Darlyn Close
CSCI340-01
Professor Dixon

Question:

What file systems perform best for small and large files on Linux?

I will be deciding which performs best by comparing the speed at which these file systems read and write files in MB/s.

Setup

I tested this by reading and writing files to a USB drives. I used GParted in order to change the file system of the USB drive.

The different filesystems I tested were FAT32, ext2, ext4, NTFS, and xfs.

For testing purposes, I will define a small file as 10MB/10MiB, and a large file as 105MB/100MiB.

I was running the tests in my Ubuntu 20.0.4 64-bit virtual machine. This virtual machine was running on my 64-bit Windows 10 computer that has an 8-core, 3GHz AMD Ryzen 9 4000 Series Processor, 16GB DDR4 memory, and a PCIe 3.0 SSD. I also had some (5-10) Brave browser tabs open while running the tests, this might have slowed down the speed at which the files were being read and written to.

How I ran tests:

I used the dd command to write random data into a file in the location where the USB drive was mounted. This command was looped multiple times and the output was stored in a file. (.py file included in tar file)

I found the mean and standard deviations, margin of error, and 95% confidence interval using R code. (.Rmd file included in tar file)

I edited the files to only include the time it took to write to the file and the speed (Mb/s). I then edited the files to be CSV files that could be manipulated using R. (CSV files included in tar file)

Editing Process for CSV files:

After reading from stderr from the dd command into a new text file, I used the commands:

```
awk 'NR%3 == 0' < {name_of_file}.txt > final{name_of_file}small.txt
awk -F " " '{print $8 $9 $10 $11}' final{name_of_file}small.txt > fin_texts/{name_of_file}.txt
```

These commands copy only the times and speeds into new text files in a new directory named fin_texts.

Darlyn Close
CSCI340-01
Professor Dixon

I then saved these text files as CSV files and opened them in excel. In excel, I created a new column to number each row and created a new row to give titles to each column. In order to make the data read as numbers rather than strings by R, I also removed the units from the data in excel.

Test Results

Sample Sizes:

All large file tests were run with a sample size of 250 tests. All small file tests were run with a sample size of 500 tests.

Critical Value:

I am using a 95% confidence interval, so my critical value will be 1.96

Margin of error formula solving for E:

$n = (z_c\sigma/E)^2$

$E = z_c\sigma \ / \sqrt{(n)}$

Statistics Chart for Speed (MB/s)

	Standard Deviation	Mean	Margin of Error	95% Confidence Interval
Ext2(large)	1.118619	56.336	0.1386655	56.19733 -56.47467
Ext2(small)	1.950042	56.5224	0.1709287	56.35147 - 56.69333
Ext4(large)	1.161574	5.9484	0.1439902	5.80441 - 6.09239
Ext4(small)	3.960607	7.675	0.3471625	7.327837 - 8.022163
Fat32(large)	1.27016	55.6568	0.1574507	55.49935 - 55.81425
Fat32(small)	2.192276	57.684	0.1921615	57.49184 - 57.87616
NTFS(large)	14.38569	29.89	1.783269	28.10673 - 31.67327
NTFS(small)	6.297324	11.0018	0.5519848	10.44982 - 11.55378
XFS(large)	15.45027	12.5504	1.915235	10.63517 - 14.46563

XFS(small)	5.003218	8.887518	0.4385514	8.448967 - 9.326069

How Accurate are our Results?

It is possible my sample sizes might have been too small to accurately represent the data, especially the large files which I used 250 sample tests instead of 500.

I calculated the 95% confidence intervals, which means that there is still a 5% chance that the true mean of the data lies outside of my confidence intervals.

I also need to keep in mind that my data is based on my definitions of small and large files. I defined a large file as 105MB and a small file as 10MB. Using smaller or larger file sizes could yield different results.

Variation in test code:

The tests for small files had a larger sample size than the tests for larger files. This was due to time constraints.

Conclusion

Large Files:

Ext4 performed the best for large files, with a 95% confidence interval of 5.8044 - 6.0924 MB/s. The next two best filesystems for large files are XFS followed by NTFS. The worst filesystems for large files are ext2 and fat32.

Small Files:

Ext4 performed the best for small files, with a 95% confidence interval of 7.328 – 8.022 MB/s. The next two best filesystems for small files are XFS followed by NTFS. The worst filesystem for small files was fat32, but ext2 performed similarly to fat32 for small files.

Comparing Results of Small and Large Files:

The tests for small and large files yielded similar results. Ext4 performed the best for both small and large files, followed by XFS and NTFS. FAT32 and ext2 performed about the same as each other for both small and large files.

What I learned:

I learned that ext4 appears to be the fastest filesystem for files of sizes around 10MB and 105MB. FAT32 and ext2 appear to be the worst filesystems of the filesystems that were tested.

Darlyn Close
CSCI340-01
Professor Dixon

I also learned how to mount a USB drive, how to change filesystems, and how to use the dd command.