MongoDB on IBM Z

Performance Tuning Guide

Marc Beyerle (marc.beyerle@de.ibm.com)
Senior Java Performance Engineer, IBM Mainframe Specialist

Document version: 1.0
Document date: 2026-05-18

Notices and disclaimers

U.S. Government Users Restricted Rights – use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed "as is" without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.

IBM products are manufactured from new parts or new and used parts. In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply.

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Notices and disclaimers, *continued*

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com and FICON, IBM FlashSystem, IBM Z, IBM z16, and z17 are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names and logos might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml

Agenda

- Introduction to MongoDB
- Performance measurement and tuning approach
- Observations and recommendations
- Summary

About MongoDB

- Quote from the mongodb.com website: *"MongoDB® is a document database with the scalability and flexibility that you want with the querying and indexing that you need"*

- Most popular ***NoSQL database*** according to db-engines.com:

434 systems in ranking, May 2026

Rank			DBMS	Database Model	Score		
May 2026	Apr 2026	May 2025			May 2026	Apr 2026	May 2025
1.	1.	1.	Oracle	Relational, Multi-model	1143.28	-14.66	-83.29
2.	2.	2.	MySQL	Relational, Multi-model	856.49	-1.20	-108.49
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	700.99	-1.08	-73.90
4.	4.	4.	PostgreSQL	Relational, Multi-model	682.68	+1.33	+8.37
5.	5.	5.	MongoDB	Document, Multi-model	384.65	-0.37	-17.87

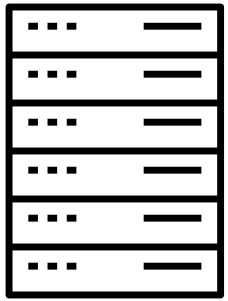
About MongoDB, *continued*

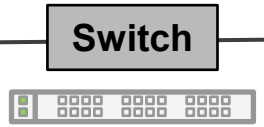
- As part of our ***Then & Now*** study that captures IBM Z® / IBM® LinuxONE generation-to-generation performance improvements for select Linux® workloads, we discovered that MongoDB is one of the "most loved" workloads on the platform
  - Conducted a series of surveys, both locally in the DACH region and worldwide
- Our team has performed a wide range of performance-related studies with MongoDB in the past, for example ***competitive comparisons*** against Intel® x86
  - For details, see the official IBM z16® Proof Points slide deck
- For this tuning guide, we analyzed MongoDB very thoroughly regarding ***possible tuning opportunities*** for the very first time
  - Found more tuning knobs than initially anticipated

Agenda

- Introduction to MongoDB
- Performance measurement and tuning approach
- Observations and recommendations
- Summary

High-level environment setup

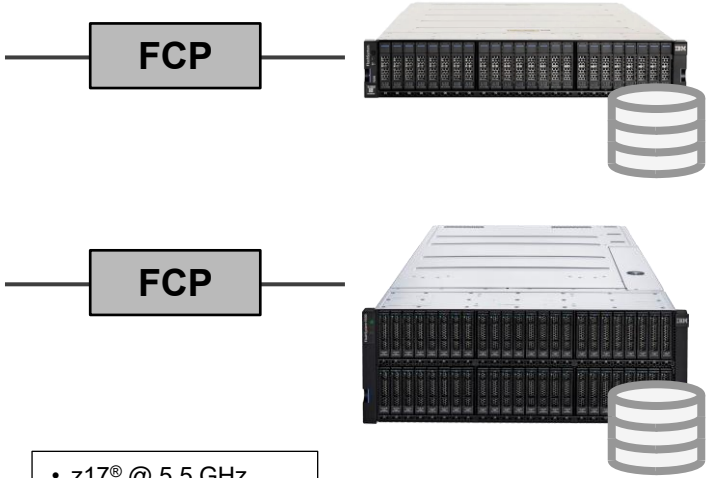




***Yahoo!*® Cloud Serving Benchmark (YCSB)**

- Intel x86, Ice Lake generation (Intel Xeon® Gold 6326 CPU @ 2.90 GHz)
- 32 CPUs, Hyper-Threading
- 512 GiB mem
- 1 x Mellanox® Technologies MT2894 Family [ConnectX®-6]
- 1 x Mellanox Technologies MT2892 Family [ConnectX-6]

- z17® @ 5.5 GHz
- 32 IFLs, SMT (24+4+4)
- 3 x 128 GiB mem
- 3 x IBM 3390 Model 54 ECKD DASD for root filesystem(s)
- 3 x Network Express 25 GbE SR
- 6 x FICON® Express 32G-4P SX



- IBM FlashSystem® 9200
- 8 x 100 GiB SCSI LUNs for MongoDB data

- IBM FlashSystem 9500
- 4 x 100 GiB SCSI LUNs for MongoDB data

Approach

- Typically, I start with a small ***sandbox environment***, in order to get familiar with the individual software components (OS, database management system, load driver) and experiment with parameters and settings
  - Same approach for this study: started with a small configuration (4 IFLs for the leader node) and grew this environment over time – first 12 IFLs, then 24 IFLs
- Next, I put the *System Under Test* (SUT) under load, ***analyze*** the runtime behavior of the workload, and try to (a) ***eliminate*** the ***bottlenecks*** that I encounter and (b) ***reduce*** the ***idle time*** left in the system
- This time, I also tried to make use of ***AI tooling*** in order to optimize throughput
  - Described my observations using prompts, answered additional questions that the AI tools had
  - ***None of the AI tools*** that I tried was able to give recommendations that had a measurable impact on performance (throughput, idle time, etc.)

Approach, *continued*

- In this study, I found some of the tuning knobs quite late in the analysis / experimentation cycle, because I entered ***partially uncharted territory***
  - See the tuning tips for details
- Started with quite a ***decent baseline configuration***
  - 3 x Network Express adapters, 2 x IBM FlashSystem storage servers with NVMe®-based disks, etc.
- Put an extra focus on the ***workload driver environment*** to make sure it's not a bottleneck
  - Ice Lake generation Intel x86 hardware
  - Ran several YCSB instances (8 for the 24 IFL config) with 64 threads each

Agenda

- Introduction to MongoDB
- Performance measurement and tuning approach
- Observations and recommendations
- Summary

Disclaimers

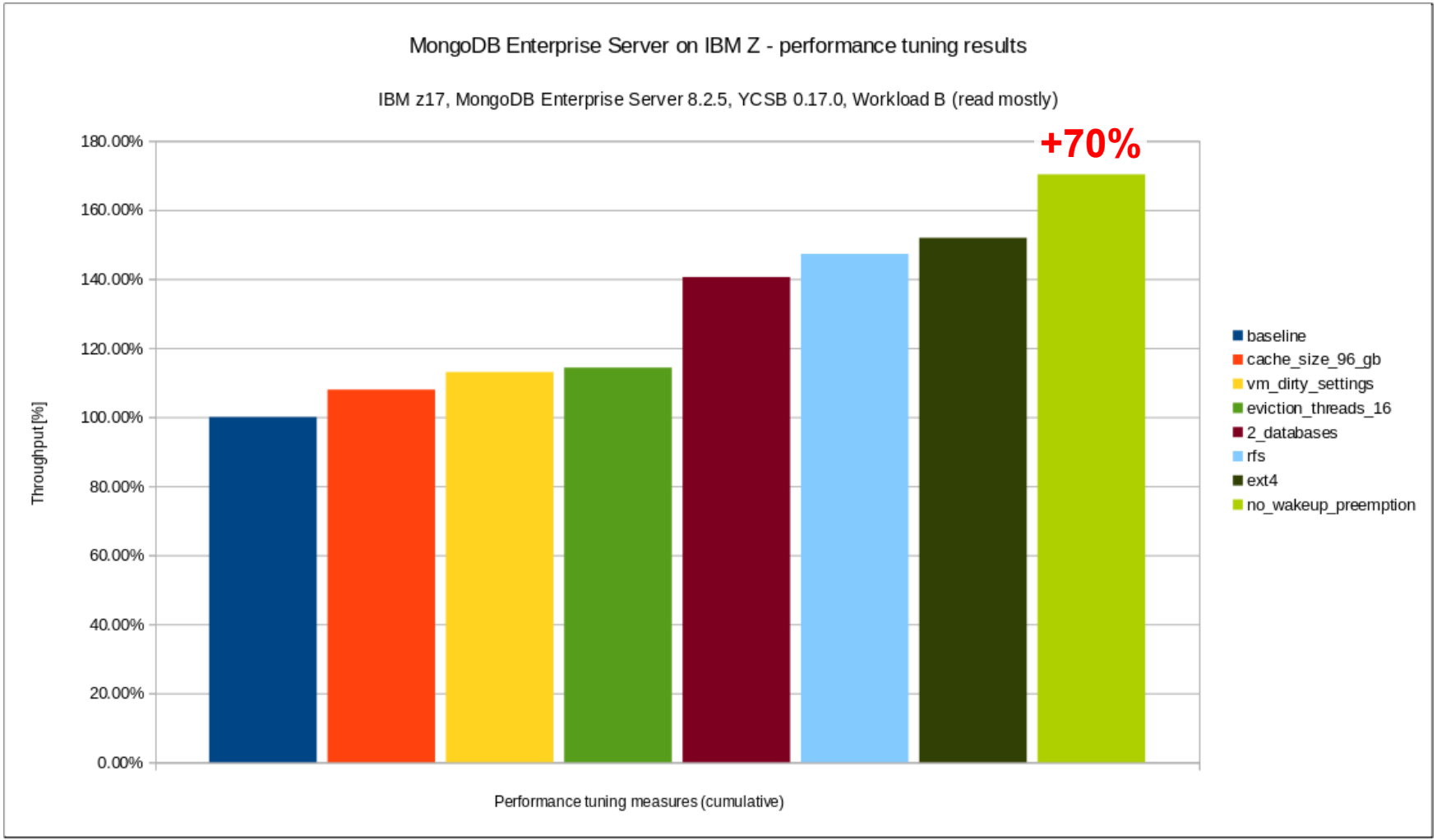
The following is ***important*** – please read it carefully

- The performance test results in the following charts were obtained in a ***controlled lab environment*** natively in *Logical Partitions* (LPARs). The measured differences in throughput might not be observed in real-life scenarios and environments other than native LPAR.

- All of the test runs were performed with Red Hat® Enterprise Linux® 10, MongoDB Enterprise Server 8.2.5, and YCSB 0.17.0. ***Other*** product versions might produce ***different*** performance results.

- All of the tests were specifically executed for ***MongoDB***. The impact of the recommendations in this chart deck on ***other*** database management systems might be ***totally different***, including ***adverse*** performance effects.

- All of the tests were specifically executed for a ***read mostly*** workload. The impact of the recommendations in this chart deck on other types of workloads – update heavy, for example – might be ***totally different***, including ***adverse*** performance effects.


MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
+70%
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Throughput [%]
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

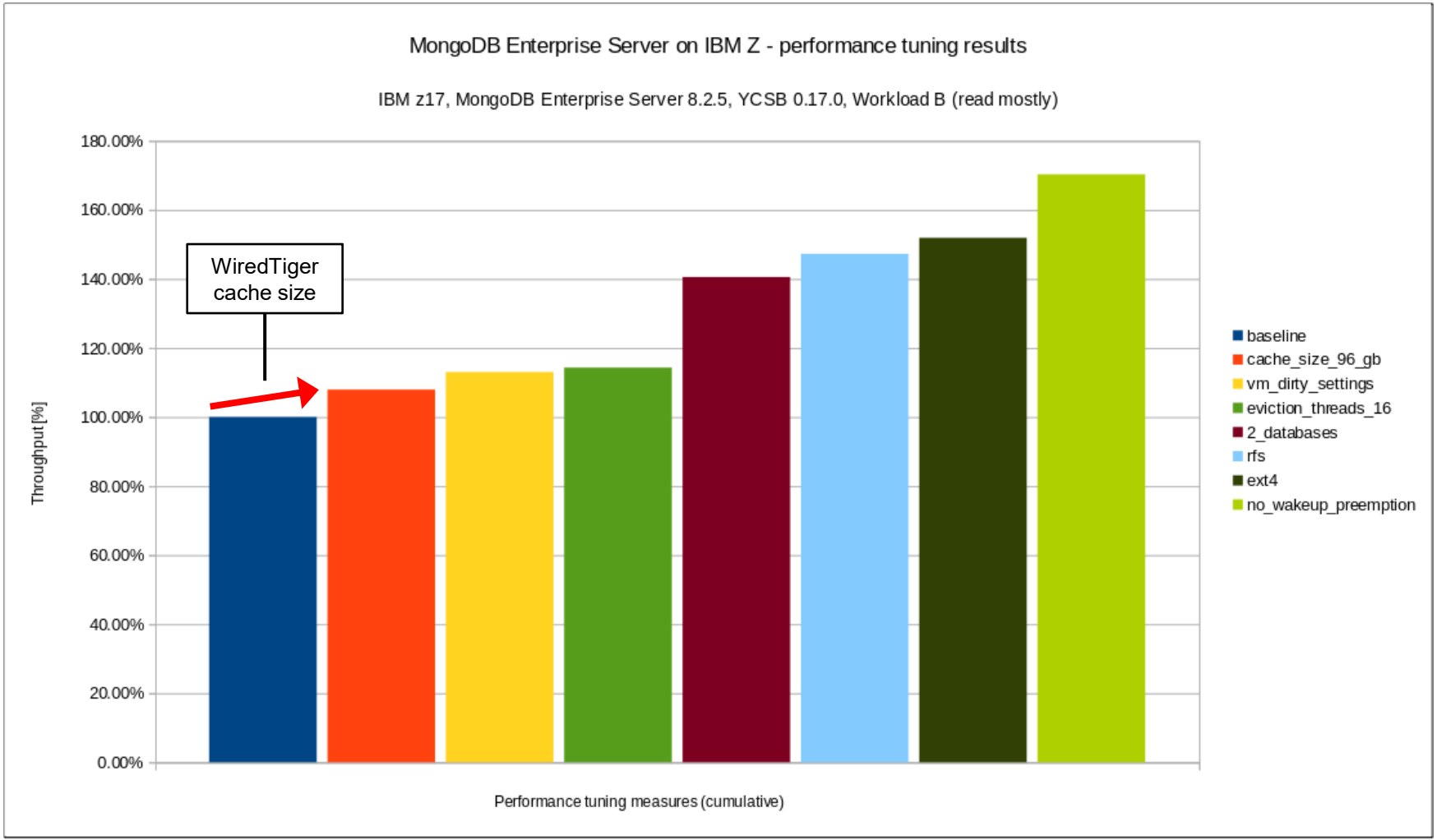
Recommendation #1

- Recommendation #1: increase the WiredTiger ***internal cache size***
- WiredTiger is the default ***storage engine*** in MongoDB
- The ***default*** WiredTiger internal cache size is the larger of either 50% of (memory – 1 GiB) or 256 MiB
  - For my 128 GiB Linux images, this corresponds to (128 GiB – 1 GiB) / 2 = 63.5 GiB
- MongoDB manual: *"You must specify a percentage of **up to 80%** of available memory."*
  - MongoDB will issue a ***warning*** in the log when you specify more than 80%, but it still starts up
- ***In my tests***, optimal setting turned out to be ***75%*** of 128 GiB memory: 96 GiB
  - Determined this number by ***empirical testing***

Recommendation #1, *continued*

- In order to configure the WiredTiger cache size for MongoDB, add the following line ***in blue*** to your `mongod.conf` configuration file:

```
...
storage:
   dbPath: /data
   directoryPerDB: true
   engine: "wiredTiger"
   wiredTiger:
      ...
      engineConfig:
         cacheSizeGB: 96
...
```


MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
WiredTiger cache size
Throughput [%]
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

Recommendation #2

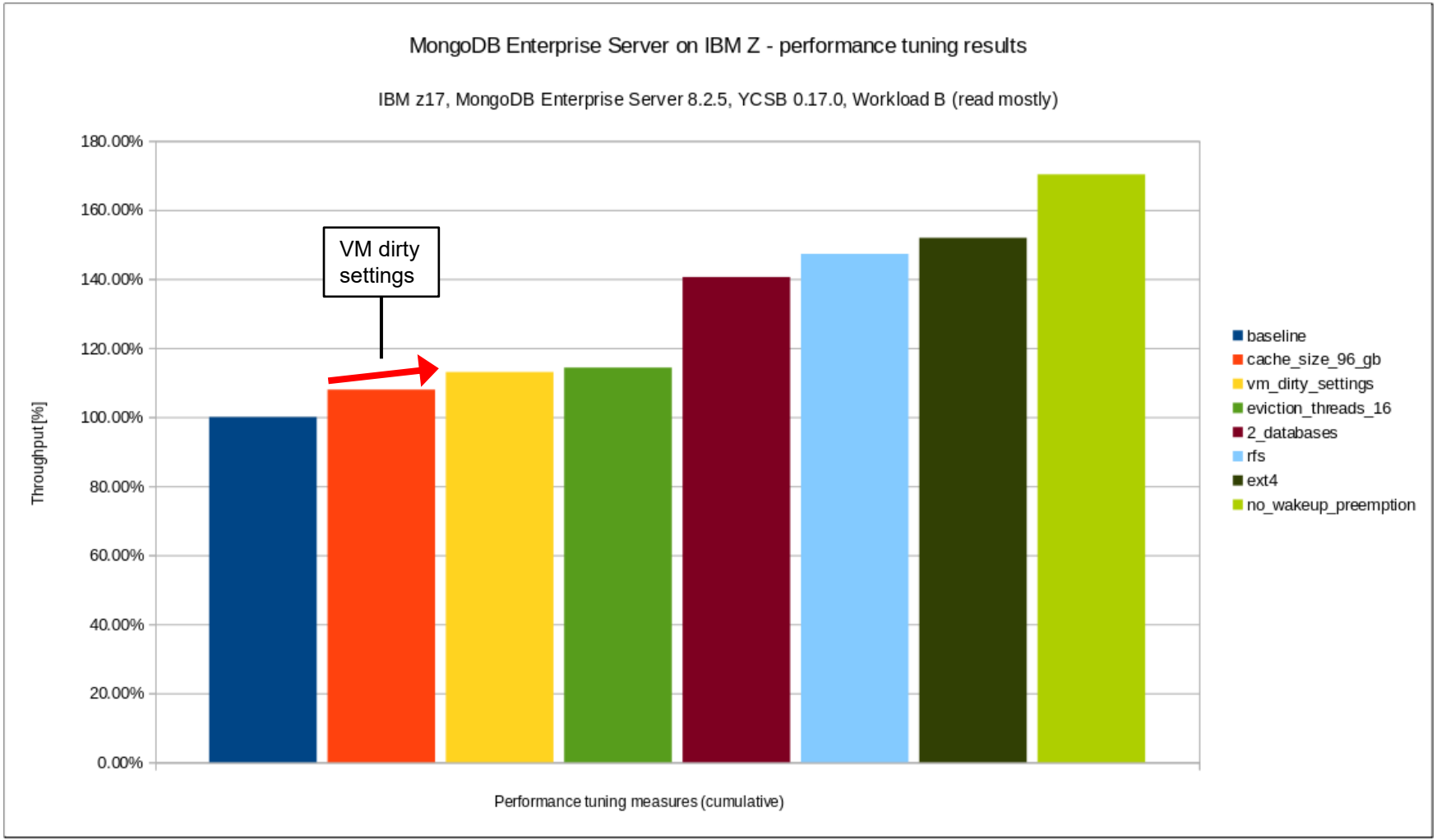
- Recommendation #2: configure the writeback of ***dirty memory*** (i.e., dirty pages) in Linux
- This task consists of changing the values of 2 related settings:
  - `vm.dirty_background_bytes` (counterpart setting in %: `vm.dirty_background_ratio`)
  - `vm.dirty_bytes` (counterpart setting in %: `vm.dirty_ratio`)
- Background – see Linux kernel documentation:
  - The `dirty_background_bytes` setting *"contains the amount of dirty memory at which the **background kernel flusher threads** will start writeback"* (quote from the above link)
  - The `dirty_bytes` setting, on the other hand, *"contains the amount of dirty memory at which a **process generating disk writes will itself** start writeback"* (quote)

Recommendation #2, *continued*

- The ***default*** values for those 2 settings are (a) configured in ***percent*** and (b) result in ***pretty large values*** for Linux images with large amounts of memory
  - For example, `vm.dirty_background_ratio` resulted in 12.8 GiB in my test environment (default value of this setting is 10%, and 10% of 128 GiB is 12.8 GiB)
  - For both settings, ***only 1 variant*** can be active at any given point in time: either the value in percent, or the absolute value in bytes
- Recommendation:
  - `vm.dirty_background_bytes=67108864` (64 MiB)
  - `vm.dirty_bytes=536870912` (512 MiB)

Recommendation #2, *continued*

- This will force the writeback of dirty memory ***much earlier*** compared to the default settings, which in turn results in a ***much more balanced*** disk I/O behavior and ***much smoother*** end user response times
  - This recommendation avoids ***burst waves*** in disk I/O
  - Therefore, the mentioned settings are more about ***response time*** than throughput
  - In numbers for the MongoDB workload: while ***throughput*** increases by ca. 5% when applying this setting, the ***95th percentile*** for the ***update latency*** decreases by ca. 22%
- In order to ***persist*** those values across reboots, add them to the `/etc/sysctl.conf` configuration file and either (a) reboot your Linux image or (b) run the `sysctl -p` command


MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
VM dirty settings
Throughput [%]
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

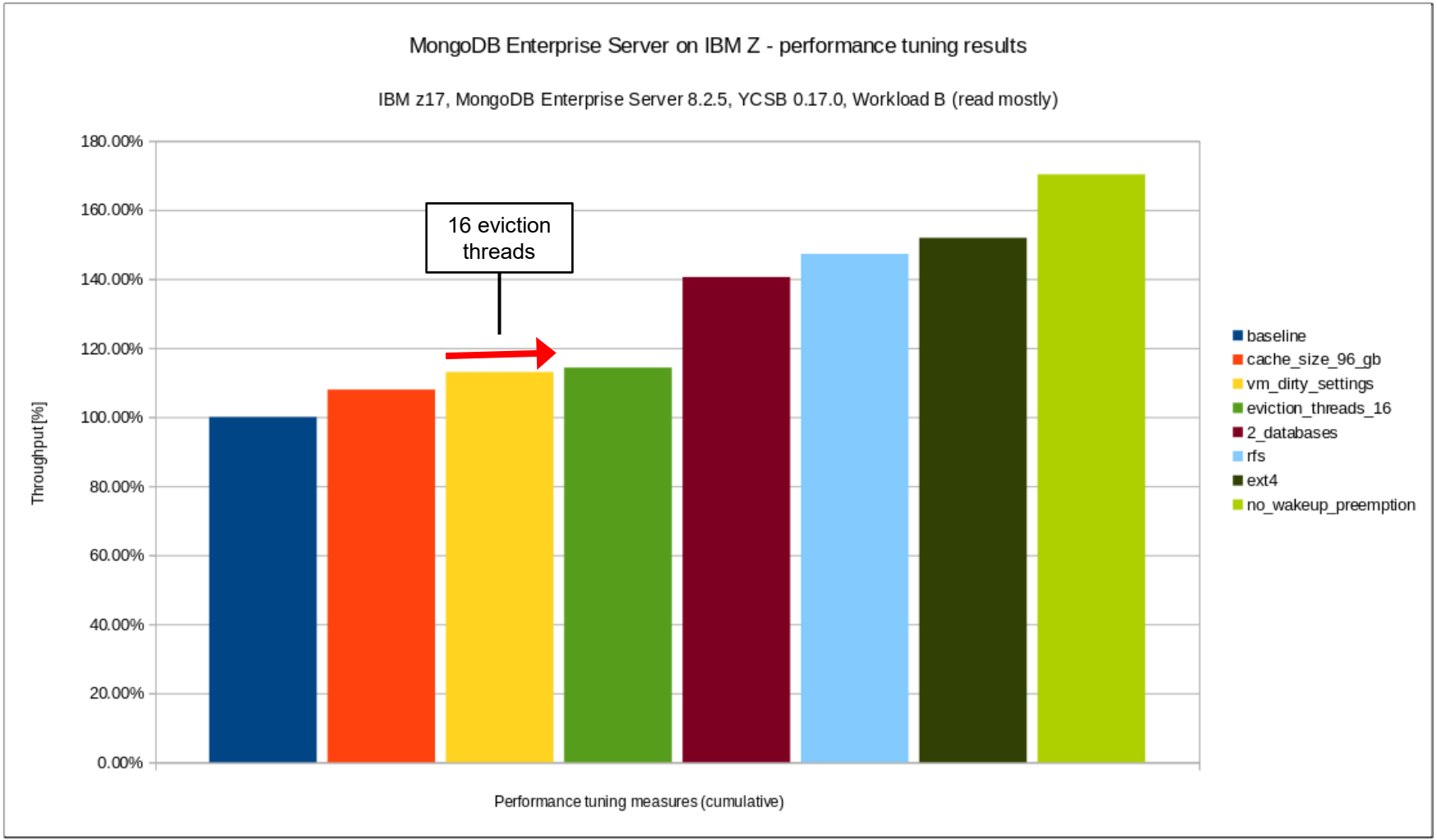
Recommendation #3

- Recommendation #3: increase the number of WiredTiger ***eviction threads***
- In WiredTiger, ***eviction*** is the process of moving not currently accessed pages out of the cache (to disk) and user-requested pages back into the cache (from disk)
  - Quote from the documentation: *"approximating a least-recently-used algorithm"*
- The ***default*** number of eviction threads for WiredTiger in MongoDB is `4`
  - According to a number of blog articles, this is ***too small*** for cache-intensive workloads
  - If the eviction worker threads cannot keep up with the eviction load, then ***application threads*** are used to assist with eviction – this can lead to ***spikes*** in application response times
- Although the impact on throughput is ***small*** in my test environment, I would still recommend this setting

Recommendation #3, *continued*

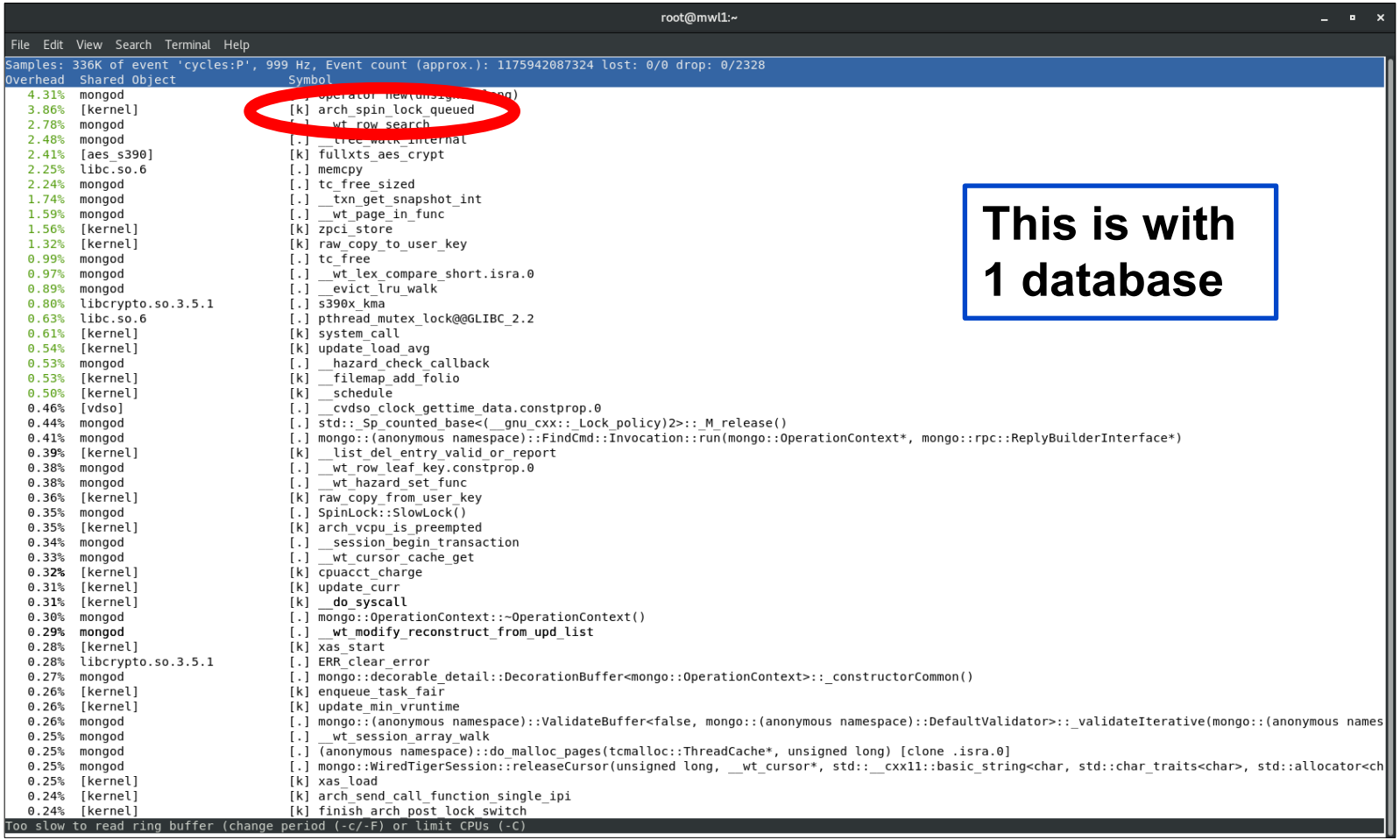
- In order to configure the number of WiredTiger eviction threads, add the following line ***in blue*** to your `mongod.conf` configuration file:

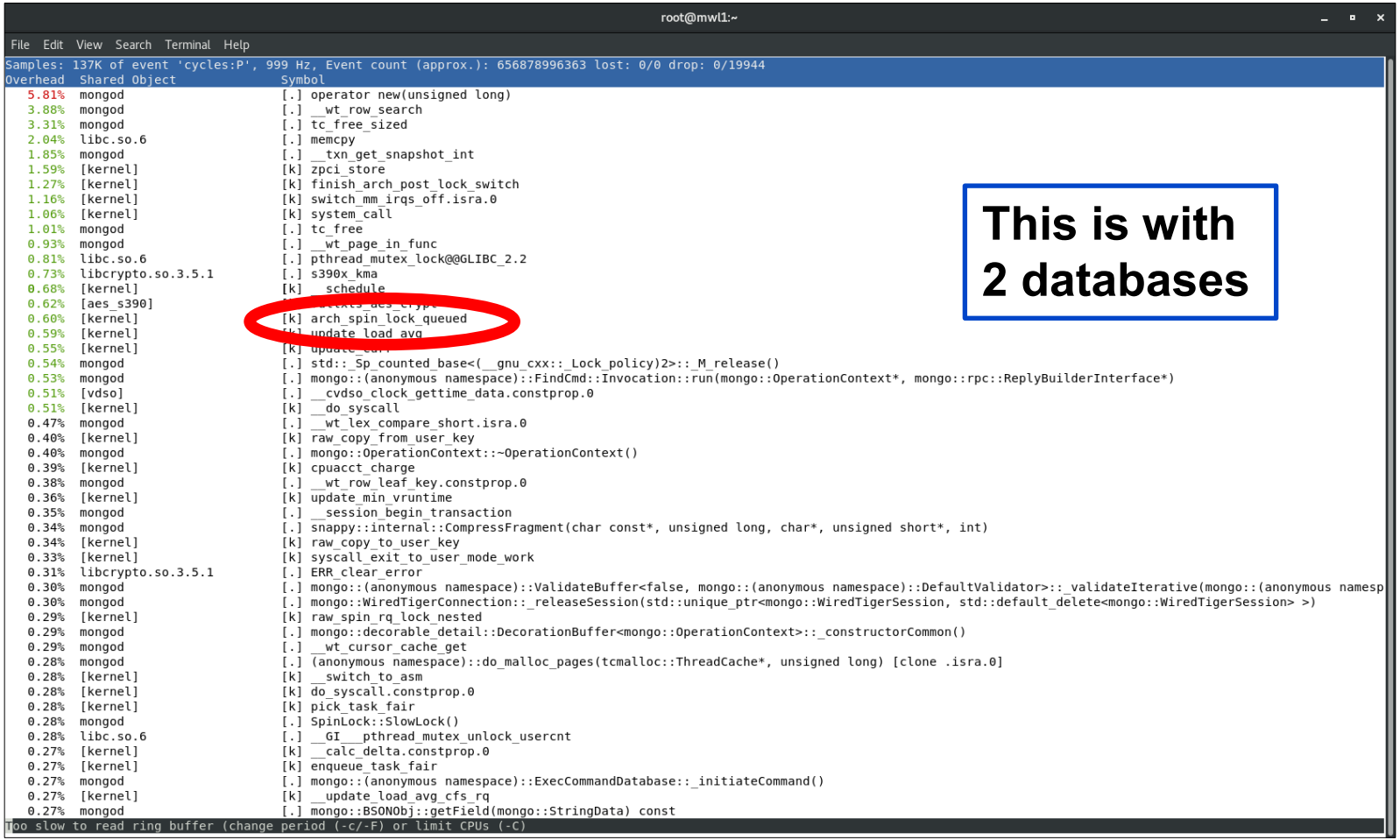
```
...
storage:
   dbPath: /data
   directoryPerDB: true
   engine: "wiredTiger"
   wiredTiger:
      ...
      engineConfig:
         cacheSizeGB: 96
         configString: "eviction=(threads_min=16,threads_max=16)"
...
```

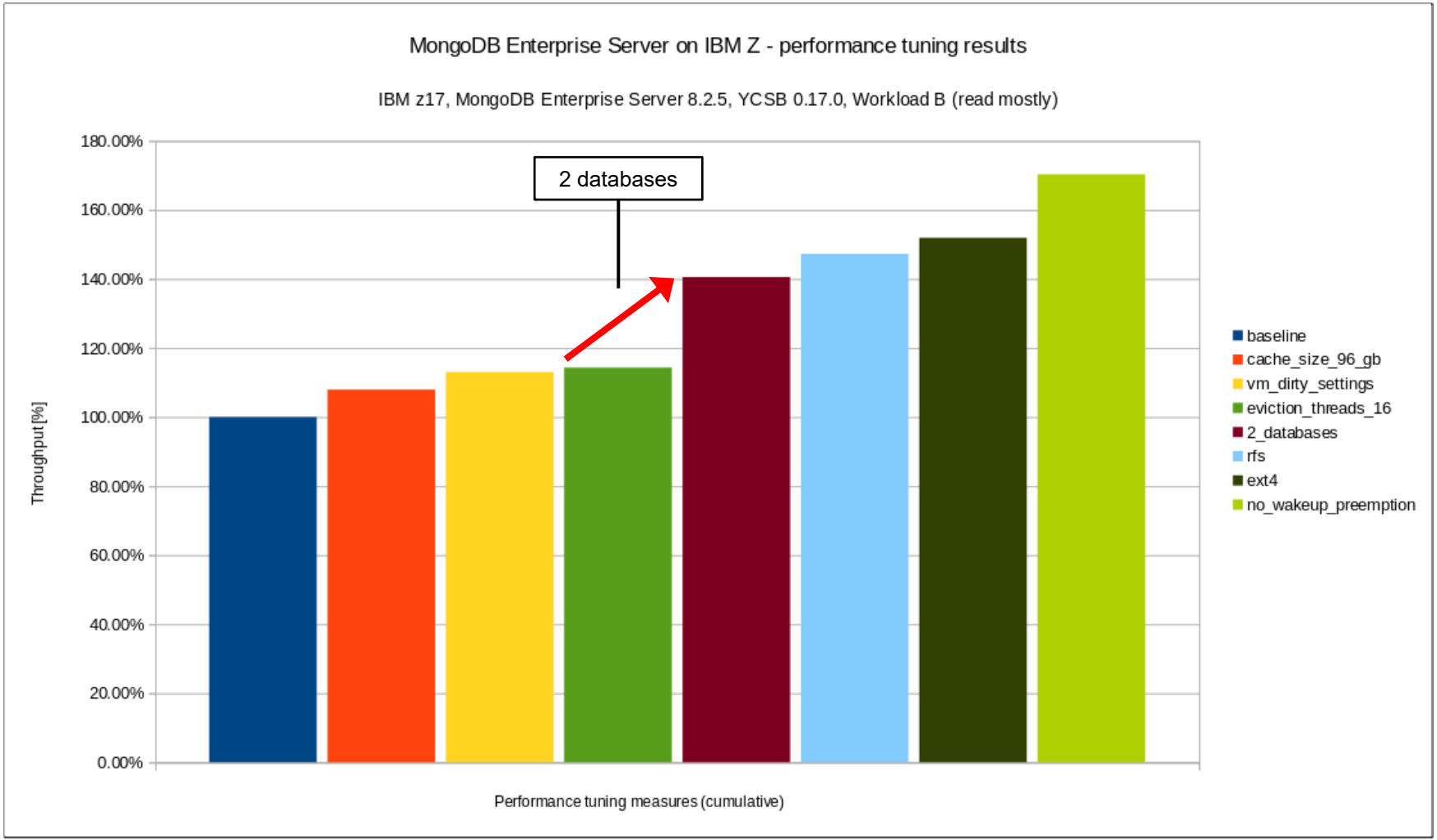

MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Throughput [%]
16 eviction threads
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption
Performance tuning measures (cumulative)

Recommendation #4

- Recommendation #4: if possible, ***split*** your MongoDB data across ***more than 1*** database
- When analyzing the MongoDB workload at runtime, I noticed that there is ***heavy locking activity*** at times when a ***checkpoint*** occurs
  - See the Linux `perf` screenshot on the next page
- In MongoDB, a checkpoint is a ***snapshot-in-time*** that ensures all ***dirty data pages*** (modified data in memory) are flushed (i.e., written out) to the database data files on disk, ensuring that the data is durable and consistent up to that point
- There is a possibility that the same improvement can be achieved with more than 1 ***collection*** in MongoDB, but using multiple databases was easier to implement with YCSB

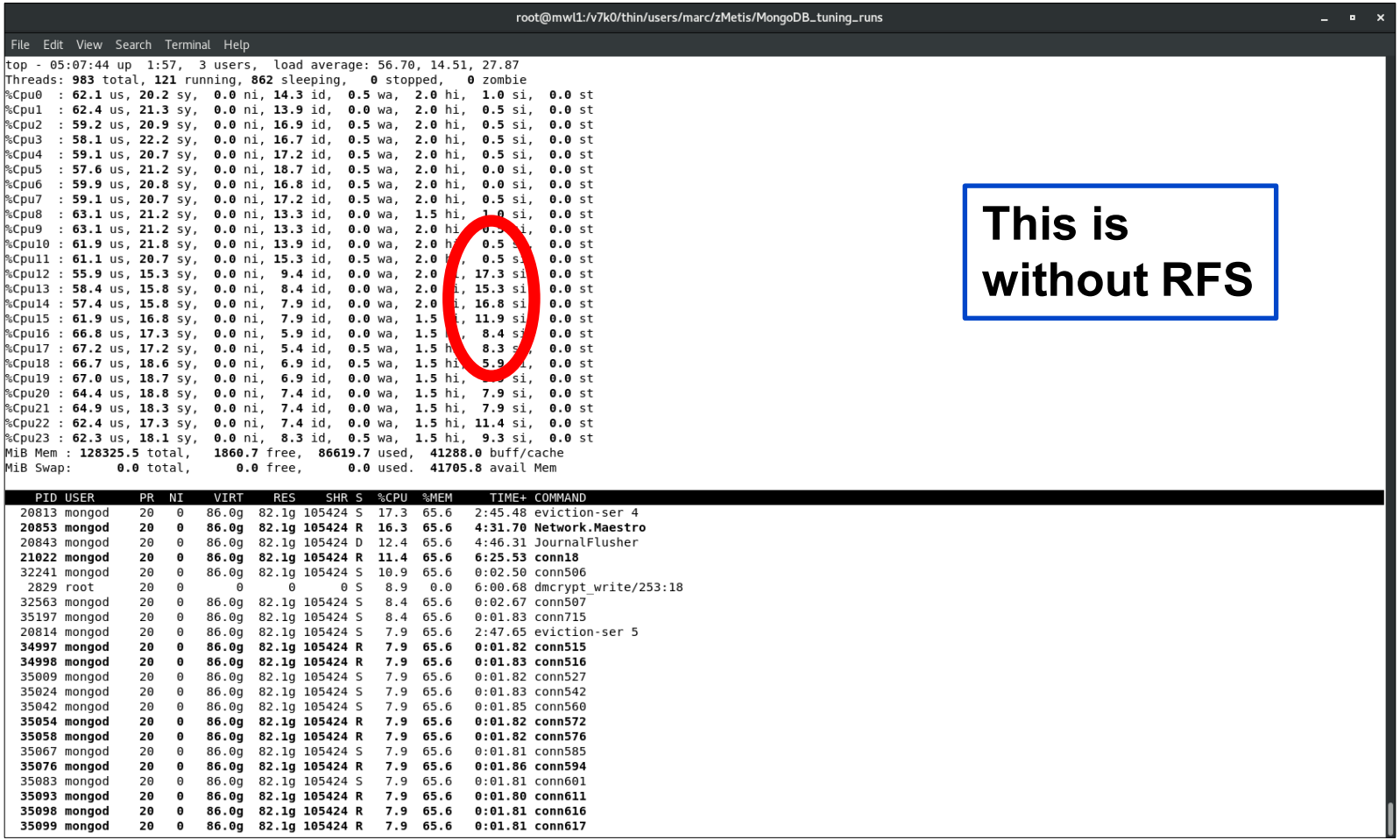

root@mwl1:~
File Edit View Search Terminal Help
Samples: 336K of event 'cycles:P', 999 Hz, Event count (approx.): 1175942087324 lost: 0/0 drop: 0/2328
Overhead Shared Object Symbol
4.31% mongod [.] operator new(unsig...ong)
3.86% [kernel] [k] arch_spin_lock_queued
2.78% mongod [.] __wt_row_search
2.48% mongod [.] __cfree_walk_internal
2.41% [aes_s390] [k] fullxts_aes_crypt
2.25% libc.so.6 [.] memcpy
2.24% mongod [.] tc_free_sized
1.74% mongod [.] __txn_get_snapshot_int
1.59% mongod [.] __wt_page_in_func
1.56% [kernel] [k] zpci_store
1.32% [kernel] [k] raw_copy_to_user_key
0.99% mongod [.] tc_free
0.97% mongod [.] __wt_lex_compare_short.isra.0
0.89% mongod [.] __evict_lru_walk
0.80% libcrypto.so.3.5.1 [.] s390x_kma
0.63% libc.so.6 [.] pthread_mutex_lock@@GLIBC_2.2
0.61% [kernel] [k] system_call
0.54% [kernel] [k] update_load_avg
0.53% mongod [.] __hazard_check_callback
0.53% [kernel] [k] __filemap_add_folio
0.50% [kernel] [k] __schedule
0.46% [vdso] [.] __cvdso_clock_gettime_data.constprop.0
0.44% mongod [.] std::_Sp_counted_base<(__gnu_cxx::_Lock_policy)2>::_M_release()
0.41% mongod [.] mongo::(anonymous namespace)::FindCmd::Invocation::run(mongo::OperationContext*, mongo::rpc::ReplyBuilderInterface*)
0.39% [kernel] [k] __list_del_entry_valid_or_report
0.38% mongod [.] __wt_row_leaf_key.constprop.0
0.38% mongod [.] __wt_hazard_set_func
0.36% [kernel] [k] raw_copy_from_user_key
0.35% mongod [.] SpinLock::SlowLock()
0.35% [kernel] [k] arch_vcpu_is_preempted
0.34% mongod [.] __session_begin_transaction
0.33% mongod [.] __wt_cursor_cache_get
0.32% [kernel] [k] cpuacct_charge
0.31% [kernel] [k] update_curr
0.31% [kernel] [k] __do_syscall
0.30% mongod [.] mongo::OperationContext::~OperationContext()
0.29% mongod [.] __wt_modify_reconstruct_from_upd_list
0.28% [kernel] [k] xas_start
0.28% libcrypto.so.3.5.1 [.] ERR_clear_error
0.27% mongod [.] mongo::decorable_detail::DecorationBuffer<mongo::OperationContext>::_constructorCommon()
0.26% [kernel] [k] enqueue_task_fair
0.26% [kernel] [k] update_min_vruntime
0.26% mongod [.] mongo::(anonymous namespace)::ValidateBuffer<false, mongo::(anonymous namespace)::DefaultValidator>::_validateIterative(mongo::(anonymous names
0.25% mongod [.] __wt_session_array_walk
0.25% mongod [.] (anonymous namespace)::do_malloc_pages(tcmalloc::ThreadCache*, unsigned long) [clone .isra.0]
0.25% mongod [.] mongo::WiredTigerSession::releaseCursor(unsigned long, __wt_cursor*, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<ch
0.25% [kernel] [k] xas_load
0.24% [kernel] [k] arch_send_call_function_single_ipi
0.24% [kernel] [k] finish_arch_post_lock_switch
Too slow to read ring buffer (change period (-c/-F) or limit CPUs (-C)
This is with
1 database

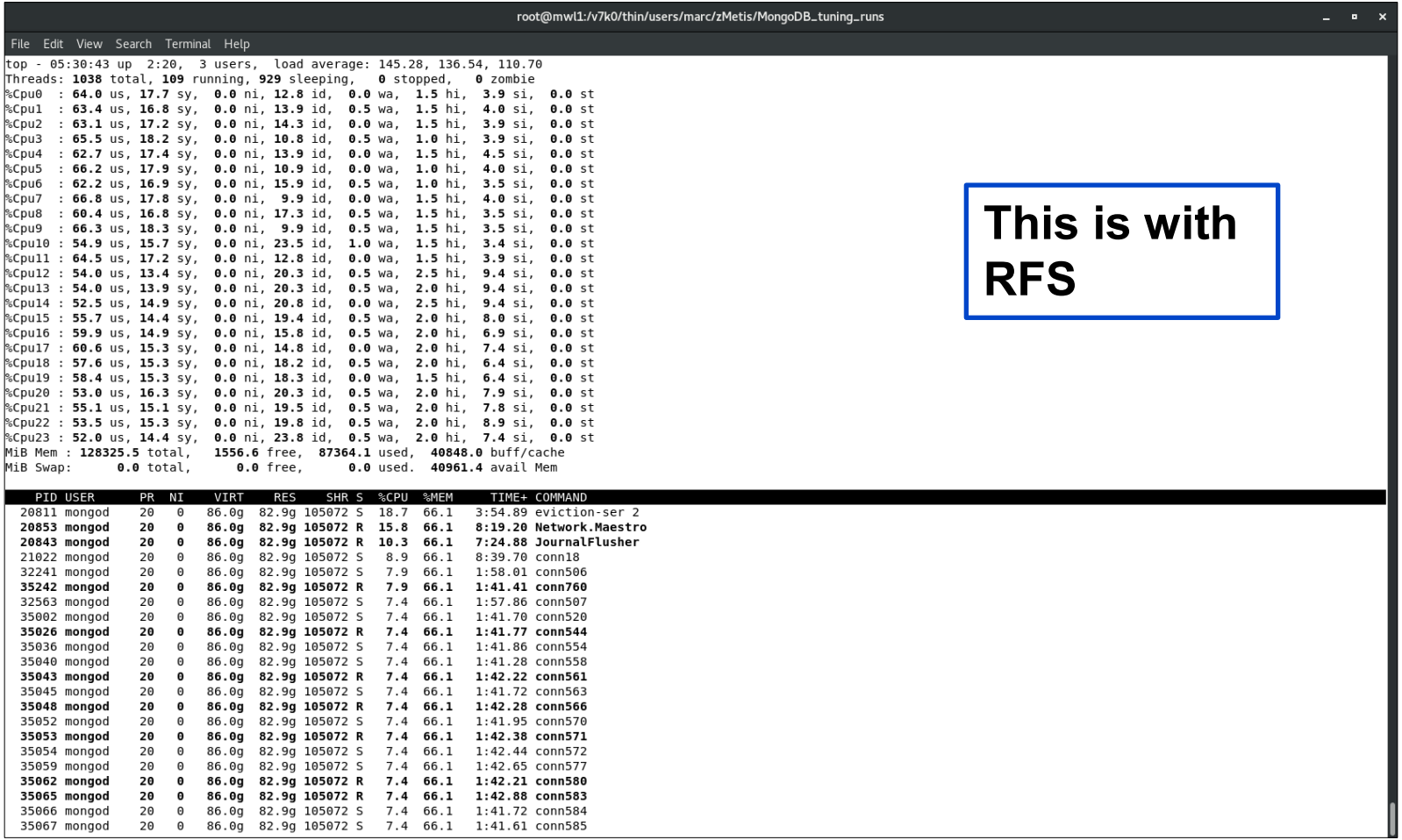

root@mwl1:~
File Edit View Search Terminal Help
Samples: 137K of event 'cycles:P', 999 Hz, Event count (approx.): 656878996363 lost: 0/0 drop: 0/19944
Overhead Shared Object Symbol
5.81% mongod [.] operator new(unsigned long)
3.88% mongod [.] __wt_row_search
3.31% mongod [.] tc_free_sized
2.04% libc.so.6 [.] memcpy
1.85% mongod [.] __txn_get_snapshot_int
1.59% [kernel] [k] zpci_store
1.27% [kernel] [k] finish_arch_post_lock_switch
1.16% [kernel] [k] switch_mm_irqs_off.isra.0
1.06% [kernel] [k] system_call
1.01% mongod [.] tc_free
0.93% mongod [.] __wt_page_in_func
0.81% libc.so.6 [.] pthread_mutex_lock@@GLIBC_2.2
0.73% libcrypto.so.3.5.1 [.] s390x_kma
0.68% [kernel] [k] __schedule
0.62% [aes_s390] [k] [illegible]
0.60% [kernel] [k] arch_spin_lock_queued
0.59% [kernel] [k] update_load_avg
0.55% [kernel] [k] update_curr
0.54% mongod [.] std::_Sp_counted_base<(__gnu_cxx::_Lock_policy)2>::_M_release()
0.53% mongod [.] mongo::(anonymous namespace)::FindCmd::Invocation::run(mongo::OperationContext*, mongo::rpc::ReplyBuilderInterface*)
0.51% [vdso] [.] __cvdso_clock_gettime_data.constprop.0
0.51% [kernel] [k] __do_syscall
0.47% mongod [.] __wt_lex_compare_short.isra.0
0.40% [kernel] [k] raw_copy_from_user_key
0.40% mongod [.] mongo::OperationContext::~OperationContext()
0.39% [kernel] [k] cpuacct_charge
0.38% mongod [.] __wt_row_leaf_key.constprop.0
0.36% [kernel] [k] update_min_vruntime
0.35% mongod [.] __session_begin_transaction
0.34% mongod [.] snappy::internal::CompressFragment(char const*, unsigned long, char*, unsigned short*, int)
0.34% [kernel] [k] raw_copy_to_user_key
0.33% [kernel] [k] syscall_exit_to_user_mode_work
0.31% libcrypto.so.3.5.1 [.] ERR_clear_error
0.30% mongod [.] mongo::(anonymous namespace)::ValidateBuffer<false, mongo::(anonymous namespace)::DefaultValidator>::_validateIterative(mongo::(anonymous namesp
0.30% mongod [.] mongo::WiredTigerConnection::_releaseSession(std::unique_ptr<mongo::WiredTigerSession, std::default_delete<mongo::WiredTigerSession> >)
0.29% [kernel] [k] raw_spin_rq_lock_nested
0.29% mongod [.] mongo::decorable_detail::DecorationBuffer<mongo::OperationContext>::_constructorCommon()
0.29% mongod [.] __wt_cursor_cache_get
0.28% mongod [.] (anonymous namespace)::do_malloc_pages(tcmalloc::ThreadCache*, unsigned long) [clone .isra.0]
0.28% [kernel] [k] __switch_to_asm
0.28% [kernel] [k] do_syscall.constprop.0
0.28% [kernel] [k] pick_task_fair
0.28% mongod [.] SpinLock::SlowLock()
0.28% libc.so.6 [.] __GI___pthread_mutex_unlock_usercnt
0.27% [kernel] [k] __calc_delta.constprop.0
0.27% [kernel] [k] enqueue_task_fair
0.27% mongod [.] mongo::(anonymous namespace)::ExecCommandDatabase::_initiateCommand()
0.27% [kernel] [k] __update_load_avg_cfs_rq
0.27% mongod [.] mongo::BSONObj::getField(mongo::StringData) const
Too slow to read ring buffer (change period (-c/-F) or limit CPUs (-C)
This is with
2 databases


MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
2 databases
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Throughput [%]
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

Recommendation #5

- Recommendation #5: enable ***Receive Flow Steering*** **(RFS)**
- When analyzing the runtime behavior of MongoDB, I noticed that ***software interrupts*** (`%si` in `top`) were balanced very ***unevenly*** across CPUs
  - See the Linux `top` screenshot on the next page
- In order to figure out which device causes software interrupts, you can check `/proc/interrupts`
- To improve this situation, you can either use *Receive Packet Steering* (RPS) or RFS
  - RFS turned out to work much better for this workload


root@mwl1:/v7k0/thin/users/marc/zMetis/MongoDB_tuning_runs
File Edit View Search Terminal Help
top - 05:07:44 up 1:57, 3 users, load average: 56.70, 14.51, 27.87
Threads: 983 total, 121 running, 862 sleeping, 0 stopped, 0 zombie
%Cpu0 : 62.1 us, 20.2 sy, 0.0 ni, 14.3 id, 0.5 wa, 2.0 hi, 1.0 si, 0.0 st
%Cpu1 : 62.4 us, 21.3 sy, 0.0 ni, 13.9 id, 0.0 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu2 : 59.2 us, 20.9 sy, 0.0 ni, 16.9 id, 0.5 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu3 : 58.1 us, 22.2 sy, 0.0 ni, 16.7 id, 0.5 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu4 : 59.1 us, 20.7 sy, 0.0 ni, 17.2 id, 0.5 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu5 : 57.6 us, 21.2 sy, 0.0 ni, 18.7 id, 0.5 wa, 2.0 hi, 0.0 si, 0.0 st
%Cpu6 : 59.9 us, 20.8 sy, 0.0 ni, 16.8 id, 0.5 wa, 2.0 hi, 0.0 si, 0.0 st
%Cpu7 : 59.1 us, 20.7 sy, 0.0 ni, 17.2 id, 0.5 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu8 : 63.1 us, 21.2 sy, 0.0 ni, 13.3 id, 0.0 wa, 1.5 hi, 1.0 si, 0.0 st
%Cpu9 : 63.1 us, 21.2 sy, 0.0 ni, 13.3 id, 0.0 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu10 : 61.9 us, 21.8 sy, 0.0 ni, 13.9 id, 0.0 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu11 : 61.1 us, 20.7 sy, 0.0 ni, 15.3 id, 0.5 wa, 2.0 hi, 0.5 si, 0.0 st
%Cpu12 : 55.9 us, 15.3 sy, 0.0 ni, 9.4 id, 0.0 wa, 2.0 hi, 17.3 si, 0.0 st
%Cpu13 : 58.4 us, 15.8 sy, 0.0 ni, 8.4 id, 0.0 wa, 2.0 hi, 15.3 si, 0.0 st
%Cpu14 : 57.4 us, 15.8 sy, 0.0 ni, 7.9 id, 0.0 wa, 2.0 hi, 16.8 si, 0.0 st
%Cpu15 : 61.9 us, 16.8 sy, 0.0 ni, 7.9 id, 0.0 wa, 1.5 hi, 11.9 si, 0.0 st
%Cpu16 : 66.8 us, 17.3 sy, 0.0 ni, 5.9 id, 0.0 wa, 1.5 hi, 8.4 si, 0.0 st
%Cpu17 : 67.2 us, 17.2 sy, 0.0 ni, 5.4 id, 0.5 wa, 1.5 hi, 8.3 si, 0.0 st
%Cpu18 : 66.7 us, 18.6 sy, 0.0 ni, 6.9 id, 0.5 wa, 1.5 hi, 5.9 si, 0.0 st
%Cpu19 : 67.0 us, 18.7 sy, 0.0 ni, 6.9 id, 0.0 wa, 1.5 hi, 5.9 si, 0.0 st
%Cpu20 : 64.4 us, 18.8 sy, 0.0 ni, 7.4 id, 0.0 wa, 1.5 hi, 7.9 si, 0.0 st
%Cpu21 : 64.9 us, 18.3 sy, 0.0 ni, 7.4 id, 0.0 wa, 1.5 hi, 7.9 si, 0.0 st
%Cpu22 : 62.4 us, 17.3 sy, 0.0 ni, 7.4 id, 0.0 wa, 1.5 hi, 11.4 si, 0.0 st
%Cpu23 : 62.3 us, 18.1 sy, 0.0 ni, 8.3 id, 0.5 wa, 1.5 hi, 9.3 si, 0.0 st
MiB Mem : 128325.5 total, 1860.7 free, 86619.7 used, 41288.0 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 41705.8 avail Mem
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
20813 mongod 20 0 86.0g 82.1g 105424 S 17.3 65.6 2:45.48 eviction-ser 4
20853 mongod 20 0 86.0g 82.1g 105424 R 16.3 65.6 4:31.70 Network.Maestro
20843 mongod 20 0 86.0g 82.1g 105424 D 12.4 65.6 4:46.31 JournalFlusher
21022 mongod 20 0 86.0g 82.1g 105424 R 11.4 65.6 6:25.53 conn18
32241 mongod 20 0 86.0g 82.1g 105424 S 10.9 65.6 0:02.50 conn506
2829 root 20 0 0 0 0 S 8.9 0.0 6:00.68 dmcrypt_write/253:18
32563 mongod 20 0 86.0g 82.1g 105424 S 8.4 65.6 0:02.67 conn507
35197 mongod 20 0 86.0g 82.1g 105424 S 8.4 65.6 0:01.83 conn715
20814 mongod 20 0 86.0g 82.1g 105424 S 7.9 65.6 2:47.65 eviction-ser 5
34997 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.82 conn515
34998 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.83 conn516
35009 mongod 20 0 86.0g 82.1g 105424 S 7.9 65.6 0:01.82 conn527
35024 mongod 20 0 86.0g 82.1g 105424 S 7.9 65.6 0:01.83 conn542
35042 mongod 20 0 86.0g 82.1g 105424 S 7.9 65.6 0:01.85 conn560
35054 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.82 conn572
35058 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.82 conn576
35067 mongod 20 0 86.0g 82.1g 105424 S 7.9 65.6 0:01.81 conn585
35076 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.86 conn594
35083 mongod 20 0 86.0g 82.1g 105424 S 7.9 65.6 0:01.81 conn601
35093 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.80 conn611
35098 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.81 conn616
35099 mongod 20 0 86.0g 82.1g 105424 R 7.9 65.6 0:01.81 conn617
This is without RFS


root@mwl1:/v7k0/thin/users/marc/zMetis/MongoDB_tuning_runs
File Edit View Search Terminal Help
top - 05:30:43 up 2:20, 3 users, load average: 145.28, 136.54, 110.70
Threads: 1038 total, 109 running, 929 sleeping, 0 stopped, 0 zombie
%Cpu0 : 64.0 us, 17.7 sy, 0.0 ni, 12.8 id, 0.0 wa, 1.5 hi, 3.9 si, 0.0 st
%Cpu1 : 63.4 us, 16.8 sy, 0.0 ni, 13.9 id, 0.5 wa, 1.5 hi, 4.0 si, 0.0 st
%Cpu2 : 63.1 us, 17.2 sy, 0.0 ni, 14.3 id, 0.0 wa, 1.5 hi, 3.9 si, 0.0 st
%Cpu3 : 65.5 us, 18.2 sy, 0.0 ni, 10.8 id, 0.5 wa, 1.0 hi, 3.9 si, 0.0 st
%Cpu4 : 62.7 us, 17.4 sy, 0.0 ni, 13.9 id, 0.0 wa, 1.5 hi, 4.5 si, 0.0 st
%Cpu5 : 66.2 us, 17.9 sy, 0.0 ni, 10.9 id, 0.0 wa, 1.0 hi, 4.0 si, 0.0 st
%Cpu6 : 62.2 us, 16.9 sy, 0.0 ni, 15.9 id, 0.5 wa, 1.0 hi, 3.5 si, 0.0 st
%Cpu7 : 66.8 us, 17.8 sy, 0.0 ni, 9.9 id, 0.0 wa, 1.5 hi, 4.0 si, 0.0 st
%Cpu8 : 60.4 us, 16.8 sy, 0.0 ni, 17.3 id, 0.5 wa, 1.5 hi, 3.5 si, 0.0 st
%Cpu9 : 66.3 us, 18.3 sy, 0.0 ni, 9.9 id, 0.5 wa, 1.5 hi, 3.5 si, 0.0 st
%Cpu10 : 54.9 us, 15.7 sy, 0.0 ni, 23.5 id, 1.0 wa, 1.5 hi, 3.4 si, 0.0 st
%Cpu11 : 64.5 us, 17.2 sy, 0.0 ni, 12.8 id, 0.0 wa, 1.5 hi, 3.9 si, 0.0 st
%Cpu12 : 54.0 us, 13.4 sy, 0.0 ni, 20.3 id, 0.5 wa, 2.5 hi, 9.4 si, 0.0 st
%Cpu13 : 54.0 us, 13.9 sy, 0.0 ni, 20.3 id, 0.5 wa, 2.0 hi, 9.4 si, 0.0 st
%Cpu14 : 52.5 us, 14.9 sy, 0.0 ni, 20.8 id, 0.0 wa, 2.5 hi, 9.4 si, 0.0 st
%Cpu15 : 55.7 us, 14.4 sy, 0.0 ni, 19.4 id, 0.5 wa, 2.0 hi, 8.0 si, 0.0 st
%Cpu16 : 59.9 us, 14.9 sy, 0.0 ni, 15.8 id, 0.5 wa, 2.0 hi, 6.9 si, 0.0 st
%Cpu17 : 60.6 us, 15.3 sy, 0.0 ni, 14.8 id, 0.0 wa, 2.0 hi, 7.4 si, 0.0 st
%Cpu18 : 57.6 us, 15.3 sy, 0.0 ni, 18.2 id, 0.5 wa, 2.0 hi, 6.4 si, 0.0 st
%Cpu19 : 58.4 us, 15.3 sy, 0.0 ni, 18.3 id, 0.0 wa, 1.5 hi, 6.4 si, 0.0 st
%Cpu20 : 53.0 us, 16.3 sy, 0.0 ni, 20.3 id, 0.5 wa, 2.0 hi, 7.9 si, 0.0 st
%Cpu21 : 55.1 us, 15.1 sy, 0.0 ni, 19.5 id, 0.5 wa, 2.0 hi, 7.8 si, 0.0 st
%Cpu22 : 53.5 us, 15.3 sy, 0.0 ni, 19.8 id, 0.5 wa, 2.0 hi, 8.9 si, 0.0 st
%Cpu23 : 52.0 us, 14.4 sy, 0.0 ni, 23.8 id, 0.5 wa, 2.0 hi, 7.4 si, 0.0 st
MiB Mem : 128325.5 total, 1556.6 free, 87364.1 used, 40848.0 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 40961.4 avail Mem
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
20811 mongod 20 0 86.0g 82.9g 105072 S 18.7 66.1 3:54.89 eviction-ser 2
20853 mongod 20 0 86.0g 82.9g 105072 R 15.8 66.1 8:19.20 Network.Maestro
20843 mongod 20 0 86.0g 82.9g 105072 R 10.3 66.1 7:24.88 JournalFlusher
21022 mongod 20 0 86.0g 82.9g 105072 S 8.9 66.1 8:39.70 conn18
32241 mongod 20 0 86.0g 82.9g 105072 S 7.9 66.1 1:58.01 conn506
35242 mongod 20 0 86.0g 82.9g 105072 R 7.9 66.1 1:41.41 conn760
32563 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:57.86 conn507
35002 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.70 conn520
35026 mongod 20 0 86.0g 82.9g 105072 R 7.4 66.1 1:41.77 conn544
35036 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.86 conn554
35040 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.28 conn558
35043 mongod 20 0 86.0g 82.9g 105072 R 7.4 66.1 1:42.22 conn561
35045 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.72 conn563
35048 mongod 20 0 86.0g 82.9g 105072 R 7.4 66.1 1:42.28 conn566
35052 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.95 conn570
35053 mongod 20 0 86.0g 82.9g 105072 R 7.4 66.1 1:42.38 conn571
35054 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:42.44 conn572
35059 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:42.65 conn577
35062 mongod 20 0 86.0g 82.9g 105072 R 7.4 66.1 1:42.21 conn580
35065 mongod 20 0 86.0g 82.9g 105072 R 7.4 66.1 1:42.88 conn583
35066 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.72 conn584
35067 mongod 20 0 86.0g 82.9g 105072 S 7.4 66.1 1:41.61 conn585
This is with RFS

Recommendation #5, *continued*

- Both RPS and RFS select a CPU to perform protocol processing ***above the interrupt handler***
  - See kernel.org for details
- Quote from the above link: *"While RPS steers packets solely based on hash, … it does not take into account* ***application locality****. This is accomplished by Receive Flow Steering (RFS)."*
  - RFS uses a ***flow lookup table*** for this

Recommendation #5, *continued*

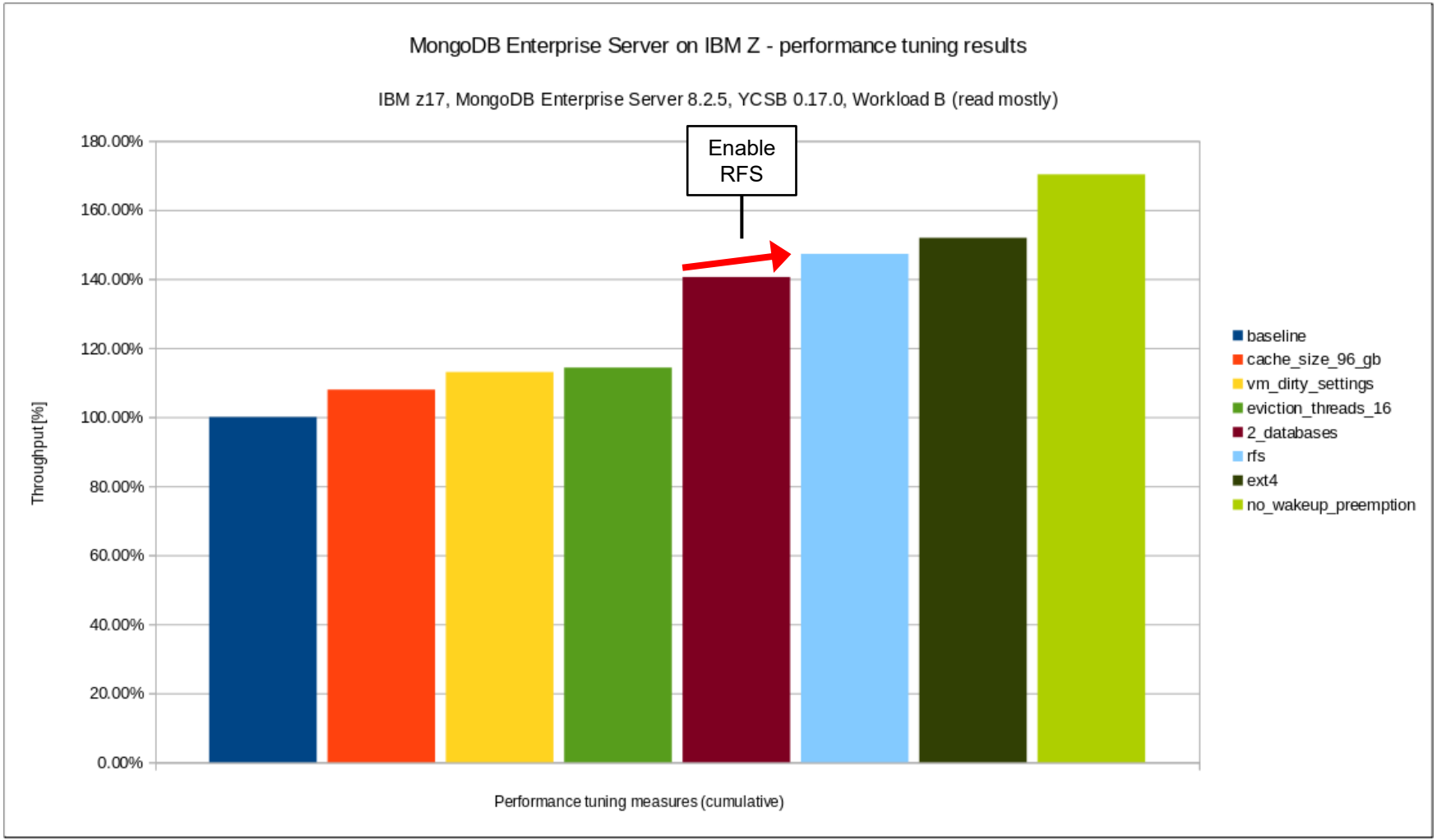
- In order to enable RFS for the leader node's Network Express adapter, I used the following script:

```
#!/bin/sh

# this is the network interface name
INTERFACE_NAME="ens14352"

# enable rfs globally
echo 32768 | sudo tee /proc/sys/net/core/rps_sock_flow_entries

# configure the rx queues of the card
echo 8192 | sudo tee /sys/class/net/${INTERFACE_NAME}/queues/rx-0/rps_flow_cnt
echo 8192 | sudo tee /sys/class/net/${INTERFACE_NAME}/queues/rx-1/rps_flow_cnt
echo 8192 | sudo tee /sys/class/net/${INTERFACE_NAME}/queues/rx-2/rps_flow_cnt
echo 8192 | sudo tee /sys/class/net/${INTERFACE_NAME}/queues/rx-3/rps_flow_cnt
echo 8192 | sudo tee /sys/class/net/${INTERFACE_NAME}/queues/rx-4/rps_flow_cnt
```

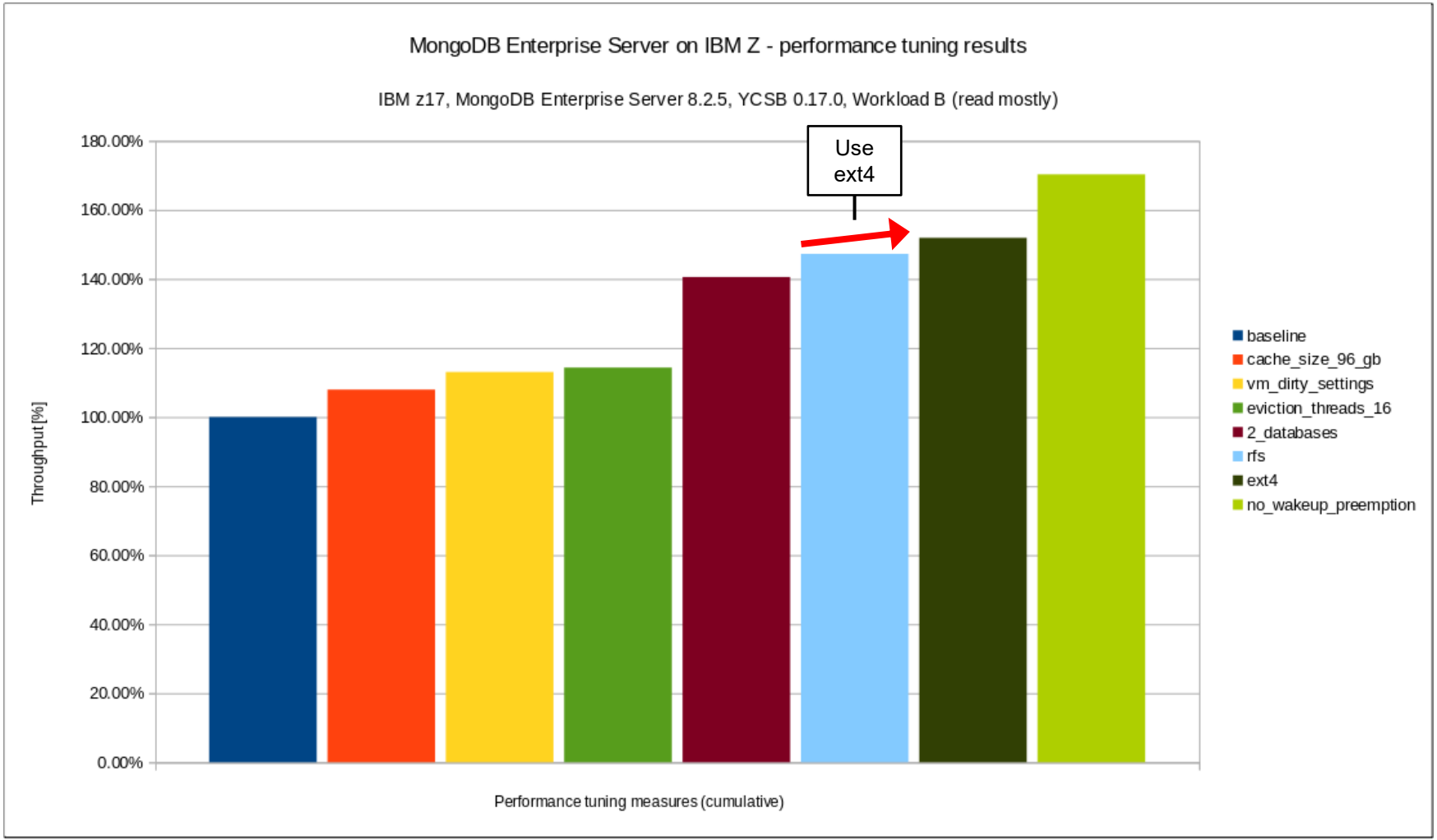

MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
Enable RFS
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Throughput [%]
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

Recommendation #6

- Recommendation #6: use ***ext4*** for storing MongoDB data
- Please keep in mind that this is solely a ***performance recommendation***
  - MongoDB itself recommends ***XFS®*** for storing data
- Observation #1: in my tests with YCSB, the entire collection was stored in a ***single file*** on disk
  - I did ***not*** use ***sharding*** for my benchmarks
- Observation #2: in the version of RHEL® 10 that I used, XFS causes ***locking issues*** when there is too much disk I/O traffic on a single file
  - See the console output on the next slide

Recommendation #6, *continued*

```
[root@mwl1 ~]# ps -eLo pid,tid,state,wchan:30,comm | grep " D "
 598495  598510 D xfs_ilock                     eviction-ser 1
 598495  598511 D xfs_ilock                     eviction-ser 2
 598495  598512 D xfs_ilock                     eviction-ser 3
 598495  598513 D -                             eviction-ser 4
 598495  598514 D xfs_ilock                     eviction-ser 5
 598495  598515 D xfs_ilock                     eviction-ser 6
 598495  598516 D xfs_ilock                     eviction-ser 7
 598495  598518 D xfs_ilock                     eviction-ser 9
 598495  598519 D xfs_ilock                     eviction-ser 10
 598495  598522 D xfs_ilock                     eviction-ser 13
 598495  598523 D xfs_ilock                     eviction-ser 14
 598495  598525 D -                             eviction-ser 16
 598495  602222 D -                             conn201
 598495  602223 D xfs_ilock                     conn202
 598495  602225 D xfs_ilock                     conn204
 598495  602227 D xfs_ilock                     conn206
 598495  602228 D -                             conn207
 598495  602232 D xfs_ilock                     conn211
 ...
```


MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
Use ext4
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Throughput [%]
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

Recommendation #7

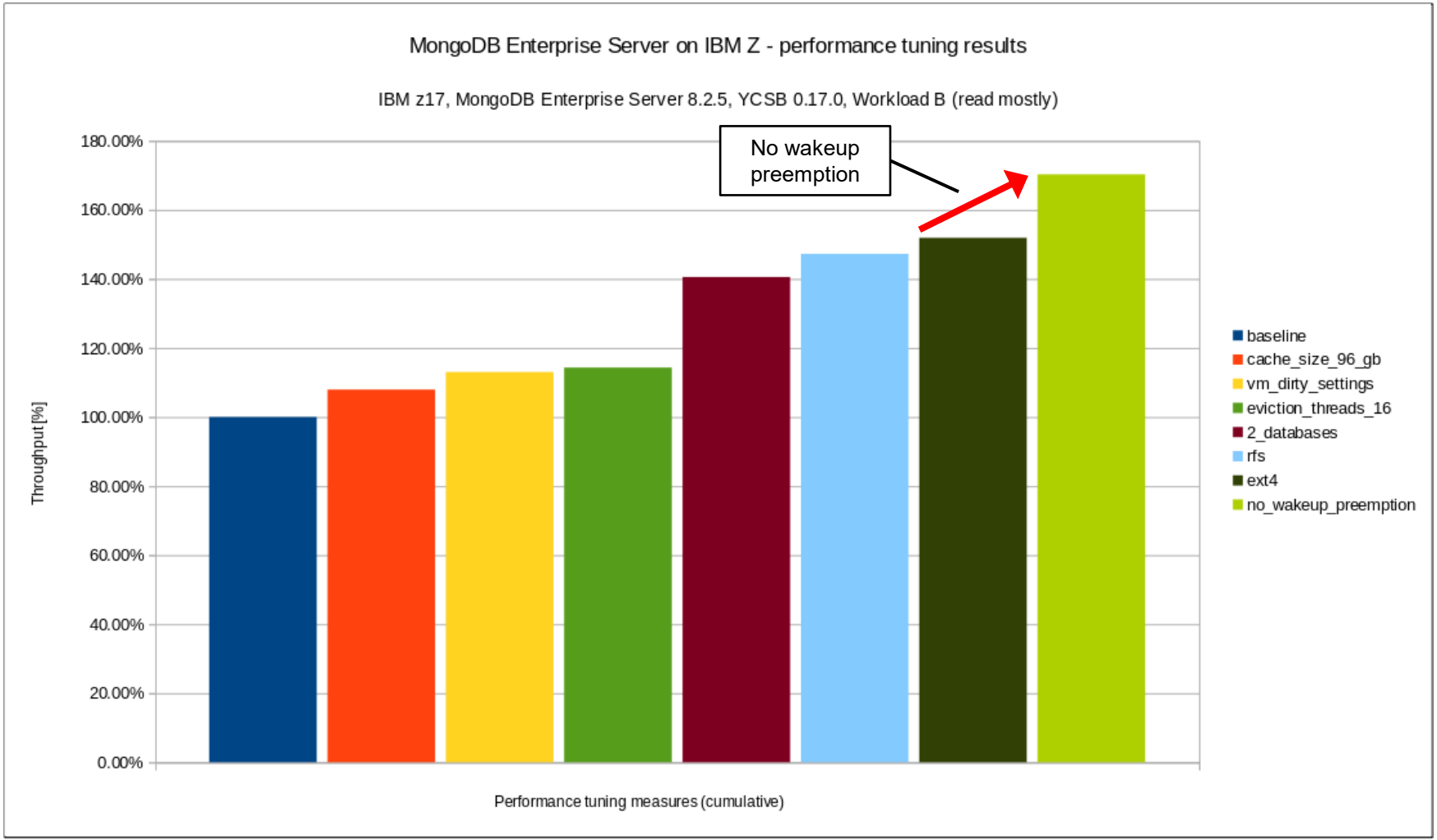
- Recommendation #7: when using EEVDF, disable ***wakeup preemption***
- Starting with version 6.6, the Linux kernel began transitioning to the ***Earliest Eligible Virtual Deadline First (EEVDF)*** scheduler
  - EEVDF replaces the former *Completely Fair Scheduler* (CFS)
  - Unlike CFS, EEVDF uses – amongst other things – the concept of ***virtual deadlines***
- "Wakeup preemption" in this context means that a newly woken-up task can ***immediately interrupt*** ("preempt") a currently running task
  - The goal of this feature is to balance ***latency*** vs. ***throughput***
  - One consequence: increased number of ***context switches***

Recommendation #7, *continued*

- You can disable wakeup preemption with the following ***command***:

```
echo NO_WAKEUP_PREEMPTION > /sys/kernel/debug/sched/features
```

- In order to ***persist*** this setting, you can add a line to `/etc/rc.local` or write a small `systemd` service
  - Nowadays, `systemd` services are the preferred way for applying settings like this


MongoDB Enterprise Server on IBM Z - performance tuning results
IBM z17, MongoDB Enterprise Server 8.2.5, YCSB 0.17.0, Workload B (read mostly)
No wakeup preemption
180.00%
160.00%
140.00%
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
Throughput [%]
Performance tuning measures (cumulative)
baseline
cache_size_96_gb
vm_dirty_settings
eviction_threads_16
2_databases
rfs
ext4
no_wakeup_preemption

Things that turned out ***not*** to improve performance

- The following is an (incomplete) list of things that I tried on top of the recommendations in this guide that turned out ***not to have*** a measurable impact on performance:
  - Commit interval for the MongoDB ***journal*** (the `commitIntervalMs` setting)
  - WiredTiger ***checkpoint interval***
  - Environment settings for the ***TCMalloc*** library
  - ***Read ahead*** value for the LUKS container holding the database data, underlying striped logical volume, and multipath'ed SCSI LUNs
  - ***Jumbo frames*** for the Ethernet devices in all LPARs
  - Linux kernel scheduler ***migration cost***
  - `vm.swappiness` Linux `sysctl` setting
  - *Transparent Huge Pages* (THP)

Agenda

- Introduction to MongoDB
- Performance measurement and tuning approach
- Observations and recommendations
- Summary

Summary

- MongoDB Enterprise Server is a ***very popular*** NoSQL database management system, also on IBM Z
  - Offers a lot of ***enterprise-level*** features
- If you follow the recommendations in this tuning guide, you can ***greatly improve performance*** of your MongoDB installation on Linux on IBM Z
  - Examples: WiredTiger cache size, RFS, no wakeup preemption
  - ***Saturating*** all 24 IFLs of my test environment was only possible after applying all of my tuning recommendations
- ***Remember***: if you're interested in a competitive comparison against Intel x86, please consult the official IBM z16 Proof Points chart deck

Thank you!

Resources

- Linux on IBM Z and IBM LinuxONE
  - Official homepage: https://www.ibm.com/products/z/linux
  - Documentation: https://www.ibm.com/docs/en/linux-on-systems?topic=linux-z-linuxone
  - Tuning hints and tips: https://www.ibm.com/docs/en/linux-on-systems?topic=performance-hot-topics
- MongoDB Enterprise Server: https://www.mongodb.com/try/download/enterprise
- MongoDB documentation: https://www.mongodb.com/docs/
- Webcast *Memory optimization when running MongoDB on Linux on IBM Z*: https://community.ibm.com/community/user/viewdocument/2024-02-07-memory-optimization-whe