Tuning Linux for MongoDB

Tim Vaillancourt
Sr. Technical Operations Architect
About Me

- Joined Percona in January 2016
- Sr Technical Operations Architect for MongoDB
- Previous:
  - EA DICE (MySQL DBA)
  - EA SPORTS (Sys/NoSQL DBA Ops)
  - Amazon/AbeBooks Inc (Sys/MySQL+NoSQL DBA Ops)
- Main techs: MySQL, MongoDB, Cassandra, Solr, Redis, queues, etc
- 10+ years tuning Linux for database workloads *(off and on)*
- Not a kernel-guy, learned from breaking things
Linux

- UNIX-like, mostly POSIX-compliant operating system
- First released on September 17th, 1991 by Linus Torvalds
  - 50Mhz CPUs were considered fast
  - CPUs had 1 core
  - RAM was measured in megabytes
  - Ethernet speed was 1 - 10mbps
- General purpose
  - It will run on a Raspberry Pi -> Mainframes
  - Geared towards many different users and use cases
- Linux 3.2+ is much more efficient
MongoDB

- Document-oriented database first released in 2009
- Thread per connection model
- Non-contiguous memory access pattern
- Storage Engines
  - MMMAPv1
    - Keeps warm data in Linux filesystem cache
    - Highly random I/O pattern
    - Cache uses all the RAM it can get
    - Few background threads
MongoDB

• Storage Engines
  • WiredTiger and RocksDB
    • Built-in Compression
    • Uses combination of in-heap cache and filesystem cache
      • In-heap cache: uncompressed pages
      • Filesystem cache: compressed pages
    • Relatively sequential write patterns, low write overhead
  • Scales with RAM, Disk and CPUs
Ulimit

- Allows per-Linux-user resource constraints
  - Number of User-level Processes
  - Number of Open Files
  - CPU Seconds
  - Scheduling Priority
  - Others...

- MongoDB
  - Should probably have its own VM, container or server
  - Creates a process for each connection
Ulimit

- MongoDB (continued)
  - Creates an open file for each active data file on disk
  - 64,000 open files and 64,000 max processes is a good start

```bash
[ti...ents@entos7 ~]$ cat /etc/security/limits.d/90-mongod.conf
# Mongodb ulimits
# https://www.percona.com/blog/2016/08/12/tuning-linux-for-mongodb/

mongod  soft     nproc  64000
mongod  hard     nproc  64000
mongod  soft     nofile 64000
mongod  hard     nofile 64000
```

- Restart mongod/mongos after the ulimit change to apply changes to ulimit
Virtual Memory: Dirty Ratio

- Dirty Pages
  - Pages stored in-cache, but needs to be written to storage
- VM Dirty Ratio
  - Max percent of total memory that can be dirty
  - VM stalls and flushes when this limit is reached
  - Start with '10', default (30) too high
- VM Dirty Background Ratio
  - Separate threshold for background dirty page flushing
  - Flushes without pauses
  - Start with '3', default (15) too high
Virtual Memory: Swappiness

- A Linux kernel sysctl setting for preferring RAM or disk for swap
  - Linux default: 60
  - To avoid disk-based swap: 1 (not zero!)
  - To allow some disk-based swap: 10
  - ‘0’ can cause unpredicted behaviour

```
[tim@centos7 ~]$ sudo vim /etc/sysctl.conf
[tim@centos7 ~]$ grep swap /etc/sysctl.conf
vm.swappiness = 1
[tim@centos7 ~]$ sudo sysctl -p
vm.dirty_ratio = 15
vm.dirty_background_ratio = 5
vm.swappiness = 1
```
Virtual Memory: Transparent HugePages

- Introduced in RHEL/CentOS 6, Linux 2.6.38+
- Merges memory pages in background *(Khugepaged process)*
- Decreases overall performance when used with MongoDB!
- Disable it
  - Add “transparent_hugepage=never” to kernel command-line (GRUB)
  - Reboot
NUMA (Non-Uniform Memory Access)

- A memory architecture that takes into account the locality of memory, caches and CPUs for lower latency
- MongoDB code base is not NUMA “aware”, causing unbalanced allocations
- Disable NUMA
  - In the server BIOS
  - Using ‘numactl’ in mongod init script BEFORE ‘mongod’ command:

  ```
  numactl --interleave=all /usr/bin/mongod <other flags>
  ```
Block Devices: IO Scheduler

- Algorithm kernel uses to commit reads and writes to disk
- CFQ
  - Linux default
  - Perhaps too clever/inefficient for database workloads
- Deadline
  - Best general default IMHO
  - Predictable I/O request latencies
- Noop
  - Use with virtualisation or (sometimes) with BBU RAID controllers
Block Devices: Block Read-ahead

- Tuning that causes data ahead of a block on disk to be read and then cached
- **Assumption:** there is a sequential read pattern and something will benefit from the extra cached blocks
- **Risk:** too high waste cache space and increases eviction work
- MongoDB tends to have very random disk patterns
- A good start for MongoDB volumes is a ’32’ (16kb) read-ahead
Block Devices: Udev rule

• Add file to ‘/etc/udev/rules.d’

/etc/udev/rules.d/60-mongodb-disk.rules:
    # set deadline scheduler and 32/16kb read-ahead for /dev/sda
    ACTION="add|change", KERNEL="sda", ATTR{queue/scheduler}="deadline", ATTR{bdi/read_ahead_kb}="16"

• Reboot (or use CLI tools to apply)
Filesystems and Options

- Use XFS or EXT4, not EXT3
- Use XFS only on WiredTiger
- Set `noatime` on MongoDB data volumes in `/etc/fstab`:

  ```
  /devmapper/data-mongodb /var/lib/mongo ext4 defaults,noatime 0 0
  ```

- Remount the filesystem after an options change, or reboot
Block Devices: Type and Layout

- **Isolation**
  - Run Mongod dbPaths on separate volume
  - Optionally, run Mongod journal on separate volume

- **RAID Level**
  - RAID 10 == performance/durability sweet spot
  - RAID 0 == fast and dangerous

- **SSDs**
  - Benefit MMAPv1 a lot
  - Benefit WT and RocksDB a bit less
  - Keep about 30% free for internal GC on the SSD

- **EBS**
  - Network-attached can be risky

- **JBOD + Replset as Data Redundancy (use at own risk)**
  - Number of Replset Members
  - Read and Write Concern
  - Proper Geolocation/Node Redundancy
Network Stack

• Defaults are not good for > 100mbps Ethernet
• Suggested starting point (add to ‘/etc/sysctl.conf’):

```
[etc/sysctl.conf]
1 net.core.somaxconn = 4096
2 net.ipv4.tcp_fin_timeout = 30
3 net.ipv4.tcp_keepalive_intvl = 30
4 net.ipv4.tcp_keepalive_time = 120
5 net.ipv4.tcp_max_syn_backlog = 4096
```

• Run “sysctl -p” as root to reload Network Stack settings
NTPd (Network Time Protocol)

- Replication and Clustering needs consistent clocks
- Run NTP daemon on all MongoDB and Monitoring hosts
- Enable on restart
- Use a consistent time source/server
SELinux (Security-Enhanced Linux)

- A kernel-level security access control module
- Modes of SELinux
  - **Enforcing:** Block and log policy violations
  - **Permissive:** Log policy violations only
  - **Disabled:** Completely disabled
- **Recommended:** Enforcing
- Percona Server for MongoDB 3.2+ RPMs install an SELinux policy on RedHat/CentOS!
A “framework” for applying tunings to Linux

- RedHat/CentOS 7
- Debian added it, not sure on official status

https://github.com/Percona-Lab/tuned-percona-mongodb

```bash
[tim@centos]~$ sudo tuned-adm list
Available profiles:
- balanced
- desktop
- latency-performance
- network-latency
- network-throughput
- powersave
- throughput-performance
- virtual-guest
- virtual-host
Current active profile: throughput-performance
```

```bash
[tim@centos]~$ sudo tuned-adm profile network-latency
```
CPUs and Frequency Scaling

- Lots of cores > faster cores
- ‘cpufreq’: a daemon for dynamic scaling of the CPU frequency
- Terrible idea for databases
- Disable or set governor to 100% frequency always, i.e. mode: ‘performance’
- Disable any BIOS-level performance/efficiency tuneable
- ENERGY_PERF_BIAS
  - A CentOS/RedHat tuning for energy vs performance balance
  - RHEL 6 = ‘performance’
  - RHEL 7 = ‘normal’ (!)
  - Advice: use ‘tuned’ to set to ‘performance’
Monitoring: Percona PMM

- Open-source monitoring suite from Percona!
- MongoDB visualisations by cluster, shard, replset, engine, etc
- DB stats groupings with OS metrics
- Simple deployment
Monitoring: Prometheus + Grafana

- PerconaLab GitHub Repositories
  - grafana_mongodb_dashboards
  - prometheus_mongodb_exporter
Links

- https://www.percona.com/blog/2016/08/12/tuning-linux-for-mongodb/
- https://docs.mongodb.com/manual/administration/production-notes/

- https://github.com/Percona-Lab/grafana_mongodb_dashboards
- https://github.com/Percona-Lab/prometheus_mongodb_exporter
Questions?
DATABASE PERFORMANCE MATTERS