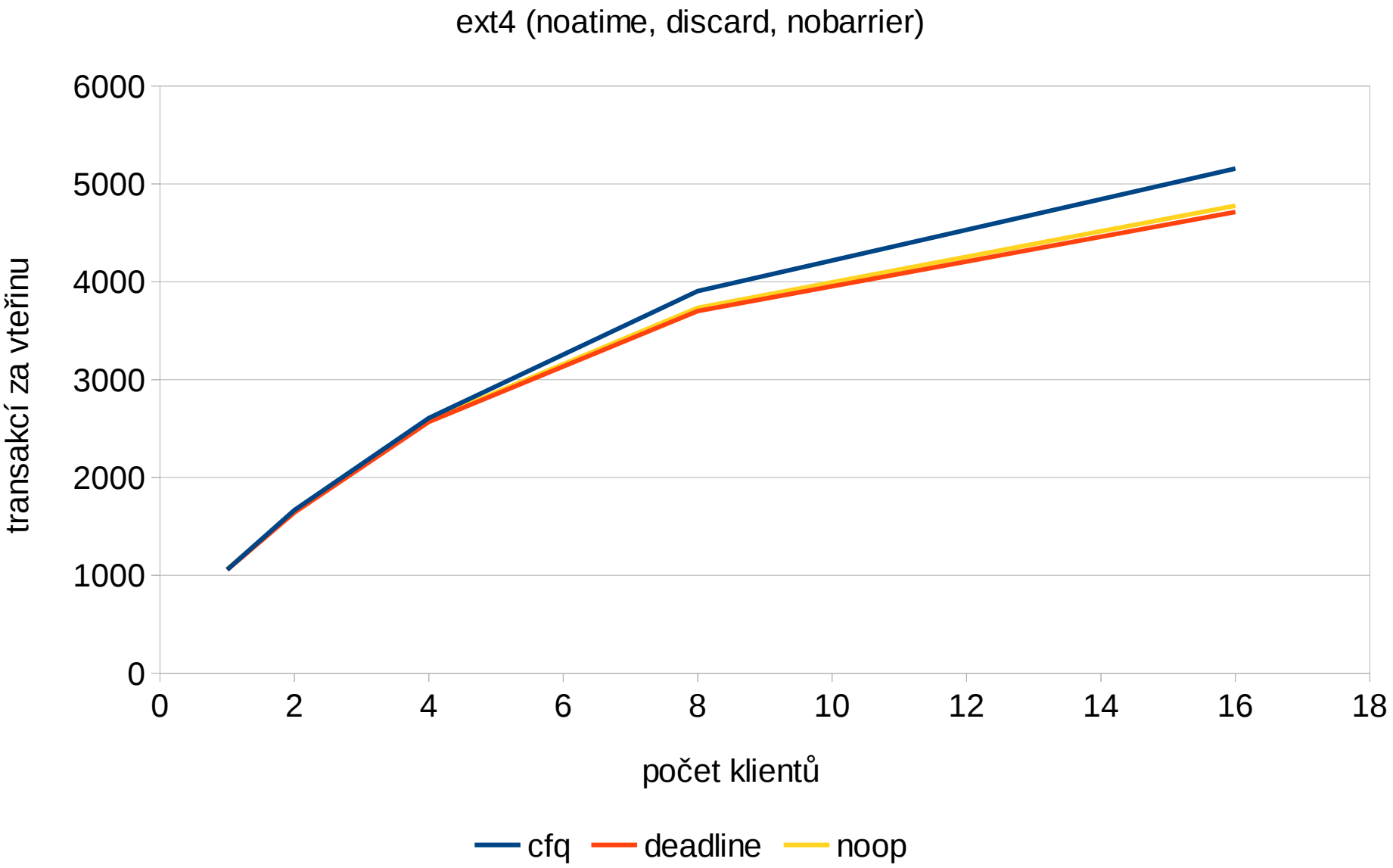
```
Tasks: 215 total,   2 running, 213 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us, 12.6%sy,  0.0%ni, 87.4%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  16432096k total, 16154512k used,   277584k free,     9712k buffers
Swap:  2047996k total,    22228k used,  2025768k free, 15233824k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
24402 root      20   0     0    0    0 R 99.7  0.0   2:28.09 kworker/u16:2
24051 root      20   0     0    0    0 S  0.3  0.0   0:02.91 kworker/5:0
    1 root      20   0 19416  608  508 S  0.0  0.0   0:01.02 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:09.10 kthreadd
    ...

Samples: 59K of event 'cpu-clock', Event count (approx.): 10269077465
Overhead  Shared Object        Symbol
  37.47%  [kernel]             [k] btrfs_bitmap_cluster
  30.59%  [kernel]             [k] find_next_zero_bit
  26.74%  [kernel]             [k] find_next_bit
   1.59%  [kernel]             [k] _raw_spin_unlock_irqrestore
   0.41%  [kernel]             [k] rb_next
   0.33%  [kernel]             [k] tick_nohz_idle_exit
   ...
```

Otázky?

pgbench / large read-write



pgbench / large read-write (16 clients)