SPLICEZEROCOPYSYSTEM OPTIMIZATION USING DUAL PIPE SCHEMES

Andy Victor[1], Maman Abdurahman[2], Fazmah Arif Yulianto[3]

[1]Magister Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom

Abstrak

Proses transfer data merupakan salah satu fungsi pemrosesan data yang dilakukan oleh sistem operasi.Secara tradisional proses ini cukup banyak mengkonsumsi penggunaan sumber dayaMemory dan CPU pada sistem komputer. Hal ini disebabkan karena adanya proses berulang yang tidak diperlukan yaitu CPU Copy yang dilakukan dua kali oleh CPU. Prinsip utama dari zerocopy adalah mencegah sepenuhnya atau setidaknya meminimalisiroperasi penyalinan data yang dilakukan oleh CPU ketika sedang melakukan prosesI/O datadidalam kernel seperti antrian jaringan dan diskdrive penyimpanan. Dari berbagaipendekatan teknik zerocopy yang ada, maka penulis fokus terhadap Splice System Calls.Splice system callmencegah proses penyalinan data dari kernel space ke user space dan juga sebaliknya. Membaca dari offset tertentu dari file input dan menuliskannya ke buffer pipe pada kernel space.

Pengujian dilakukan terhadap file tunggal dengan 10 jenis file dan ukuran yang berbeda. Proses transfer data dengan jenis file, nama file dan ukuran file yang sama akan dilakukan dan diuji dengan 2 metode yang berbeda yaitu dengan metode splice dan splice dual pipe. Konsumsi waktu dari setiap proses dengan metode yang berbeda akan dicatat dan kemudian dibandingkan.Proses pengujian untuk setiap file dan setiap metode dilakukan sebanyak 10 kali, sehingga proses transfer satu buah file akan menghasilkan 20 data pengujian dengan rincian 10 kali untuk splice dan 10 kali untuk splice dual pipe. Hasil eksperimen menunjukkan percepatan konsumsi waktu yang lebih baik dibandingkan dengan metode yang sudah ada. Data rate rata-rata yang dihasilkan oleh Splice Dual Pipes adalah 418.56 byte per milidetik, ini lebih cepat daripada Splice yang hanya menghasilkan 65.60 byte per milidetik

Kata Kunci : Transfer Data, Zero Copy, Splice System Calls, Splice Dual Pipe System Calls

Abstract

The process of transferring data is one function of the data processing performed by the operating system. Traditionally, due to the unnecessary iterative process or CPU copy performed twice by the CPU, this process consumes relatively a lot of memory and CPU resources on a computer system.

The main principle of zerocopy is to prevent or to minimize the operation of copying data performed by the CPU during I/O data processing in the kernel such as during network queuing and diskdrive storing. Nowadays, various approaches to zerocopy are carried out andthis thesis focuses on the Splice System Calls. This Splice System Call prevents copying all data from user space to kernel space and vice versa. It reads data from the specified offset from the input file and writes them to the pipe buffer in the kernel space.

The tests were conducted on a single file with 10 types of files with different sizes. The process of transferring data of the same file, name and size was tested usingtwo different methods, namely splice system calls and splice dual pipe system calls. The durations of each process with different methods were recorded and then compared. Each file using each method was tested 10 times; thus, each file had 20 pieces of files, 10pieces using splice and 10pieces using splice dual pipe. The experiments showedthat using Splice Dual Pipes System Calls the transferring process was quicker than using the Splice System Calls methods.The average data rate generated by splice dual pipes was418.56 bytes per millisecond. This was faster than the Splice which wasonly 65.60 bytes per millisecond.

Keywords : Transferring data, Zero Copy, Splice System Calls, Splice Dual Pipe System Calls

CHAPTER 1
THE PROBLEM

1.1. Rationale

Nowadays, transferring data is daily routine activities of computer users. They transfer data from or to various media, CD/DVD, Hard drive, Flashdisk, or from one computer to others. In general, the process of transferring is quite simple, selecting the data to be transferred from specified source, then specifying the destination location of transferring data, and finally data are transferred. During the process of transferring, the operating system reacts and coordinates with all the resources associated with this process. Figure 1.1 showed the typical file transfer operation as performed by some kind of computer [3]. Data from hard drive copied by DMA to kernel buffer, then data copied to user buffer by CPU copy. These two processes is called as a read process. The next process of transferring file is copying data to socket buffer by CPU copy, and then data copied to Network interface by DMA copy. These last two processes is known as write process.

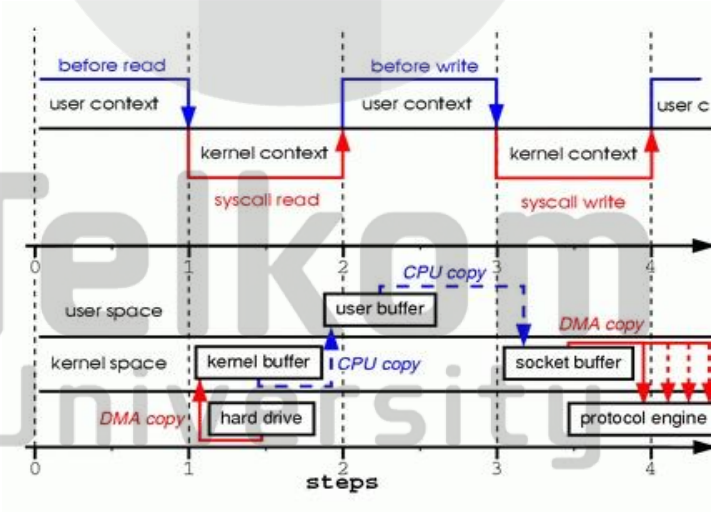

Figure 1.1 Traditional Data Transfer [3]

The whole procedure consumes a lot of CPU cycles and memory bandwith, for data must be copied between application memory space and kernel memory space[1]. The kernel processes messages, causing many times of data copy and a lot of content transforming and at the end resulting in the high delay between network point to

points[2]. In general, the whole unnecessary process of data copying cause the computer have a poor performance. Zero copy is a common name for various techniques and design improvements aimed at reduction of unnecessary memory accesses, ussually involving avoidance of data copying [3]. Many researchers have developed zerocopy methods to optimizing the traditional data transfer to increase good I/O performance.

One of them is the one developed splice zerocopy, Larry McVoy[4]. These zerocopy techniques aims to avoid and optimizing kernel data copying. Use an integer file descriptor to replace data copying from kernel buffer to user buffer and from user buffer to socket buffer. In other word, these techniques eliminate CPU copy from kernel buffer to user buffer and from buffer to socket buffer.

Another study which focus on area avoidance and optimizing kernel data copying is mmap (memory mapping), sendfile and sendfile with DMA scatter/gather copy. Table 1.1 showed advantages and disadvantages of each techniques in optimizing kernel data copying area.

Table. 1.1 Advantages and Disadvantages Zerocopy Techniques [1]

Method / Techniques	CPU Copy	DMA Copy	Context Switch	Advantages And Disadvantages
Memory Mapping	1	2	4	Virtual memory operation is costly. COW (copy on write) needs to be implemented and this is costly
Sendfile	1	2	2	Sendfile hard to implement due to asynchronous network transfers.
Sendfile with Scatter/Gather Copy	0	2	2	Need hardware which supports DMA scatter/gather copy. Sendfile hard to implement due to asynchronous network transfers.
Splice	0	2	2	When using splice system call, there must has two file descriptors which opened to both the input and the output devices.

Table 1.1 shows the comparison of each techniques that focus on avoidance and optimizing kenel data copying in order of CPU Copy, DMA Copy and Context Switch elimination for good improvement I/O performance. The result show that the splice techniques performs better than the else techniques.

1.2. Theoritical Framework

These study proposes a new approach to modify the pipe buffer that used on splice techniques. The main objective is to reduce time consumption that needed by adding another pipes buffer to store file that will be transmitted. In other word, new splice method that proposed uses two pipes buffer. File to be transmitted is divided into two parts, partially into the first pipe and some into the second pipe. Read and write process on first pipe will be represented by file descriptor FDIn1 and FDOut1, whereas the second pipe will be represented by FDIn2 and FDOut2. The proposed method will work in parallel using two pipes, this is done to speed up the time consumption required to perform file transfer.

1.3. Conceptual Framework/Paradigm

Some of the main variables used in this research are the use of CPU resources, memory resources, and the operating system used in the data transfer process. Another variables that is also important is the network technology such as the use of ethernet network. But in this case the technology of the computer network is not a major variable, in this case the computer network is a variable that is only a trigger factor related to he processes within the operating system. The whole process of interaction between the main variables in this study is carried out and occurs in operating system. Since the harddisk and network card is equal as a I/O device[4], network card will be replaced by harddisk device. Measurement of file size using bytes, and time consumption using milliseconds.

1.4. Statement of the problem

Traditionally, whole procedure of transfering data from one to another computer consumes a lot of CPU cycles and memory. A lot of data duplication is not really necessary to hold things up in Operating System. The impact of this process is time and resource consuming operations, like memory copy operations, limiting the number of concurrent tasks that can be executed by the computer. This can make memory and CPU

resources unavailable for other tasks, and these processes make the computer have poor performance. The longer time is needed to perform the data transfer process, the more it will have an impact on poor performance. Therefore, the faster process of transferring will greatly affect good performance improvement.

1.5. Hypothesis

With adding more pipes in splice mechanism, Time consumption that expected from proposed method is shown on figure 1.2, Splice Dual Pipes Zerocopy will be faster than prior method.

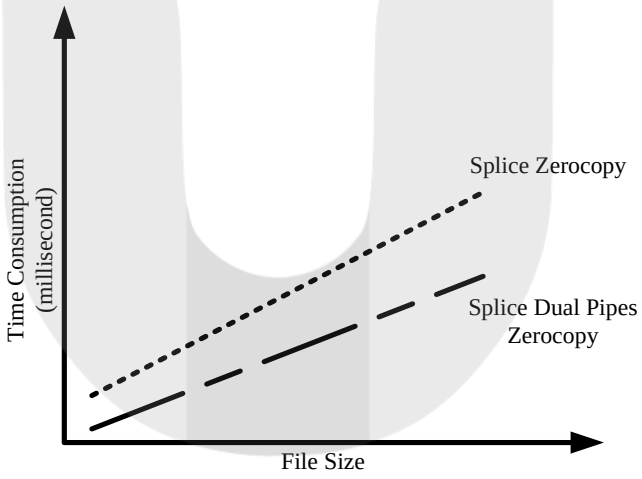

Figure 1.2 Time Consumption Hypothesis

1.6. Assumption

The following are some of the assumptions of this study :

1. There is no difference treatment of various types of data transfer such as video, audio and text.
2. This study focuses on improving the time consuming, since other research has carried out studies to decrease CPU and memory consumption.
3. Harddisk and Network card is equal as a I/O Device

1.7. Scope and Delimitation

This study focuses on these following areas :

1. The research area is on the process transfer data that involved in operating system.

2. The method used is to look for the minimum time consumption.
3. The method focuses on the kernel and not on the hardware.

1.8. Importance of the study

Optimizing the process of transferring data on the operating system in a way to reduce the number of copies between user buffer and kernel buffer. This process will help the CPU and Memory resource work efficiently and make the computer stable and able to work well. In addition, there will be more task that can be performed in parallel by the CPU at the same time.

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

5.1. Conclusions

1. Based on the proposed method with adding dual pipes to the splice mechanism showed that the proposed method improved time consumption faster and efficient than prior method, it was possible because the process of splice do by paralel scheme during the file transfer process.
2. The average data rate generated by splice dual pipes is 418.56 bytes per millisecond, this was faster than the splice which only produced 65.60 bytes per millisecond.

5.2. Recommendations

It is acknowledged that this study still has some limitations and needs further study such as :

1. The file to be tested is multiplied in future work.
2. Other suggestion is experiment should be tested on variety of condition such a busy process with lot of application running on operating system.

REFERENCE

1. Jia Song and Jim Alves-Foss : Performance Review of Zero Copy Techniques. International Journal of Computer Science and Security (IJCSS), Volume (6) : Issue (4) : 2012.
2. Liu Tianhua, Zhu Hongfeng, Liu Jie and Zhou Chuansheng: Design and Implementation of Zero-Copy for Linux. International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM). 2011.
3. E. Brose. "ZeroCopy: Techniques, Benefits and Pitfalls," http://kbs.cs.tuberlin.de/teaching/ws2005/htos/papers/streams-zcpy.pdf, 2005.
4. Larry McVoy. "The Splice I/O Model. Revision 1.4 of 06/24/98. im@bitmover.com.
5. www.linuxjournal.com Network Buffers and memory management.
6. http://www.linuxfoundation.org/collaborate/workgroups/networking/skbuff
7. http://www.tldp.org/LDP/tlk/ds/ds.html Linux Data Structure
8. D. Stancevic. "Zero Copy I: User-Mode Perspective," Linux Journal, vol. 2003 no 105, Jan.2003, pp. 3.
9. J.Corbet, A.Rubini, and G.Kroah-Hartman. "Memory Mapping and DMA" in Linux Device Drivers, third edition, O'REILLY, Feb, 2005, pp.450.
10. Linux Programmer's Manual, http://www.unix.com/man-page/Linux/2/sendfile/ Feb. 2010.
11. Linux Programmer's Manual, http://www.unix.com/man-page/Linux/2/splice/ Sep. 2009.
12. P. Halvorsen, E. Jorde, K. Skevik, V. Goebel, T. Plagemann, "Performance Tradeoffs for Static Allocation of Zero-Copy Buffers," in Proc. Euromicro Conference, 2002, pp. 138-143.
13. M. Chiang and Y. Li, "LyraNET: A Zero-Copy TCP/IP Protocol Stack for Embedded Operating Systems," in Proc. IEEE International Conference on Embedded and Real-TimeComputing Systems and Applications, Aug. 2005, pp. 123-128.
14. K. Fall and J. Pasquale, "Exploiting In-Kernel Data Paths to Improve I/O Throughput and CPU Availability," in Proc. Winter 1993 USENIX Conference, 1993, pp. 327-333.
15. Daniel P. Bovet and Marco Cesati, "Understanding The Linux Kernel" third edition, 2006 O'Reilly Media, Inc.
16. http://man7.org/linux/man-pages/man2/splice.2.html
17. http://man7.org/linux/man-pages/man2/tee.2.html