Consolidating with Media Streaming Server and Network Storage Card

YongJu Lee[1], SongWoo Sok[1], HagYoung Kim[1], MyungJoon Kim[1], and CheolHoon Lee[2]

[1] Dept. of Internet Server Group, Digital Home Research Division, Electronics and Telecommunications Research Institute, 161 Gajeong-Dong, Yuseong-Gu, Daejeon, Korea
{yongju,swsok,h0kim,joonkim}@etri.re.kr
[2] System Software Lab., Dept. of Computer Engineering, Chungnam National University, Daejeon 305-764, Korea
chlee@ce.cnu.ac.kr

Abstract. Recently there have been a large number of multimedia demands of IP-based applications in tremendous Internet world. However, Internet-based multimedia services typically do not offer any quality of service because of its own characteristics such as network bandwidth. In case of Korea, assumption that dense residential communities sharing their livings and information have their infrastructures like high speed communication network is able to achieve quality of service at a close range. This trend creates new market by creating new type of service in Korean-style Internet server. In this paper, we design and implement the high performance media streaming server focusing on local communities which obtain high speed network bandwidth, using streaming-accelerated network storage card which is enable to offer "zero-copy" interfaces and raw-disk I/O.

1 Introduction

Multimedia content on websites can be accessed in one of two ways: either by downloading the entire content, or by using a streaming server. The streaming server enables a client to start viewing the content much faster, and is the preferred method for long streaming or for watching real-time events on a network. Anyone who has played multimedia streams over the Internet has encountered its quality's degradation severely. However, multimedia services in Korean-style residential district with dense complexity are more adaptable to deploy high-density playback because of abundant Internet infrastructure and closer network nodes. In order to provide a high quality service for multimedia demands, powerful network bandwidth is needed. In this purpose, network storage add-on card (NSCard) for streaming-acceleration is made recently. This NS card can transfer stream data from disks to end users directly in a zero-copy technique via the network without any intervention of host processor in the server[1, 2]. The streaming server with this powerful NS card allows support for varying

T. Enokido, L. Barolli, and M. Takizawa (Eds.): NBiS 2007, LNCS 4658, pp. 495–504, 2007.

quality of service streaming and for providing more plenty of concurrent users. Therefore, we design and implement the high performance streaming server on network-storage card. The main contribution of this paper is the deployment of an NS card and the development of high performance streaming server using it. Through alleviating CPU's burden and gaining zero-copy based I/O bandwidth, we have proved to reach the limits of a system significantly if it is well-configured and apparently used. The remainder of this paper is structured as follows. In section 2, we analyze the present problems and introduce an NS card's practical strength and weakness. In section 3, we discuss our high performance streaming server's architecture and implementation in detail. And then, we show the efficacy of our system through extensive experiment in Section 4. Finally, we conclude the paper in Section 5.

2 System Requirements

In this section, we present preliminary requirements for deploying an NS card and obtaining better performance in Linux. An NS card has its own software device drivers: Stream Disk Array(SDA), PCI Memory(PMEM), TCP Offloading Engine(TOE). Streaming data copied to the PMEM is directly transmitted to network without additional memory copy operations(so-called zero-copy)[3]. Zero-copy technique using above drivers gives better performance if a system is well-configured and used appropriately. SDA is a special purpose disk array optimized for large sequential disk accesses in a way of pipelined I/O.(e.g. a SDA device driver support 1M/512K/256K/128K block sizes). These large block sizes make easier way to alleviate numbers of read() operations, but streaming server designer must consider units of buffering/caching and also be aware of the fact that a fixed block size is not changed dynamically because of applying to modify a unit size in a system widely, is not used in common with separate files because of interval's differentiation among files. Peripheral memory is equipped with DRAM memory modules and is dedicated to temporary buffer on disk-to-network data path without user's context switching. In general, PMEM is not adaptable to calculating like a main memory.(e.g. PMEM's memcpy/memset call is approximately four or five times slower than main memory's). This limitation also is considered for designing non-arithmetic streaming operations. Lastly, TOE provides protocol-based offload network interfaces without modification in the light of an application developer. In general, a streaming server makes use of threads for each connection to clients because of the independency among connections. Demands of large scale performance lead to the difficulty of this approach because of many threads. However, the Native POSIX Thread Library(NPTL) is encouraged to us for using independent thread handling without system degradation[4]. Together with this effort, we choose an N:M model to improve CPU's time-slice ticks and to avoid silly waiting. In conventional operation, we allocate a main memory and opens a suitable file and a socket. Thereafter, we read a file and send its data until end-of-file. In our fast-path I/O operation, we prepare a PMEM device and allocate a SDA_SIZE's memory