Distributed Data Oriented Centralized Storage Management in a Scalable Streaming Media System

YUHUI DENG\textsuperscript{1}, FRANK WANG\textsuperscript{1}, DAN FENG\textsuperscript{2}, NA HELIAN\textsuperscript{3}, DAVE MILWARD\textsuperscript{4}, STEVE THOMPSON\textsuperscript{4}

\textsuperscript{1}Center for Grid Computing, Cambridge-Cranfield High Performance Computing Facilities, Cranfield University Campus, Bedfordshire MK430AL, United Kingdom
\textsuperscript{2}Key Laboratory of Data Storage System, Ministry of Education, School of Computer, Huazhong University of Science and Technology, Wuhan 430074, P. R. China
\textsuperscript{3}London Metropolitan University, United Kingdom
\textsuperscript{4}Xyratex, United Kingdom

Abstract: Streaming media is pervasive on the Internet now and continues to grow rapidly. The dramatic advances in both the capacity and cost-effectiveness of storage systems have made disk-based storage an increasingly attractive alternative for streaming media system. A large scale streaming media system may demand thousands of disks to satisfy both the bandwidth and storage capacity requirements imposed by thousands of concurrent clients. Distributed data storage is normally adopted to improve the performance and scalability. Unfortunately, an effective storage management mechanism for a large scale distributed streaming media system presents several research challenges. This paper designs and implements a highly reliable and available Distributed Data Oriented Storage Management (DDOSM) mechanism in a scalable streaming media system. All heterogeneous storage resources across the system are aggregated as a single logical view which strikes a good balance between a distributed data storage and centralized storage management. The centralized management of all physical and logical storage resources dramatically simplifies the storage administration as the size and complexity of the streaming media system grows. Fault tolerance and dynamic load balance at the disk level and system level are achieved to provide high reliability and availability due to the DDOSM.

Key-Words: Streaming Media, Storage Management, System Architecture, Fault Tolerance, Load Balance

1 Introduction
Since the emergence of streaming media, due to the rapid advances in computer systems, high-speed networks, packet switching, data compression etc, the streaming media has experienced a dramatic growth\textsuperscript{[1]}.

The dramatic advances in both the capacity and cost-effectiveness of storage systems have made disk-based storage an increasingly attractive alternative for streaming media system \textsuperscript{[2]}. Due to the intensive nature of the media data, the applications require a high and stable bandwidth and throughput to provide Quality of Service (QoS). The Redundant Array of Independent Disks (RAID) \textsuperscript{[3]} is at present the most efficient way to provide such a performance requirement, to a certain degree. However, the design of large scale and scalable multimedia servers imposes several research challenges that demand more scalable alternatives than the RAID solution.

There have been some research efforts invested in designing scalable video servers over the past few years\textsuperscript{[4,5,6]}. In our earlier research, we designed and implemented an innovative High Performance Streaming Media System (HPSMS) based on the logical separation in the streaming media transport protocol \textsuperscript{[7]}. The architecture avoids store-and-forward data copying between the streaming media server and storage devices to eliminate the server bottleneck. The salient feature is that the system bandwidth dynamically increases with the expansion of the storage system capacity.

Data management has many aspects. While performance has long been the focus of storage systems research, manageability has become the dominant criterion in evaluating storage solutions, as the cost of storage management outweighs the cost of the storage devices themselves by a factor of three to eight \textsuperscript{[8,9]}.

A large streaming media system may demand thousands of disks to satisfy both the bandwidth and storage capacity requirements imposed by
thousands of concurrent clients [10]. Although a single disk is fairly reliable, with a large number of disks, the aggregate rate of disk failure can be too high [11]. Undoubtedly, an effective storage management mechanism for a large scale streaming media system presents several research challenges.

In this paper, we design and implement a highly reliable and available Distributed Data Oriented Storage Management (DDOSM) mechanism in HPSMS. All heterogeneous storage resources across the HPSMS are organized as a single logical view which strikes a good balance between a centralized storage management and distributed data storage. Centralized management of all the physical and logical storage resources dramatically simplify storage administration as the size and complexity of the streaming media system grows. Fault tolerance and dynamic load balance at disk level and system level are achieved to provide high reliability and availability due to the DDOSM.

The remainder of the paper is organized as follows. The system architecture of HPSMS is briefly introduced in Section 2. Section 3 describes the prototype implementation of the DDOSM. The implemented prototype is evaluated in section 4. Section 5 concludes the paper with remarks on main contributions of the paper and future research indication.

2 The HPSMS Architecture

The DDOSM is based on the HPSMS that is fully described and evaluated in [7]. To make this paper self-contained, we give a brief overview of the HPSMS.

![HPSMS Architecture Diagram](image)

Fig. 1 HPSMS Architecture

The key feature of HPSMS is that the storage system (Net-RAID) has two interfaces, a Small Computer System Interface (SCSI) channel that links Net-RAID to the streaming media server, and a network channel that directly connects to the switch for the data transmission. All Net-RAIDs are centrally controlled by the streaming media server through the SCSI channel for the convenience of management, while all network interfaces of Net-RAIDs are allowed parallel data transmission (see Fig. 1). Real-time Transport Protocol (RTP) consists of two protocols, RTP for real-time transmission of data packets and Real-time Transport Control Protocol (RTCP) for monitoring QoS and for conveying participants' identities in a session. Because RTP uses different logical channels to transport control and data stream, we move the logical channel of RTP from streaming media server to the physical network channel of Net-RAID.

Although the NICs can be used for communication between the streaming media server and the Net-RAIDs, the storage management can still be problematic, due to the distributed nature of the data of Net-RAID group. By keeping the SCSI channel of Net-RAID connected to the streaming media server to exert central control, HPSMS strikes a good balance between a centralized storage management and distributed data storage.

Storage system capacity must keep pace with the continuous growth of streaming media data. HPSMS achieves this scalability by expanding the system storage capacity incrementally with additional Net-RAIDs along with associated network interfaces that expand the data transmission rate proportionally.

3 The Prototype Implementation

We detail the design and implementation of Net-RAID in [7]. In this section, We will illustrate how to virtualize the heterogenous and distributed storage resources residing on multiple Net-RAIDs as a single logical view to execute centralized management.

3. 1 The System Buffer Cache Based DDOSM

The implementation of single system image for disk I/O can be categorized into user level, file system level and device driver level [12]. User level designs have higher portability and lower implementation cost. However, using system calls to perform I/O may decrease the performance. File system level designs can have full control in data
distribution. But changing the file system does not guarantee strict compatibility with current applications. Device driver level designs minimize the file systems modifications. The disadvantage is that it is difficult to mask the heterogenous storage device and control the distribution pattern of files.

All of the Linux file systems use a common buffer cache to cache data from the underlying devices to help speed up access to the physical devices. The buffer cache makes the Linux file systems independent from the underlying physical storage device and the device drivers that support them. All block devices (Even the relatively complex block devices such as SCSI devices) register themselves with the Linux kernel and present a uniform, block based, and asynchronous interface.

The DDOSM consists of virtual layer, mapping layer and data redirecting layer. Layered design and standardized system interfaces are adopted to provide compatibility and code reusability.

3.2.1 The Virtual Layer
Virtual layer is a main frame of the DDOSM control software. This layer simulates a standard block device driver to register the DDOSM in Linux kernel and provides the same system interfaces to the operating system as a physical hard disk driver. The layer parses the storage capacity of heterogeneous storage nodes and calculates the overall storage capacity of the whole streaming media system. It provides virtual device parameters such as Cylinder/Head/Sector (C/H/S) and logical block addressing (LBA) to the operating system.

3.2.2 The Mapping Layer
The basic function of the mapping layer is to translate virtual addresses of the single logical view to the physical storage nodes address. Different mapping algorithms achieve different functions. By leveraging the high-speed communication afforded by the SCSI and FC interconnect, large files can be stored in a scalable fashion by striping the data across multiple storage nodes.

3.2.3 The Data Redirecting Layer
The DDOSM enables administrators to consolidate heterogeneous storage resources across multiple Net-RAIDs onto one single logical view with performance guarantees, by intercepting every I/O request from the file system and redirecting them to back-end physical storage nodes. The DDOSM can improve system performance to a certain degree because multiple storage nodes can operate in parallel.

In the Linux operating system, every block device driver maintains a request queue used by default. The block request function performs some optimization of the request queue such as the consecutive sector combination. We intercept the I/O sub-requests from the mapping layer, and modify the sub-requests to link them on the request queue of the SCSI block device driver. The SCSI stack controls the device driver to perform the real I/O operations. FC block devices provide the same system interfaces as SCSI.

3.2 The Software Components of DDOSM

SCSI is an attractive alternative for storage interface that provides a high performance, highly reliable solution to the increasing demands for speed and flexibility in storage devices. SCSI is the widely used protocol over FC since the FC commands are an encapsulation of SCSI command set.

We design and implement the buffer cache based DDOSM between the file system level and block device driver level (Fig.2). Compared with previous approaches that implement single system image for disk I/O, the method presented in this paper can shield the heterogenous storage resources(SCSI and FC devices), provide a complete single logical view to the file system, control the data distribution pattern in a manageable way, minimize the file system modification, and avoid the context switch overhead of system call.

Fig.2 Implement DDOSM at System Buffer Level
3.2.4 The Fault Tolerance and Load Balance

In the chained-declustering algorithm [13], there are two copies (primary copy and backup copy) of each data block stored on neighboring nodes. Every pair of neighboring nodes has data blocks in common. During the normal mode of operation, read requests are directed to the fragments of the primary copy and write operations result in both copies being updated. In the event of a node failure that results in a fragment of the primary copy being unavailable, the corresponding fragment of the backup copy will be promoted to become the primary fragment and all data access will be directed to it. The chained-declustering algorithm gives clients highly available access to data by automatically bypassing failed nodes. Dynamic load balance eliminates system bottlenecks by ensuring uniform load distribution even in the face of node failures.

The chained-declustering data placement scheme in the DDOSM is depicted in Fig.3. Data A (in the dashed rectangle) is striped across storage nodes A, B, C, D. Due to this arrangement, if node B fails, node A and C will automatically share node B’s read load. Since node D has copies of some data from node A and C, node A and C can offload some of their normal read load on to node D and achieve uniform dynamic load balance.

The Net-RAID employs RAID technology to provide fault tolerance at disk level because it is undesirable for a single disk failure to impact upon the whole streaming media system. The chained-declustering algorithm is adopted to support fault tolerance at system level in DDOSM. Fault tolerance at two levels guarantees highly reliable and available storage management in HPSMS.

4 Prototype Evaluations

We adopt a similar system configuration as HPSMS [7] to validate the DDOSM. The prototype consists of one streaming media server and three attached Net-RAID through FC. Two SCSI disks in the Net-RAID are configured as RAID0. The chained-declustering data placement scheme is implemented in the DDOSM. Table 1 shows the configuration of the prototype. All read and write requests in the following tests are 100% sequential.

<table>
<thead>
<tr>
<th>CPU</th>
<th>Streaming media server</th>
<th>Net-RAID</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Celeron 1.7GHZ</td>
<td>Pentium3 500MHz</td>
</tr>
<tr>
<td>Memory</td>
<td>128M</td>
<td>128M</td>
</tr>
<tr>
<td>NIC</td>
<td>RealTek100M</td>
<td>RealTek100M</td>
</tr>
<tr>
<td>Target FC HBA</td>
<td>Qlogic QLA2100</td>
<td>Qlogic QLA2100</td>
</tr>
<tr>
<td>Initiator FC HBA</td>
<td>NCR 53C875</td>
<td>NCR 53C875</td>
</tr>
<tr>
<td>OS</td>
<td>RH7.1(2.4.2-2)</td>
<td>RT Linux(2.2.12)</td>
</tr>
<tr>
<td>SCSI Disks</td>
<td>ST173404LC</td>
<td>ST173404LC</td>
</tr>
</tbody>
</table>

Fig.4 shows the measure results of the prototype. Lmdd benchmark [14] reports that the Net-RAID has a read bandwidth of about 27MBytes/s and a write bandwidth of 23MBytes/s with 8KB stripe size. With the chained-declustering data placement scheme implemented across three Net-RAIDs in DDOSM, the aggregate read bandwidth gives linear improvement when the
request block size is changed from 8KByte to 64KByte. Due to the limitation of computer hardware components (e.g., HBA, I/O bus and disk drive), the maximal I/O Control Block (IOCB) issued by the FC HBA of the streaming media server is limited up to 64KB. The 64KB request block size saturates the read bandwidth with 49.1MBytes/s. The aggregate write bandwidth is around 30.3MBytes/s and is not affected by request block size. That’s because the DDOSM is based on buffer cache and adopts a write back policy. We achieve nearly 31% improvement in write bandwidth and around 79% improvement in read bandwidth, respectively.

Fig.4 also illustrates the performance of the DDOSM when one of the three nodes fails. The read bandwidth decreases by about 16.3%, whereas write bandwidth decreases by about 3% compared with single Net-RAID. The results indicate that the dynamical workload balance is working effectively due to the chained-declustering data placement scheme after the working Net-RAIDs are reduced from three to two.

5 Conclusions
In this paper, we designed and implemented a highly available DDOSM mechanism that aggregates the heterogeneous storage resources residing in HPSMS as a single logical view. The centralized management dramatically simplifies the storage management of HPSMS. The two level fault tolerance solution guarantees system availability. The DDOSM directly addresses the needs of both administrators and users by increasing media data availability, optimizing storage capacity, and improving system performance, all of which contribute to a significant reduction in the total cost of ownership of a large scale streaming media system. Experimental results validate the effect of the DDOSM.

Possible directions for future work include the data mapping algorithms, hot data migration, and dynamic system reconfiguration etc.

References:


