CS344 Operating Systems Lab
Assignment - 4

Group Number : M14

Akshat Jain - 200123005
Jwalit Bharatkumar Devalia - 200123026
Varun Bhardwaj - 200123068

Part - 1

For this assignment, we are going to use **ZFS** and **EXT4** File Systems.

EXT4 File System

The performance and capacity of the EXT4 file system are its main priorities. Instead of fixed-size blocks, extents are used in this system to allocate storage. Extents are simply identified by their beginning and ending locations on the hard drive. This method of storing the essential location of data in files avoids fragmentation of the memory allocated by the EXT4 file system and enables the storage of the file's location of data with the use of a few pointers rather than a pointer referring to every block of memory that the file occupies.Additionally, it uses delayed allocation, which boosts performance and makes it easier for the file system to allocate contiguous blocks of memory because it already knows how much memory it needs to map before allocating any memory.

ZFS (Zettabyte File System)

Sun Microsystems Inc. developed ZFS, a local file system and logical volume manager, to guide and regulate the placement, storage, and retrieval of data in enterprise-class computing systems. The ZFS file system and volume manager is characterized by **data integrity**, **high scalability** and built-in storage features such as:

- **Replication** - the action of creating a copy or replica
- **Snapshots** - a set of reference markers for data at a particular point in time.
- **Compression** - a reduction in the number of bits needed to represent data.
- **Deduplication** - a process that eliminates redundant copies of data and reduces storage overhead.
- **Clones** - an identical copy of something.

● **Data protection** - the process of safeguarding important information from corruption and/or loss.

Here is a description of both the features we have picked to compare for this assignment:

Deduplication:

Deduplication gets rid of duplicate data. In some cases if there is a lot of duplicate content, with or without minor changes, this can clear up a lot of space on the hard drive and is extremely helpful. The strategy should only be utilised in the most dire circumstances, though, due to the high cost of the overhead computations. Deduplication is accomplished by hashing (using a secure hash, such as SHA256) a piece of the data to get an estimated unique signature and storing them in a hash table. When new data's signature is compared to values that already exist in the hash table, data with a pre-existing signature is taken into account to be a replica of the data whose signature matches it. Deduplication can be carried out at many levels, with a growing trade-off between overhead computations and the space saved by not copying redundant data, depending on the amount of data that needs to be hashed into a signature. At the file, block, and byte levels, respectively. Depending on whether the process occurs as the data is being written or if the copies are hashed and eliminated when the CPU is free, deduplication can be either synchronous or asynchronous. Block-level synchronous deduplication serves as the foundation for ZFS's deduplication capabilities. Deduplication is not supported by EXT4.

Large File creation:

The **EXT4** file system provides a maximum volume of **1 EiB (ExbiByte)=2^60Bytes** and a maximum file size of **16 TiB (TebiBytes)=2^44 Bytes** with standard **4KiB blocks** and **48 bit** block addressing. Only file systems and file sizes up to **2 TiB** are supported by **EXT3**. File systems up to **16 TiB** in size are supported by **ZFS**. Very huge files can be created and handled very efficiently with **EXT4**. This is so that Extents can preserve and retrieve enormous mappings of the data blocks included in big files more quickly and with less storage space. This new extent-based mapping method can operate properly and effectively thanks to additional **EXT4** features. **EXT4**'s multiblock allocation feature makes it simple to allocate contiguous blocks of memory while saving money by allocating multiple blocks at once as opposed to one block per call. This works in conjunction with delayed allocation, which doesn't always write to disc but instead makes a note of the data to be written before using multiblock allocation to write a substantial quantity of data into a contiguous memory segment.

Installation & Setup of File Systems :

ZFS

For our study, ZFS was installed on a USB device. The steps used for the same are :

1) ZFS utilities were installed using the command -

sudo apt install zfs-fuse

2) ZFS daemon is started using the command -

/etc/init.d/zfs-fuse start

3) The USB drive path was obtained using -

sudo fdisk -l

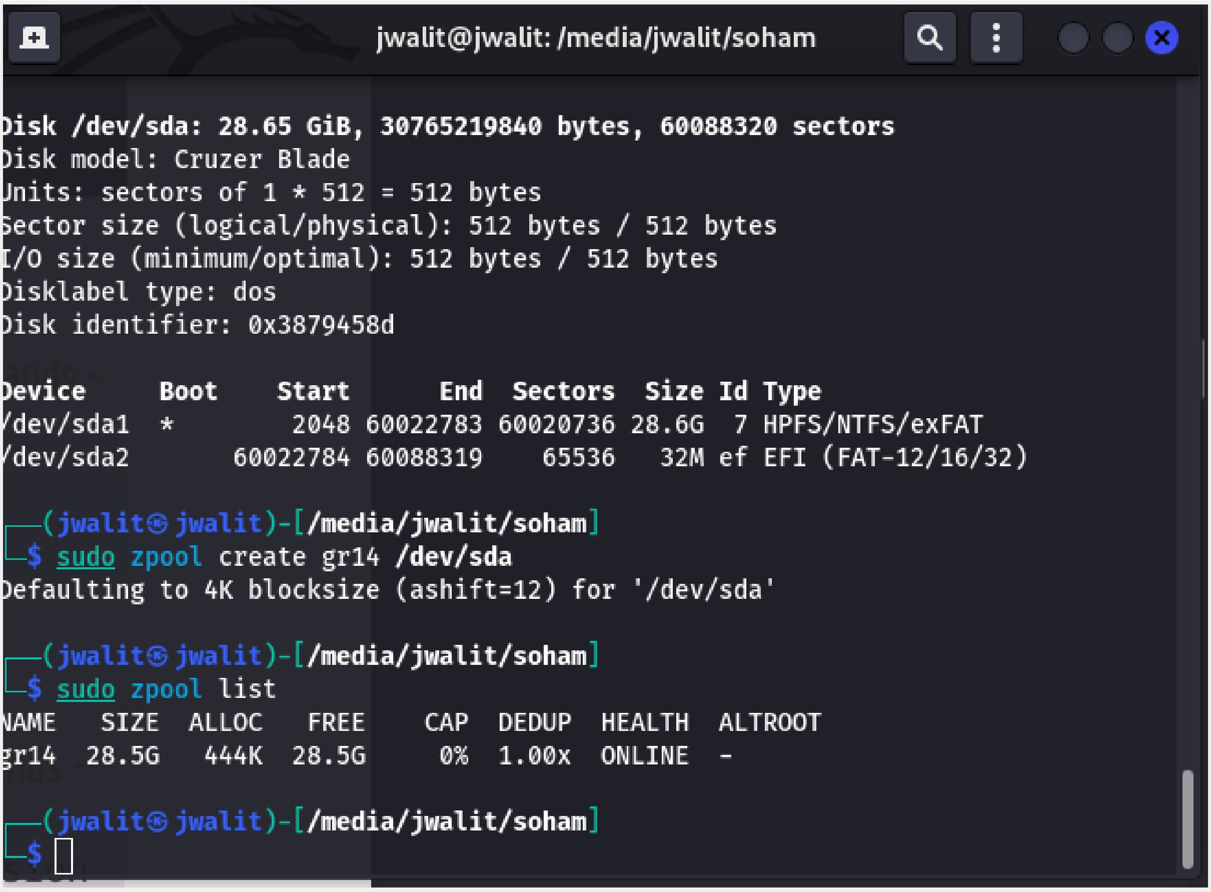
4) A ZFS pool was created on the USB drive using -

sudo zpool create gr14 /dev/sda

NOTE: After running this command, the USB drive was mounted in the root directory as the folder m14. All subsequent workload operations are done on this folder from vdbench.

5) The created zfs pool can be seen using command -

sudo zpool list



```
Disk /dev/sda: 28.65 GiB, 30765219840 bytes, 60088320 sectors
Disk model: Cruzer Blade
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x3879458d

Device     Boot    Start      End  Sectors  Size Id Type
/dev/sda1  *        2048 60022783 60020736 28.6G  7 HPFS/NTFS/exFAT
/dev/sda2       60022784 60088319    65536   32M ef EFI (FAT-12/16/32)

┌──(jwalit㉿jwalit)-[/media/jwalit/soham]
└─$ sudo zpool create gr14 /dev/sda
Defaulting to 4K blocksize (ashift=12) for '/dev/sda'

┌──(jwalit㉿jwalit)-[/media/jwalit/soham]
└─$ sudo zpool list
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
gr14  28.5G   444K  28.5G     0%  1.00x  ONLINE  -

┌──(jwalit㉿jwalit)-[/media/jwalit/soham]
└─$ 
```

6) Deduplication feature can be enabled/disabled using the following commands -

```
sudo zfs set dedup=on gr14
```

EXT4

1). EXT4 is preinstalled and the default file system on Ubuntu. The following instructions tell you how to format a disk and set up the EXT4 filesystem on it.

2). Open the Application “Disks”:

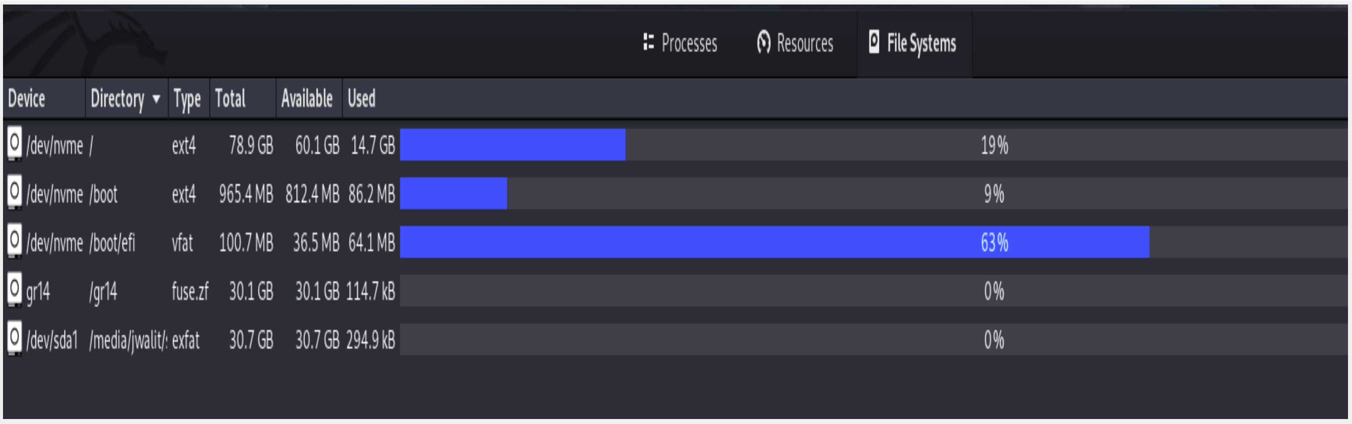
3). Choose the disk you want **(SHOULD HAVE AT LEAST 5 GB DISK SPACE)** and click the “Gear” icon. Then choose “Format Partition”:

4). Then, choose a name for the new disk and choose the Ext4 option (first option in my computer). Check the “Erase” switch. Then click next:

5). Then Select “Format”:

6). Once the disk is formatted, make sure that the disk is mounted. If not, then open “mount options” for the disk (Gear Icon->Edit Mount Options) and then uncheck “User Session Defaults” and check “Mount on system startup”. Then Reboot:

After completing the above steps, we can see the created file systems by going into the system monitor in ubuntu. A screenshot of the same is attached below



Device	Directory ▾	Type	Total	Available	Used	
/dev/nvme	/	ext4	78.9 GB	60.1 GB	14.7 GB	19%
/dev/nvme	/boot	ext4	965.4 MB	812.4 MB	86.2 MB	9%
/dev/nvme	/boot/efi	vfat	100.7 MB	36.5 MB	64.1 MB	63%
gr14	/gr14	fuse.zf	30.1 GB	30.1 GB	114.7 kB	0%
/dev/sda1	/media/jwalit/	exfat	30.7 GB	30.7 GB	294.9 kB	0%

Part - 2

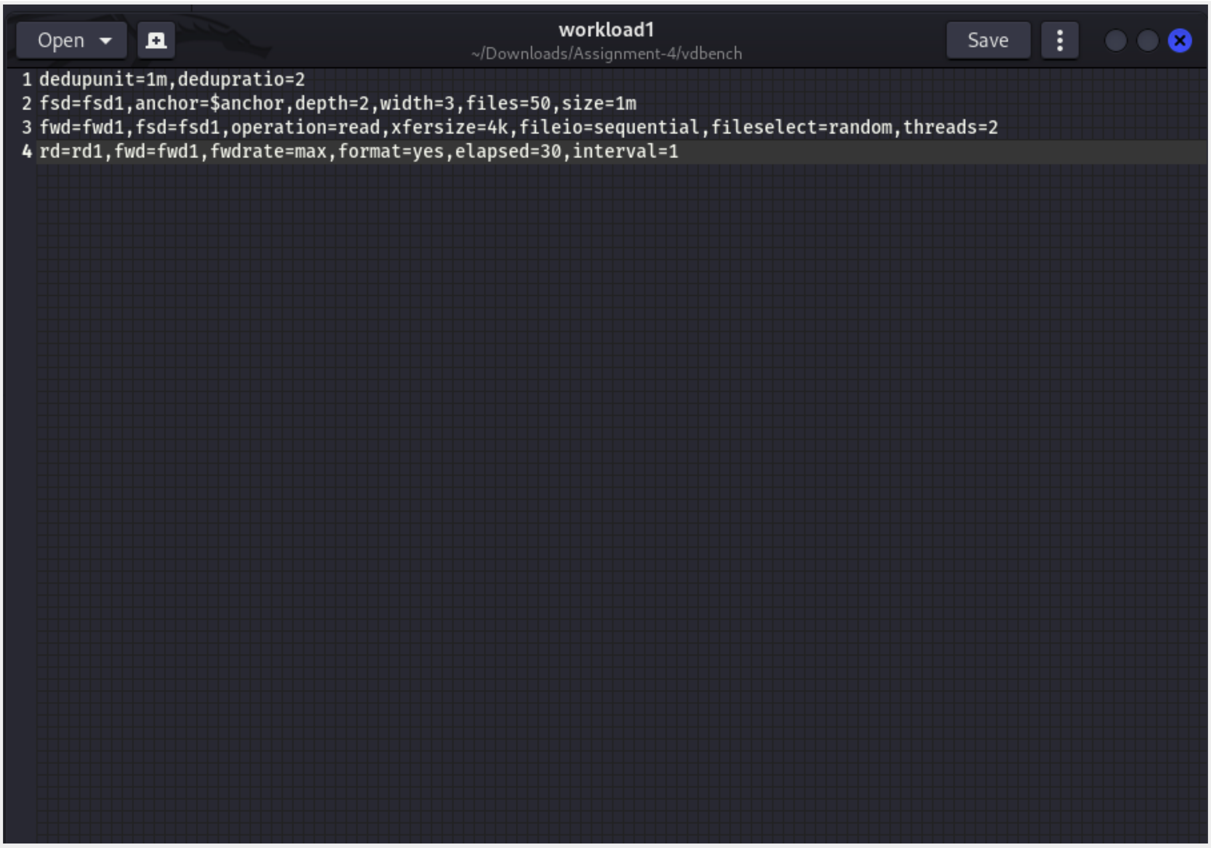
Deduplication:

1) ZFS has a data deduplication feature which we turned on using (**gr14** is the name of the zfs pool we set up):

sudo zfs set dedup=on zfs_pool

2) We created the following workload for data deduplication (**workload1**). We will use this workload to compare the space occupied by the new files in ZFS with the space occupied in ext4.

3)



```
dedupunit=1m,dedupratio=2
fsd=fsd1,anchor=$anchor,depth=2,width=3,files=50,size=1m
fwd=fwd1,fsd=fsd1,operation=read,xfersize=4k,fileio=sequential,fileselect=random,threads=2
rd=rd1,fwd=fwd1,fwdrate=max,format=yes,elapsed=30,interval=1
```

4) In essence, we are producing **450** files (**50*3*3**) each of size **1MB**, in a nested folder structure with **depth 2** and **width 3**. Then, for thirty seconds, these files are read consecutively to track statistics (although this part is not important since the deduplication is done during file creation).

5) **dedupunit** is set to **1MB** and **dedupratio** is set to **2. dedupratio** is the ratio of the total number of blocks (of size **dedupunit**) with the number of blocks containing unique data. **dedupunit** on the other hand is the size of the block which will be compared with pre-existing blocks to check for duplicates. **We set it to 1MB because this is the size of one file.** So basically, half of the files will be duplicates of the other half.

6) We will be running this workload by setting anchor to the directory of the ZFS pool.

sudo ./vdbench -f workload1 anchor=/gr14

7) By setting anchor to the directory of the folder pointing to the ext4 drive, we perform this workload on the **ext4** file system:

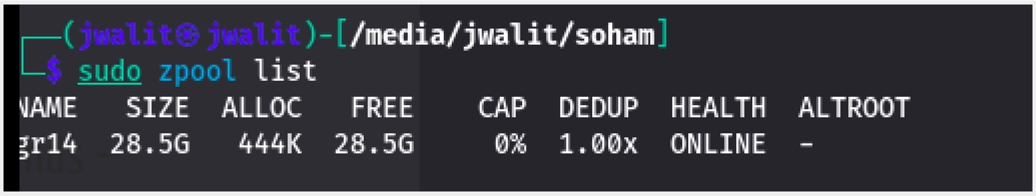
sudo ./vdbench -f workload1 anchor=/mnt

8) Results are as follows:

- **ZFS**:

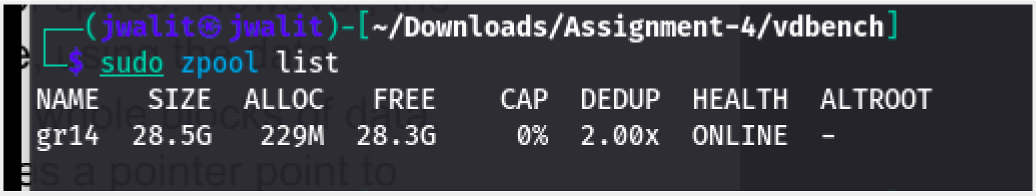
1. Initially, the empty **ZFS** folder had **444 KB** of data
2. The **ZFS** folder contained **229.9** MB of data when the workload had been completed.
3. We observed a deduplication ratio of **2.00x**
4. This indicates that **229.5** MB of space was consumed by the new files. The desired space, however, is **450MB** (**1MB*450**). Therefore, when duplicates are discovered, ZFS merely makes a pointer to the old data utilising the data deduplication capability rather than maintaining entire blocks of data.

Before Workload:



```
┌──(jwalit㉿jwalit)-[/media/jwalit/soham]
└─$ sudo zpool list
NAME    SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
gr14  28.5G   444K  28.5G     0%  1.00x  ONLINE  -
```

After Workload:

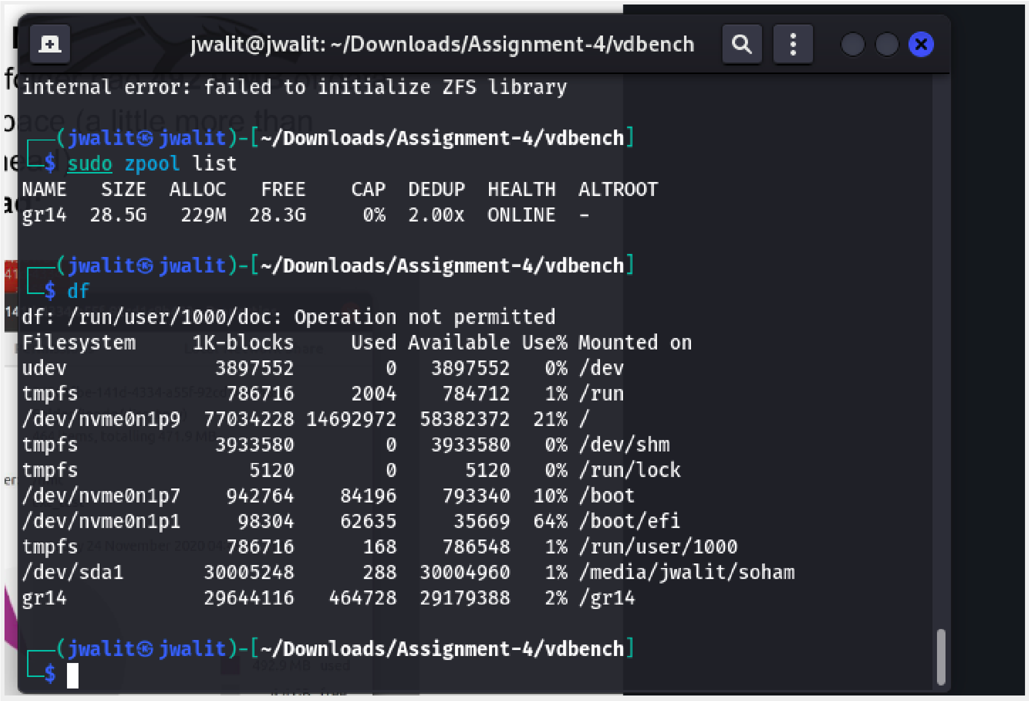


```
┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ sudo zpool list
NAME    SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
gr14  28.5G   229M  28.3G     0%  2.00x  ONLINE  -
```

- **Ext4:**

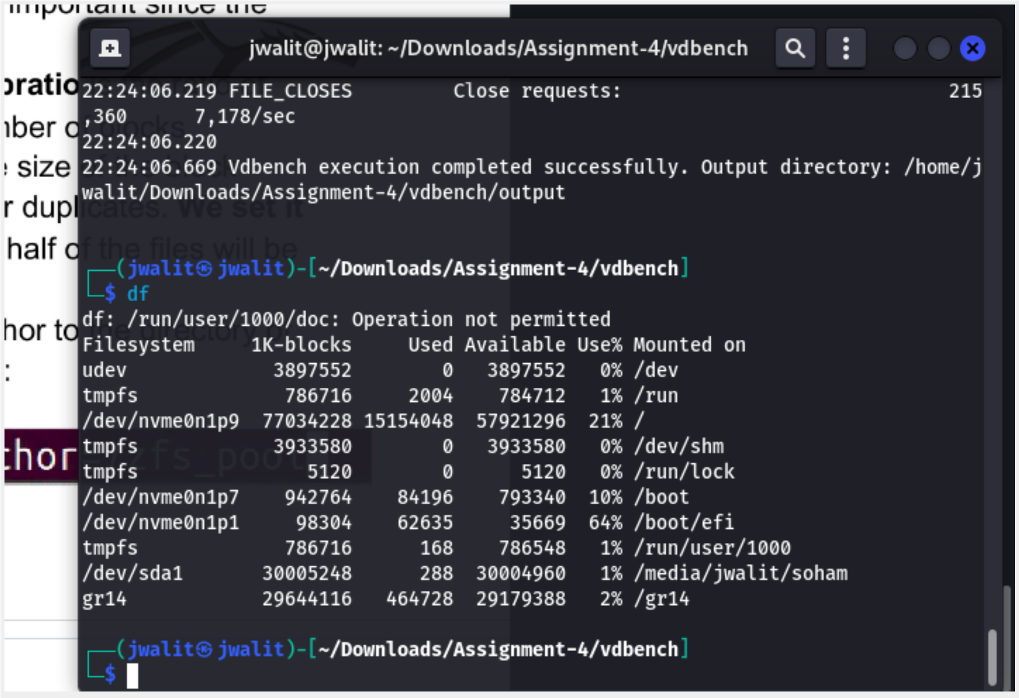
1. The initial data in the ext4 folder was **14692972 KB**.
2. After completing the workload, **15154048 KB** of data were present in the ext4 folder.
3. As a result, the new files consumed **~461 MB** of space, which was slightly more than intended due to metadata overhead.

Before Workload:



```
internal error: failed to initialize ZFS library

┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ sudo zpool list
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
gr14  28.5G   229M  28.3G     0%  2.00x  ONLINE  -

┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ df
df: /run/user/1000/doc: Operation not permitted
Filesystem     1K-blocks     Used Available Use% Mounted on
udev             3897552        0   3897552   0% /dev
tmpfs             786716     2004    784712   1% /run
/dev/nvme0n1p9  77034228 14692972  58382372  21% /
tmpfs            3933580        0   3933580   0% /dev/shm
tmpfs               5120        0      5120   0% /run/lock
/dev/nvme0n1p7    942764    84196    793340  10% /boot
/dev/nvme0n1p1     98304    62635     35669  64% /boot/efi
tmpfs             786716      168    786548   1% /run/user/1000
/dev/sda1       30005248      288  30004960   1% /media/jwalit/soham
gr14            29644116   464728  29179388   2% /gr14

┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ 
```

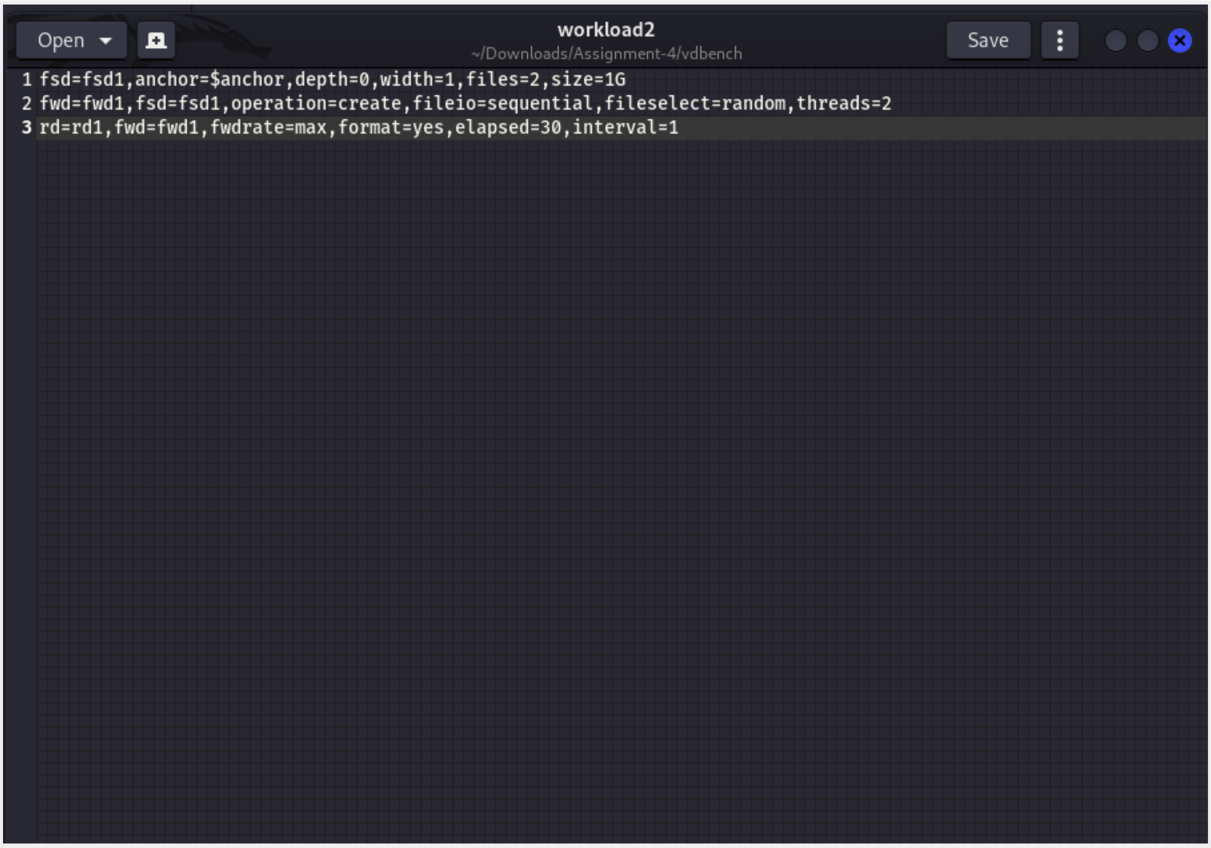
After Workload:



```
22:24:06.219 FILE_CLOSES         Close requests:                         215
,360        7,178/sec
22:24:06.220
22:24:06.669 Vdbench execution completed successfully. Output directory: /home/j
walit/Downloads/Assignment-4/vdbench/output


┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ df
df: /run/user/1000/doc: Operation not permitted
Filesystem     1K-blocks     Used Available Use% Mounted on
udev             3897552        0   3897552   0% /dev
tmpfs             786716     2004    784712   1% /run
/dev/nvme0n1p9  77034228 15154048  57921296  21% /
tmpfs            3933580        0   3933580   0% /dev/shm
tmpfs               5120        0      5120   0% /run/lock
/dev/nvme0n1p7    942764    84196    793340  10% /boot
/dev/nvme0n1p1     98304    62635     35669  64% /boot/efi
tmpfs             786716      168    786548   1% /run/user/1000
/dev/sda1       30005248      288  30004960   1% /media/jwalit/soham
gr14            29644116   464728  29179388   2% /gr14

┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ 
```

Large File Creation:

1. Using our workload, it is obvious that **ext4** optimises the creation of large files better than **ZFS**.
2. We created the following workload for testing large file creation (**workload2**):



```
fsd=fsd1,anchor=$anchor,depth=0,width=1,files=2,size=1G
fwd=fwd1,fsd=fsd1,operation=create,fileio=sequential,fileselect=random,threads=2
rd=rd1,fwd=fwd1,fwdrate=max,format=yes,elapsed=30,interval=1
```

3. Here, we're making two **1GB** files and placing them in a folder. Since we are testing the creation of files, the operation used is "**create**."
4. By setting anchor equal to the directory referring to the **ZFS** pool(i.e. **/gr14** in our case), we execute this on the ZFS file system:

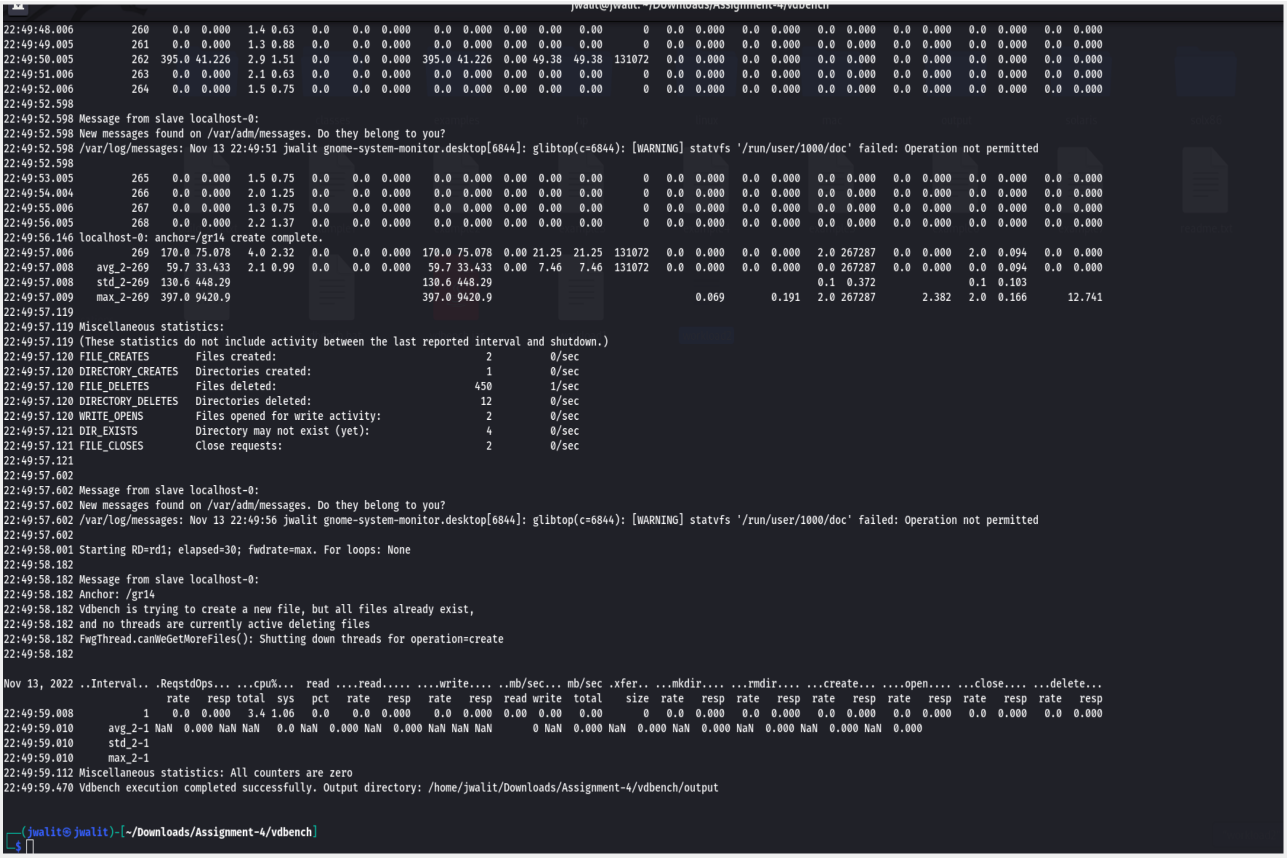
sudo ./vdbench -f workload2 anchor=/gr14

5. By setting anchor equal to the directory pointing to the ext4 drive, we run this workload on the **ext4** file system:

sudo ./vdbench -f workload2 anchor=/mnt

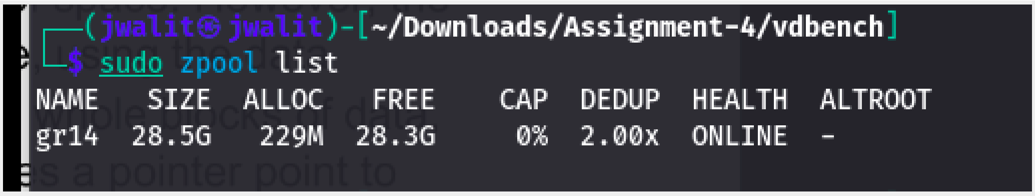
6. Results:

- **ZFS**:



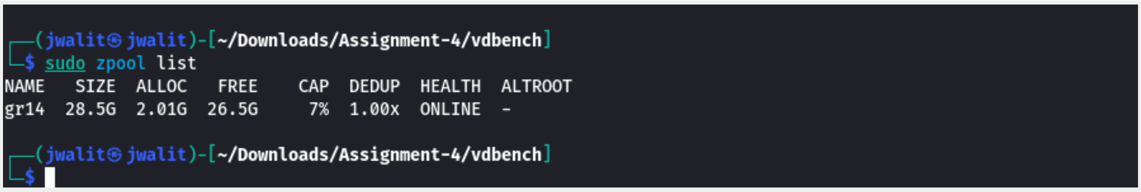
```
22:49:48.006          260    0.0  0.000   1.4 0.63   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:49.005          261    0.0  0.000   1.3 0.88   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:50.005          262  395.0 41.226   2.9 1.51   0.0    0.0  0.000  395.0 41.226  0.00 49.38  49.38  131072   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:51.006          263    0.0  0.000   2.1 0.63   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:52.006          264    0.0  0.000   1.5 0.75   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:52.598
22:49:52.598 Message from slave localhost-0:
22:49:52.598 New messages found on /var/adm/messages. Do they belong to you?
22:49:52.598 /var/log/messages: Nov 13 22:49:51 jwalit gnome-system-monitor.desktop[6844]: glibtop(c=6844): [WARNING] statvfs '/run/user/1000/doc' failed: Operation not permitted
22:49:52.598
22:49:53.005          265    0.0  0.000   1.5 0.75   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:54.004          266    0.0  0.000   2.0 1.25   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:55.006          267    0.0  0.000   1.3 0.75   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:56.005          268    0.0  0.000   2.2 1.37   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:56.146 localhost-0: anchor=/gr14 create complete.
22:49:57.006          269  170.0 75.078   4.0 2.32   0.0    0.0  0.000  170.0 75.078  0.00 21.25  21.25  131072   0.0  0.000   0.0  0.000   2.0 267287   0.0  0.000   2.0  0.094   0.0  0.000
22:49:57.008    avg_2-269   59.7 33.433   2.1 0.99   0.0    0.0  0.000   59.7 33.433  0.00  7.46   7.46  131072   0.0  0.000   0.0  0.000   0.0 267287   0.0  0.000   0.0  0.094   0.0  0.000
22:49:57.008    std_2-269  130.6 448.29                                 130.6 448.29                                                             0.1  0.372               0.1  0.103
22:49:57.009    max_2-269  397.0 9420.9                                 397.0 9420.9                                      0.069        0.191   2.0 267287        2.382   2.0  0.166       12.741
22:49:57.119
22:49:57.119 Miscellaneous statistics:
22:49:57.119 (These statistics do not include activity between the last reported interval and shutdown.)
22:49:57.120 FILE_CREATES       Files created:                                2         0/sec
22:49:57.120 DIRECTORY_CREATES  Directories created:                          1         0/sec
22:49:57.120 FILE_DELETES       Files deleted:                              450         1/sec
22:49:57.120 DIRECTORY_DELETES  Directories deleted:                         12         0/sec
22:49:57.120 WRITE_OPENS        Files opened for write activity:              2         0/sec
22:49:57.121 DIR_EXISTS         Directory may not exist (yet):                4         0/sec
22:49:57.121 FILE_CLOSES        Close requests:                               2         0/sec
22:49:57.121
22:49:57.602
22:49:57.602 Message from slave localhost-0:
22:49:57.602 New messages found on /var/adm/messages. Do they belong to you?
22:49:57.602 /var/log/messages: Nov 13 22:49:56 jwalit gnome-system-monitor.desktop[6844]: glibtop(c=6844): [WARNING] statvfs '/run/user/1000/doc' failed: Operation not permitted
22:49:57.602
22:49:58.001 Starting RD=rd1; elapsed=30; fwdrate=max. For loops: None
22:49:58.182
22:49:58.182 Message from slave localhost-0:
22:49:58.182 Anchor: /gr14
22:49:58.182 Vdbench is trying to create a new file, but all files already exist,
22:49:58.182 and no threads are currently active deleting files
22:49:58.182 FwgThread.canWeGetMoreFiles(): Shutting down threads for operation=create
22:49:58.182

Nov 13, 2022 ..Interval.. .ReqstdOps... ...cpu%...  read ....read..... ....write.... ..mb/sec... mb/sec .xfer.. ...mkdir.... ...rmdir.... ...create... ....open.... ...close.... ...delete...
                             rate   resp total  sys   pct   rate   resp   rate   resp  read write  total    size  rate   resp  rate   resp  rate   resp  rate   resp  rate   resp  rate   resp
22:49:59.008            1    0.0  0.000   3.4 1.06   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:49:59.010      avg_2-1 NaN  0.000 NaN NaN   0.0 NaN  0.000 NaN  0.000 NaN NaN NaN       0 NaN  0.000 NaN  0.000 NaN  0.000 NaN  0.000 NaN  0.000 NaN  0.000
22:49:59.010      std_2-1
22:49:59.010      max_2-1
22:49:59.112 Miscellaneous statistics: All counters are zero
22:49:59.470 Vdbench execution completed successfully. Output directory: /home/jwalit/Downloads/Assignment-4/vdbench/output


┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ 
```

Before command



```
┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ sudo zpool list
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
gr14  28.5G   229M  28.3G     0%  2.00x  ONLINE  -
```

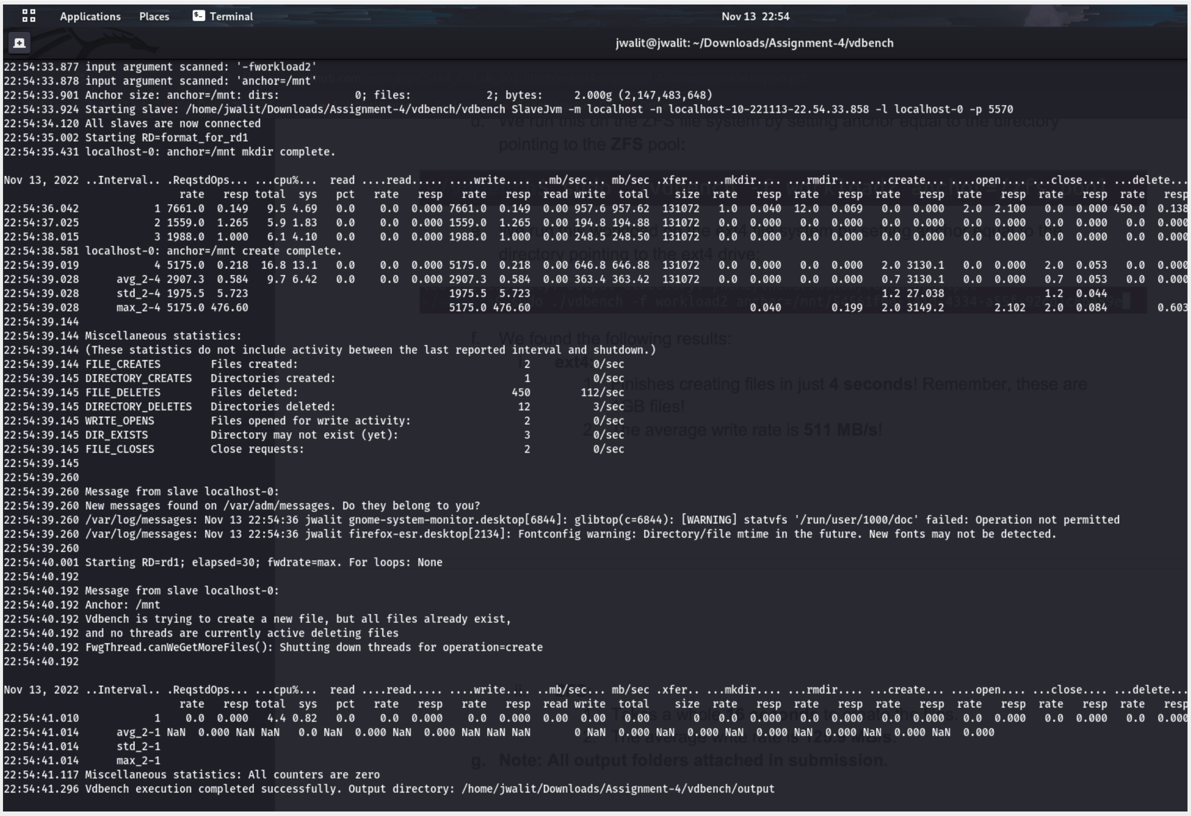
After command



```
┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ sudo zpool list
NAME   SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
gr14  28.5G  2.01G  26.5G     7%  1.00x  ONLINE  -

┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ 
```

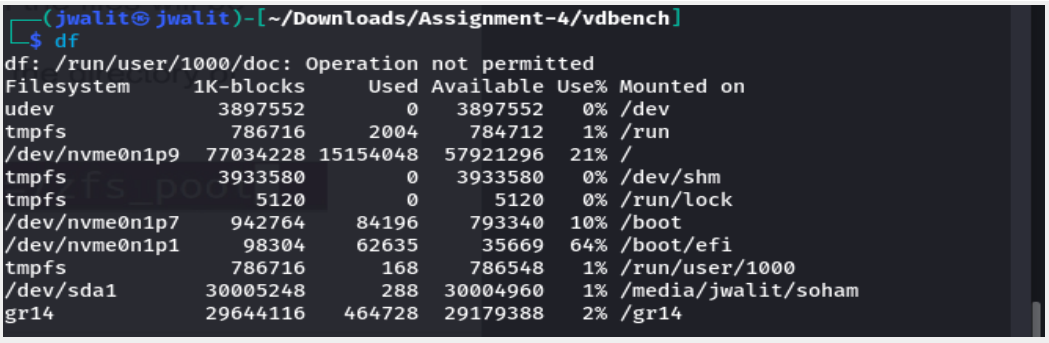
1. Takes 10 seconds to create the files.
2. The average write rate is **200MB/s**

- **Ext4**:



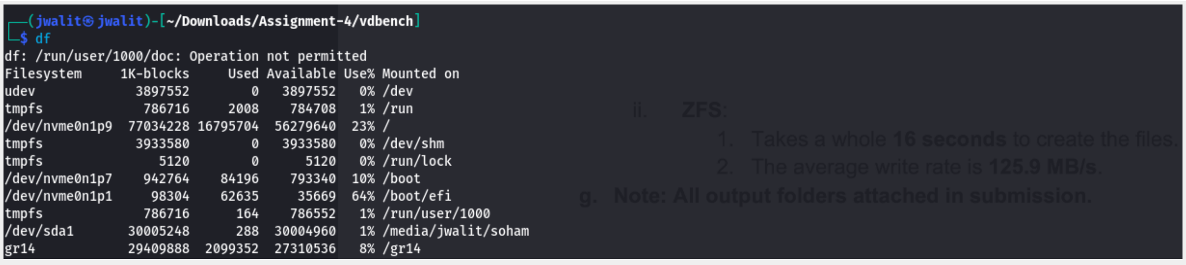
```
22:54:33.877 input argument scanned: '-fworkload2'
22:54:33.878 input argument scanned: 'anchor=/mnt'
22:54:33.901 Anchor size: anchor=/mnt: dirs:           0; files:           2; bytes:     2.000g (2,147,483,648)
22:54:33.924 Starting slave: /home/jwalit/Downloads/Assignment-4/vdbench/vdbench SlaveJvm -m localhost -n localhost-10-221113-22.54.33.858 -l localhost-0 -p 5570
22:54:34.120 All slaves are now connected
22:54:35.002 Starting RD=format_for_rd1
22:54:35.431 localhost-0: anchor=/mnt mkdir complete.

Nov 13, 2022 ..Interval.. .ReqstdOps... ...cpu%...  read ....read..... ....write.... ..mb/sec... mb/sec .xfer.. ...mkdir.... ...rmdir.... ...create... ....open.... ...close.... ...delete...
                              rate   resp total  sys   pct   rate   resp   rate   resp  read write  total    size  rate   resp  rate   resp  rate   resp  rate   resp  rate   resp  rate   resp
22:54:36.042           1 7661.0  0.149   9.5 4.69   0.0    0.0  0.000 7661.0  0.149  0.00 957.6 957.62  131072   1.0  0.040  12.0  0.069   0.0  0.000   2.0  2.100   0.0  0.000 450.0  0.138
22:54:37.025           2 1559.0  1.265   5.9 1.83   0.0    0.0  0.000 1559.0  1.265  0.00 194.8 194.88  131072   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:54:38.015           3 1988.0  1.000   6.1 4.10   0.0    0.0  0.000 1988.0  1.000  0.00 248.5 248.50  131072   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:54:38.581 localhost-0: anchor=/mnt create complete.
22:54:39.019           4 5175.0  0.218  16.8 13.1   0.0    0.0  0.000 5175.0  0.218  0.00 646.8 646.88  131072   0.0  0.000   0.0  0.000   2.0 3130.1   0.0  0.000   2.0  0.053   0.0  0.000
22:54:39.028     avg_2-4 2907.3  0.584   9.7 6.42   0.0    0.0  0.000 2907.3  0.584  0.00 363.4 363.42  131072   0.0  0.000   0.0  0.000   0.7 3130.1   0.0  0.000   0.7  0.053   0.0  0.000
22:54:39.028     std_2-4 1975.5  5.723                            1975.5  5.723                                                        1.2 27.038               1.2  0.044
22:54:39.028     max_2-4 5175.0 476.60                            5175.0 476.60                                0.040         0.199   2.0 3149.2        2.102   2.0  0.084         0.603
22:54:39.144
22:54:39.144 Miscellaneous statistics:
22:54:39.144 (These statistics do not include activity between the last reported interval and shutdown.)
22:54:39.144 FILE_CREATES        Files created:                              2        0/sec
22:54:39.145 DIRECTORY_CREATES   Directories created:                        1        0/sec
22:54:39.145 FILE_DELETES        Files deleted:                            450      112/sec
22:54:39.145 DIRECTORY_DELETES   Directories deleted:                       12        3/sec
22:54:39.145 WRITE_OPENS         Files opened for write activity:            2        0/sec
22:54:39.145 DIR_EXISTS          Directory may not exist (yet):              3        0/sec
22:54:39.145 FILE_CLOSES         Close requests:                             2        0/sec
22:54:39.145
22:54:39.260
22:54:39.260 Message from slave localhost-0:
22:54:39.260 New messages found on /var/adm/messages. Do they belong to you?
22:54:39.260 /var/log/messages: Nov 13 22:54:36 jwalit gnome-system-monitor.desktop[6844]: glibtop(c=6844): [WARNING] statvfs '/run/user/1000/doc' failed: Operation not permitted
22:54:39.260 /var/log/messages: Nov 13 22:54:36 jwalit firefox-esr.desktop[2134]: Fontconfig warning: Directory/file mtime in the future. New fonts may not be detected.
22:54:39.260
22:54:40.001 Starting RD=rd1; elapsed=30; fwdrate=max. For loops: None
22:54:40.192
22:54:40.192 Message from slave localhost-0:
22:54:40.192 Anchor: /mnt
22:54:40.192 Vdbench is trying to create a new file, but all files already exist,
22:54:40.192 and no threads are currently active deleting files
22:54:40.192 FwgThread.canWeGetMoreFiles(): Shutting down threads for operation=create
22:54:40.192

Nov 13, 2022 ..Interval.. .ReqstdOps... ...cpu%...  read ....read..... ....write.... ..mb/sec... mb/sec .xfer.. ...mkdir.... ...rmdir.... ...create... ....open.... ...close.... ...delete...
                              rate   resp total  sys   pct   rate   resp   rate   resp  read write  total    size  rate   resp  rate   resp  rate   resp  rate   resp  rate   resp  rate   resp
22:54:41.010           1    0.0  0.000   4.4 0.82   0.0    0.0  0.000    0.0  0.000  0.00  0.00   0.00       0   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000   0.0  0.000
22:54:41.014     avg_2-1 NaN  0.000 NaN NaN   0.0 NaN  0.000 NaN  0.000 NaN NaN NaN       0 NaN  0.000 NaN  0.000 NaN  0.000 NaN  0.000 NaN  0.000 NaN  0.000
22:54:41.014     std_2-1
22:54:41.014     max_2-1
22:54:41.117 Miscellaneous statistics: All counters are zero
22:54:41.296 Vdbench execution completed successfully. Output directory: /home/jwalit/Downloads/Assignment-4/vdbench/output
```

Before Command



```
┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ df
df: /run/user/1000/doc: Operation not permitted
Filesystem     1K-blocks     Used Available Use% Mounted on
udev             3897552        0   3897552   0% /dev
tmpfs             786716     2004    784712   1% /run
/dev/nvme0n1p9  77034228 15154048  57921296  21% /
tmpfs            3933580        0   3933580   0% /dev/shm
tmpfs               5120        0      5120   0% /run/lock
/dev/nvme0n1p7    942764    84196    793340  10% /boot
/dev/nvme0n1p1     98304    62635     35669  64% /boot/efi
tmpfs             786716      168    786548   1% /run/user/1000
/dev/sda1       30005248      288  30004960   1% /media/jwalit/soham
gr14            29644116   464728  29179388   2% /gr14
```

After Command



```
┌──(jwalit㉿jwalit)-[~/Downloads/Assignment-4/vdbench]
└─$ df
df: /run/user/1000/doc: Operation not permitted
Filesystem     1K-blocks     Used Available Use% Mounted on
udev             3897552        0   3897552   0% /dev
tmpfs             786716     2008    784708   1% /run
/dev/nvme0n1p9  77034228 16795704  56279640  23% /
tmpfs            3933580        0   3933580   0% /dev/shm
tmpfs               5120        0      5120   0% /run/lock
/dev/nvme0n1p7    942764    84196    793340  10% /boot
/dev/nvme0n1p1     98304    62635     35669  64% /boot/efi
tmpfs             786716      164    786552   1% /run/user/1000
/dev/sda1       30005248      288  30004960   1% /media/jwalit/soham
gr14            29409888  2099352  27310536   8% /gr14
```

1. File creation finished in **4** seconds.
2. The average write rate is **500MB**/s

Disadvantages of Duplication:

- **Performance:** In the first workload, **EXT4** setup the file system in just **1** seconds compared to **4** seconds for the **ZFS** system. **ZFS** had a write speed of **133.33** MB/s on average, while **EXT4** had a write speed of **450.75 MB/s** on average. **EXT4** performed substantially better in the second task as well, as was already mentioned in the section titled "**Large File Optimization**." **ZFS's** deduplication overhead and **EXT4's** huge file optimization are both contributing factors in this. This demonstrates how deduplication's overhead hurts a file system's performance.

- **CPU Utilisation:** There is more cpu utilization in case of deduplication. Since **EXT4** doesn’t support deduplication, value of average cpu utilization in case of **ZFS** is greater than cpu utilization of **EXT4**, for both workloads.

Disadvantages of Optimising Large File Creation:

- **Greater metadata overhead for small files:** The files only needed **450 MB** when **workload 1** was executing. However, after running **workload1**, additional used space was **461 MB** (**11 MB** of overhead). However, the overhead in **ZFS** was really minimal. For little files, keeping **the extent trees takes up a significant amount more space than the actual data.**

- **No possible recovery from corruption: Ext4** uses extents, delayed and contiguous allocation, and extends to optimise the construction of huge files. Since there is relatively little metadata stored for huge files that are kept in numerous contiguous blocks, it is difficult for any data correction procedures to exist.