Linnéuniversitetet
Kalmar Växjö

Individual Assignment 3

Report and answers of all tasks of theird assignment of Opration Systems 1DV512-HT21

*Author:* Saleh Shalabi

ss225bx@student.lnu.se

*Semester:* HT21

*Course code: 1DV512*

*Date: 16/01/2022*

1. Did some of the earlier file systems supported by Linux before BTRFS focus on support for larger volumes of stored data? If yes, which ones?

Yes, both Ext4 and XFS.

2. Does BTRFS share any ideas and design goals with ZFS? If yes, which ones?

Both ZFS and BTRFS are a COW filesystem. The goal of ZFS is to support an enterprise class Solaris server while BTRFS was designed to work on all that run Linux. That mean that where the ZFS is used the BTRFS can also be used.

Both ZFS and BTRFS calculates checksum. The difference is that ZFS kepps them in the block-pointers while BTRFS keeps them in a separate tree.

Snapshots vs. clones: the build snapshots ZFS uses birth-time while BTRFS uses the reference-counting. The result is however very similar.

RAID: while ZFS uses RAID-Z and support several other levels as single/dual parity, BTRFS uses something closer to standard layout of RAID.

3. What is the "extent" concept used for storage by BTRFS? What are its pros and cons compared to the alternative approaches?

Extents is a contiguous on-disk area which is a page-aligned and its length is of multiple of pages. to simplify it is using series of contiguous area of memory to store information.

Using extents make it faster in reading files especially large ones. Even the risk of fragmentation is reduced.

Using extents eliminates the need for special block sizes but it leaves highly fragmented free space which still make writing to disk inefficient.

4. Which structures does BTRFS use to support larger storage devices and map between physical and logical storage space?

A chunk tree is used to support large volume devices. By splitting the device into chunks where no chunk is larger than 10% of the device size. At the time of writing 1GB chunks are used for data and 256MB chunks are used to metadata. Chunk tree maintains a mapping from logical chunks to physical chunks and device tree maintain the reverse mapping while the rest of the filesystem sees logical chunks and extent references address logical chunks.

5. Does BTRFS support copy-on-write? If yes, is it possible to disable it? Could such an approach have impact on performance and fragmentation?

Yes, the BTRFS support copy-on-write (COW) and it is possible to disable it for a particular file or for the entire filesystem.

Disable it will make sense for workloads where COW could be very expensive such as in database workloads that do small random updates followed by sequential scans

6. Do changes to the file system immediately get written to the disk, when using BTRFS? How does this proceed?

No. Because of the COW instead of updating the data in its original location it load the data from the disk to memory, modify it there then write it elsewhere. This minimizes the risks of example power failure and/or partial update.

7. Did Rodeh et al. discover large differences in performance between BTRFS and ZFS in their comparisons? How/why?

They did not compare the performance of BTRFS and ZFS because that the ZFS is not a linux native filesystem.

8. Which workloads did Rodeh et al. use for their benchmark tests with FileBench? How did BTRFS compare to the more mature file systems in these tests?

They Used the FileBench toolkit with web, file, mail and OLTP personalities tests and all tests where done on HDD, SSD flash disk.

On the HDD the OLTP and file testes gave similar results for the three filesystems. The BTRFS suffer on mail test duo mail server uses fsync which is 4-8 times slower than Ext4. The Ext4 have the advantage even in web workload duo to its HTree data structure which is 2 levels deep while the XFS and BTRFS uses B-Tree which is much deeper.

On SSD again for the web and mail workloads BTRFS is less efficient than Ext4 and XFS because of the fsync implementation. OLTP workload works bad on BTRFS in its standards. But disabling COW gets the result to be similar as the other filesystems. On the file workload BTRFS does slightly better than the others.

## Source reference:

All answers is reffered to the paper "BTRFS: The Linux B-Tree Filesystem" By OHAD RODEH, JOSEF BACIK and CHRIS MASON

# TASK 2

For the task I choose to both create all tasks and make the threads ready before the simulation starts, that is because if the simulation starts and then the task to be created it will take time from the simulation time and I wanted to get as accuret result as possible.

When the simulation start in the first loop it will assign the tasks sequantialy to the threads which starts execute at once then added the rest of the tasks in qeue to be executed as soon as some execution is finishid. The tasks keep tracks of its own start time, and finish time, and booleans is assigned to know is task have been completely executed, interupted or have not starts its execution at all. When the simulation end it force all thread to stop catching interupted error in task and assigning the task as interupted.

There is not enough time in the simulation to execute all tasks, but some of them. While the tasks burst time is assigned randomly between 1 – 10 seconds and the simulation time is only 15 seconds. In all simulation there were 4 interupted tasks.  simulation #0 had 17 waited task while #1 and #4 hade 15 and #2 hade 16.

The jstack output givs a lot of information which is very hard to understand but looking closly we find the Thread pool threades with information about them as the priority. We even notice the current status of the threads which is in this case only sleeping.

```
"pool-1-thread-1" #14 prio=5 os_prio=0 cpu=1,45ms elapsed=12,63s tid=0x00007f4c48194300 nid=0x1cb01 waiting on condition  [0x00007f4bebcfa000]
   java.lang.Thread.State: TIMED_WAITING (sleeping)
        at java.lang.Thread.sleep(java.base@17.0.1/Native Method)
        at MultithreadedService$Task.run(MultithreadedService.java:49)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.java:1136)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor.java:635)
        at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

   Locked ownable synchronizers:
        - <0x0000000718e1fdb8> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"pool-1-thread-2" #15 prio=5 os_prio=0 cpu=1,35ms elapsed=12,63s tid=0x00007f4c48195340 nid=0x1cb02 waiting on condition  [0x00007f4bebbf9000]
   java.lang.Thread.State: TIMED_WAITING (sleeping)
        at java.lang.Thread.sleep(java.base@17.0.1/Native Method)
        at MultithreadedService$Task.run(MultithreadedService.java:49)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.java:1136)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor.java:635)
        at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

   Locked ownable synchronizers:
        - <0x0000000718e203a8> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"pool-1-thread-3" #16 prio=5 os_prio=0 cpu=1,22ms elapsed=12,63s tid=0x00007f4c48196340 nid=0x1cb03 waiting on condition  [0x00007f4bebaf8000]
   java.lang.Thread.State: TIMED_WAITING (sleeping)
        at java.lang.Thread.sleep(java.base@17.0.1/Native Method)
        at MultithreadedService$Task.run(MultithreadedService.java:49)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.java:1136)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor.java:635)
        at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

   Locked ownable synchronizers:
        - <0x0000000718e20628> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"pool-1-thread-4" #17 prio=5 os_prio=0 cpu=1,14ms elapsed=12,63s tid=0x00007f4c48197360 nid=0x1cb04 waiting on condition  [0x00007f4beb9f7000]
   java.lang.Thread.State: TIMED_WAITING (sleeping)
        at java.lang.Thread.sleep(java.base@17.0.1/Native Method)
        at MultithreadedService$Task.run(MultithreadedService.java:49)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.java:1136)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor.java:635)
        at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

   Locked ownable synchronizers:
```

## Simulation output:

Running simulation #0

Simulation results:

---

Completed tasks:

Task ID: 0, burstTime: 4000, StartTime: 0, FinishTime: 4003

Task ID: 1, burstTime: 8100, StartTime: 0, FinishTime: 8107

Task ID: 2, burstTime: 8500, StartTime: 0, FinishTime: 8507

Task ID: 3, burstTime: 2400, StartTime: 0, FinishTime: 2402

Task ID: 4, burstTime: 1800, StartTime: 2402, FinishTime: 4204

Task ID: 5, burstTime: 7200, StartTime: 4003, FinishTime: 11209

Task ID: 6, burstTime: 2900, StartTime: 4204, FinishTime: 7106

Task ID: 7, burstTime: 3100, StartTime: 7106, FinishTime: 10209

Task ID: 8, burstTime: 2100, StartTime: 8107, FinishTime: 10209

Task ID: 9, burstTime: 2200, StartTime: 8507, FinishTime: 10709

Interrupted tasks:

| 10 | 11 | 12 | 13

Waiting tasks:

| 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30

Running simulation #1

Simulation results:

---

Completed tasks:

Task ID: 0, burstTime: 9600, StartTime: 0, FinishTime: 9608

Task ID: 1, burstTime: 1100, StartTime: 0, FinishTime: 1101

Task ID: 2, burstTime: 2400, StartTime: 0, FinishTime: 2402

Task ID: 3, burstTime: 1200, StartTime: 0, FinishTime: 1201

Task ID: 4, burstTime: 2900, StartTime: 1101, FinishTime: 4004

Task ID: 5, burstTime: 5800, StartTime: 1201, FinishTime: 7006

Task ID: 6, burstTime: 2200, StartTime: 2402, FinishTime: 4604

Task ID: 7, burstTime: 4800, StartTime: 4004, FinishTime: 8808

Task ID: 8, burstTime: 2800, StartTime: 4604, FinishTime: 7407

Task ID: 9, burstTime: 3300, StartTime: 7006, FinishTime: 10309

Task ID: 10, burstTime: 7500, StartTime: 7407, FinishTime: 14913

Task ID: 12, burstTime: 1100, StartTime: 9608, FinishTime: 10709

Interrupted tasks:

| 11 | 13 | 14 | 15

Waiting tasks:

| 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30

Running simulation #2

Simulation results:

---

Completed tasks:

Task ID: 0, burstTime: 6200, StartTime: 0, FinishTime: 6204

Task ID: 1, burstTime: 3300, StartTime: 0, FinishTime: 3302

Task ID: 2, burstTime: 3400, StartTime: 0, FinishTime: 3402

Task ID: 3, burstTime: 8600, StartTime: 0, FinishTime: 8606

Task ID: 4, burstTime: 9800, StartTime: 3302, FinishTime: 13110

Task ID: 5, burstTime: 2700, StartTime: 3402, FinishTime: 6104

Task ID: 6, burstTime: 6300, StartTime: 6104, FinishTime: 12409

Task ID: 7, burstTime: 4300, StartTime: 6204, FinishTime: 10508

Task ID: 8, burstTime: 5400, StartTime: 8606, FinishTime: 14010

Task ID: 9, burstTime: 4200, StartTime: 10508, FinishTime: 14711

Task ID: 11, burstTime: 1100, StartTime: 13110, FinishTime: 14211

Interrupted tasks:

| 10 | 12 | 13 | 14

Waiting tasks:

| 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30

Running simulation #3

Simulation results:

---

Completed tasks:

Task ID: 0, burstTime: 4000, StartTime: 1, FinishTime: 4004

Task ID: 1, burstTime: 5900, StartTime: 1, FinishTime: 5906

Task ID: 2, burstTime: 5900, StartTime: 1, FinishTime: 5905

Task ID: 3, burstTime: 1800, StartTime: 1, FinishTime: 1802

Task ID: 4, burstTime: 3900, StartTime: 1802, FinishTime: 5705

Task ID: 5, burstTime: 1600, StartTime: 4004, FinishTime: 5605

Task ID: 6, burstTime: 8400, StartTime: 5605, FinishTime: 14010

Task ID: 7, burstTime: 2500, StartTime: 5705, FinishTime: 8207

Task ID: 8, burstTime: 6600, StartTime: 5905, FinishTime: 12510

Task ID: 9, burstTime: 8600, StartTime: 5906, FinishTime: 14513

Task ID: 10, burstTime: 6100, StartTime: 8207, FinishTime: 14312

Task ID: 11, burstTime: 2300, StartTime: 12510, FinishTime: 14812

Interrupted tasks:

| 12 | 13 | 14 | 15

Waiting tasks:

| 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30

---

Exiting...

## Jstak output:

2022-01-16 17:24:28

Full thread dump OpenJDK 64-Bit Server VM (17.0.1+12-Ubuntu-120.04 mixed mode, sharing):

Threads class SMR info:

_java_thread_list=0x00007f4bfc001e20, length=16, elements={

0x00007f4c480120d0, 0x00007f4c48164640, 0x00007f4c48165a20, 0x00007f4c4816bba0,

0x00007f4c4816cf50, 0x00007f4c4816e360, 0x00007f4c4816fd10, 0x00007f4c48171240,

0x00007f4c4817a6a0, 0x00007f4c48181e40, 0x00007f4c4818cc70, 0x00007f4c48194300,

0x00007f4c48195340, 0x00007f4c48196340, 0x00007f4c48197360, 0x00007f4bfc000e60

}

"main" #1 prio=5 os_prio=0 cpu=12645,57ms elapsed=12,67s tid=0x00007f4c480120d0 nid=0x1caef runnable [0x00007f4c4e2a7000]

java.lang.Thread.State: RUNNABLE

at MultithreadedService.runNewSimulation(MultithreadedService.java:100)

at MultithreadedService.main(MultithreadedService.java:163)

Locked ownable synchronizers:

- None

"Reference Handler" #2 daemon prio=10 os_prio=0 cpu=0,15ms elapsed=12,64s tid=0x00007f4c48164640 nid=0x1caf6 waiting on condition [0x00007f4c281b3000]

java.lang.Thread.State: RUNNABLE

at java.lang.ref.Reference.waitForReferencePendingList(java.base@17.0.1/Native Method)

at java.lang.ref.Reference.processPendingReferences(java.base@17.0.1/Reference.java:253)

at java.lang.ref.Reference$ReferenceHandler.run(java.base@17.0.1/Reference.java:215)

Locked ownable synchronizers:

- None

"Finalizer" #3 daemon prio=8 os_prio=0 cpu=0,19ms elapsed=12,64s tid=0x00007f4c48165a20 nid=0x1caf7 in Object.wait() [0x00007f4c197fe000]

java.lang.Thread.State: WAITING (on object monitor)

at java.lang.Object.wait(java.base@17.0.1/Native Method)

- waiting on <0x0000000718e02fa0> (a java.lang.ref.ReferenceQueue$Lock)

at java.lang.ref.ReferenceQueue.remove(java.base@17.0.1/ReferenceQueue.java:155)

- locked <0x0000000718e02fa0> (a java.lang.ref.ReferenceQueue$Lock)

at java.lang.ref.ReferenceQueue.remove(java.base@17.0.1/ReferenceQueue.java:176)

at java.lang.ref.Finalizer$FinalizerThread.run(java.base@17.0.1/Finalizer.java:172)

Locked ownable synchronizers:

- None

"Signal Dispatcher" #4 daemon prio=9 os_prio=0 cpu=0,32ms elapsed=12,64s tid=0x00007f4c4816bba0 nid=0x1caf8 waiting on condition [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

Locked ownable synchronizers:

- None

"Service Thread" #5 daemon prio=9 os_prio=0 cpu=0,12ms elapsed=12,64s tid=0x00007f4c4816cf50 nid=0x1caf9 runnable [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

Locked ownable synchronizers:

- None

"Monitor Deflation Thread" #6 daemon prio=9 os_prio=0 cpu=0,22ms elapsed=12,64s tid=0x00007f4c4816e360 nid=0x1cafa runnable [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

Locked ownable synchronizers:

- None

"C2 CompilerThread0" #7 daemon prio=9 os_prio=0 cpu=4,29ms elapsed=12,64s tid=0x00007f4c4816fd10 nid=0x1cafb waiting on condition [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

No compile task

Locked ownable synchronizers:

- None

"C1 CompilerThread0" #10 daemon prio=9 os_prio=0 cpu=3,67ms elapsed=12,64s tid=0x00007f4c48171240 nid=0x1cafc waiting on condition [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

No compile task

Locked ownable synchronizers:

- None

"Sweeper thread" #11 daemon prio=9 os_prio=0 cpu=0,03ms elapsed=12,63s tid=0x00007f4c4817a6a0 nid=0x1cafd runnable [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

Locked ownable synchronizers:

- None

"Notification Thread" #12 daemon prio=9 os_prio=0 cpu=0,04ms elapsed=12,63s tid=0x00007f4c48181e40 nid=0x1cafe runnable [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

Locked ownable synchronizers:

- None

"Common-Cleaner" #13 daemon prio=8 os_prio=0 cpu=0,15ms elapsed=12,63s tid=0x00007f4c4818cc70 nid=0x1cb00 in Object.wait() [0x00007f4bebdfb000]

java.lang.Thread.State: TIMED_WAITING (on object monitor)

at java.lang.Object.wait(java.base@17.0.1/Native Method)

- waiting on <0x0000000718e18690> (a java.lang.ref.ReferenceQueue$Lock)

at java.lang.ref.ReferenceQueue.remove(java.base@17.0.1/ReferenceQueue.java:155)

- locked <0x0000000718e18690> (a java.lang.ref.ReferenceQueue$Lock)

at jdk.internal.ref.CleanerImpl.run(java.base@17.0.1/CleanerImpl.java:140)

at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

at jdk.internal.misc.InnocuousThread.run(java.base@17.0.1/InnocuousThread.java:162)

Locked ownable synchronizers:

- None

"pool-1-thread-1" #14 prio=5 os_prio=0 cpu=1,45ms elapsed=12,63s tid=0x00007f4c48194300 nid=0x1cb01 waiting on condition [0x00007f4bebcfa000]

java.lang.Thread.State: TIMED_WAITING (sleeping)

at java.lang.Thread.sleep(java.base@17.0.1/Native Method)

at MultithreadedService$Task.run(MultithreadedService.java:49)

at java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.java:1136)

at
java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor
.java:635)

at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

Locked ownable synchronizers:

- <0x0000000718e1fdb8> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"pool-1-thread-2" #15 prio=5 os_prio=0 cpu=1,35ms elapsed=12,63s
tid=0x00007f4c48195340 nid=0x1cb02 waiting on condition [0x00007f4bebbf9000]

java.lang.Thread.State: TIMED_WAITING (sleeping)

at java.lang.Thread.sleep(java.base@17.0.1/Native Method)

at MultithreadedService$Task.run(MultithreadedService.java:49)

at
java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.j
ava:1136)

at
java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor
.java:635)

at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

Locked ownable synchronizers:

- <0x0000000718e203a8> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"pool-1-thread-3" #16 prio=5 os_prio=0 cpu=1,22ms elapsed=12,63s
tid=0x00007f4c48196340 nid=0x1cb03 waiting on condition [0x00007f4bebaf8000]

java.lang.Thread.State: TIMED_WAITING (sleeping)

at java.lang.Thread.sleep(java.base@17.0.1/Native Method)

at MultithreadedService$Task.run(MultithreadedService.java:49)

at
java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.j
ava:1136)

at
java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor
.java:635)

at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

Locked ownable synchronizers:

- <0x0000000718e20628> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"pool-1-thread-4" #17 prio=5 os_prio=0 cpu=1,14ms elapsed=12,63s
tid=0x00007f4c48197360 nid=0x1cb04 waiting on condition [0x00007f4beb9f7000]

java.lang.Thread.State: TIMED_WAITING (sleeping)

at java.lang.Thread.sleep(java.base@17.0.1/Native Method)

at MultithreadedService$Task.run(MultithreadedService.java:49)

at
java.util.concurrent.ThreadPoolExecutor.runWorker(java.base@17.0.1/ThreadPoolExecutor.j
ava:1136)

at
java.util.concurrent.ThreadPoolExecutor$Worker.run(java.base@17.0.1/ThreadPoolExecutor
.java:635)

at java.lang.Thread.run(java.base@17.0.1/Thread.java:833)

Locked ownable synchronizers:

- <0x0000000718e208a8> (a java.util.concurrent.ThreadPoolExecutor$Worker)

"Attach Listener" #18 daemon prio=9 os_prio=0 cpu=0,21ms elapsed=0,10s
tid=0x00007f4bfc000e60 nid=0x1cb53 waiting on condition [0x0000000000000000]

java.lang.Thread.State: RUNNABLE

Locked ownable synchronizers:

- None

"VM Thread" os_prio=0 cpu=0,70ms elapsed=12,64s tid=0x00007f4c48160720 nid=0x1caf5 runnable

"GC Thread#0" os_prio=0 cpu=0,11ms elapsed=12,64s tid=0x00007f4c480621c0 nid=0x1caf0 runnable

"G1 Main Marker" os_prio=0 cpu=0,13ms elapsed=12,64s tid=0x00007f4c48072e50 nid=0x1caf1 runnable

"G1 Conc#0" os_prio=0 cpu=0,09ms elapsed=12,64s tid=0x00007f4c48073db0 nid=0x1caf2 runnable

"G1 Refine#0" os_prio=0 cpu=0,10ms elapsed=12,64s tid=0x00007f4c48130090 nid=0x1caf3 runnable

"G1 Service" os_prio=0 cpu=0,67ms elapsed=12,64s tid=0x00007f4c48130f80 nid=0x1caf4 runnable

"VM Periodic Task Thread" os_prio=0 cpu=2,04ms elapsed=12,63s tid=0x00007f4c48183780 nid=0x1caff waiting on condition

JNI global refs: 6, weak refs: 0