Propose and present a design to improve the "read ()" & "write ()" system call performance in Minix

1. Buffer

We know about the read () and write () system call syntax, ex. read (fd, buff, sizeof(buff)) and for write(fd, buff, sizeof(buff)). First thing to improve the performance of the read and write operation is to reduce the buffer size of the file. when the file is reducing the size then the system call method performs very sufficiently.

Because when we want to improve the performance of the read and write system calls operation first, we should reduce the size of the buffer. Because this will happen less context switching between user space to kernel space. Because the system will take time to switch between two spaces, user space to kernel space.

Second thing is that we have to reduce the number of system calls. If we increment more than one time call read () and write () system calls. then it will automatically reduce the time to perform read and write system calls. Because context switching happens more and systems have to switch between user space to kernel space multiple times. which is not good for our system. For example, we actually cannot perform read () or write () system calls in for loop.

References: -

https://www.javatpoint.com/buffering-in-operating-system#:~:text=There%20are%20three%20main%20types%20of%20buffering%20in,another%20buffer%20simultaneously.%20...%203%203.%20Circular%20Buffer

https://www.geeksforgeeks.org/i-o-buffering-and-its-various-techniques/

Class Notes and Lecture.

2. Cache:

Cache is used to improve the performance of accessing and modifying a file

It is based on Principle of locality that once a data is accessed recently; it is likely to be accessed again in the near future.

Increasing the cache size will result in significant increase of read and write system call performance and vice-versa decreasing the size of cache will result in decrease of read and write system call performance.

References: -

https://en.wikipedia.org/wiki/Cache_(computing)

https://dl.acm.org/doi/abs/10.1145/48012.48037

Class Notes and Lecture.

3. Using of System process instead of terminal process(threads).

A process is a running program in an operating system. It is the fundamental unit of resource allocation in an operating system. A process has a start and an end. It must execute and progress logically. A process is managed and scheduled by the operating system.

Terminal process(threads) will need to switch each time an event is handled i.e., it requires context switching for every event. This will decrease the performance.

System process can handle several events before requires context switching. Less number of contexts switching means better performance of read and write system call but programmability of system process is complex.

References: -

https://guidervision.com/process-in-operating-system-process-states/#:~:text=A%20process%20in%20operating%20system%20is%20a%20running,The%20operating%20system%20manages%20and%20schedules%20a%20process.

https://stackoverflow.com/questions/617787/why-should-i-use-a-thread-vs-using-a-process

https://www.backblaze.com/blog/whats-the-diff-programs-processes-and-threads/

4. Increasing the block of size in pipe:

Capacity of buffer varies from system to system. If a large write has to made then it takes a vast amount of time.

Increasing the block size in pipe increases the performance of write system call significantly.

The downside is it occupies too much kernel much.

References: -

(Credit:- StackOverflow)

https://stackoverflow.com/questions/36986901/increasing-the-size-of-a-pipe-output-in-c https://docs.oracle.com/en/database/oracle/oracle-database/18/cwaix/increasing-system-block-size-allocation.html