Quanta Data Storage: A New Storage Paradigm

Prabhanjan C. Gurumohan
Arizona State University
gpj@asu.edu

Sai S. B. Narasimhamurthy
Arizona State University
saib@asu.edu

Joseph Y. Hui
Arizona State University
jhui@asu.edu

Abstract

TCP layer and poor iSCSI implementations have been identified as the main bottlenecks in realizing high iSCSI performance. With the addition of security mechanisms the throughput achieved by the storage system using iSCSI is further reduced. Along with the above mentioned problems, we argue that the excessive processing redundancy introduced by several protocol layers and use of protocols designed for non-storage specific requirements result in poor storage network architectures. In order to overcome these issues, we introduce a new storage paradigm in which data is manipulated, encrypted and stored in fixed block sizes called quanta. Each quanta is manipulated by a single effective cross layer (ECL) that includes security features, iSCSI functionalities, direct data placement techniques and data transport mechanisms. Further, the new architecture emphasizes majority of burden of computation for achieving security on the clients.

Qualitative description of the idea is presented. Performance improvements observed during tests of the idea are presented. Through emulation and analysis we also show that the size of the quanta must be equal to the minimum path MTU for maximum throughput.

1. Introduction

IP has been accepted as a de-facto standard for Internet applications. IP based networks also provide relatively inexpensive and convenient solution [5][13] for transportation of bulk data. Considering these facts, transportation of storage data on the IP network provides a cheap and easy alternative to the SCSI and Fiber Channel based networks. iSCSI [5] is the proposed protocol for storage (block) data transport over IP networks. Most effort has been concentrated on designing the protocol over the existing TCP/IP protocol. Initial versions of the implementation of this protocol are available. Although the idea to develop these new protocols over existing protocols was done to ease the integration of iSCSI, it has resulted in over layering and crowding of protocols stacks.

Stephen et. al [11] have presented the findings of implementing an iSCSI based *target* on a specialized hardware. Their work concentrated on comparing the overall performance of various network configurations using iSCSI protocol. The performance results from their work indicate that the iSCSI protocol can be severely limited by implementation. They suspect this problem due to inefficient handling of underlying network properties and poor iSCSI implementation.

Y.Lu and D.Du [8] have examined different storage protocols (viz, iSCSI, NFS,SMB) and have analyzed their performance. The work shows that iSCSI storage with Gigabit connection could have performance very close to directly attached fiber channel-arbitrated loop storage. From their study they also show that iSCSI based file access performance outperforms the NAS schemes on both Windows and Linux platforms. However, they point out that the advantage of iSCSI reduces as the file size increases. The reasoning for this reduction in performance has not been presented in this work.

Shuang-Yi Tang et. al [10] have also shown that with addition of security features such as IPsec below iSCSI and TCP/IP protocol stack leads to high degradation in throughput. Further the idea of storage security is not well severed by usage of IPsec because it does not encrypt the data that will be stored on the storage device. It only protects from attacks between two communicating points. Hence, using iSCSI over TCP/IP/IPsec leads to high overhead and complex layering structure. Further, use of IPsec leads to encryption and decryption at both sever and client ends.

End system copies and TCP packet reassembly form throughout bottlenecks for many applications including iSCSI [1][2][3]. End system copies can be eliminated by Direct Data Placement (DDP) where the application entities are directly placed into application buffers without system copies [1]. DDP packets have to be modified to suite the existing TCP functions. This is enabled by a Framing protocol for TCP [3]. Additional extensions to the DDP protocol are provided by the RDMA protocol, which operates over the DDP protocol. The RDMA, DDP and the Framing protocol form the iWARP suite. This is the proposed solution for end system copy issues [12]. The iWARP suite is also applicable for iSCSI [2]. The iWARP suite is proposed to operate between iSCSI and TCP [12].

The introduction of iWARP leads to increased overheads and redundancy. Two simple examples of redundant functions at different layers are as follows. First, the CRCs/checksums are computed at the iSCSI layer, MPA or framing layer, and the TCP layer. Second, sequencing information is repeated in both iSCSI and transport layer.

Motivated to overcome these problems, we propose the use of fixed block size data units called *quanta*. These data units are chosen to be of size 512, 1024, 2048 or 4096 bytes. A quanta is encrypted and formatted according to an Effective Cross Layer (ECL) by the client wishing to write data to a server and is stored *as is* on the storage device. The effective cross layer combines the functionalities of encryption, buffer management for direct data placement, iSCSI formatting and transport functions. The key used for encryption of such a quanta is stored on a separate key server. Any valid client can access the encrypted and preformatted data from the server and decrypt it using keys obtained from the key server. The client does the encryption and decryption, checksum calculation, and any other formatting. This lets the target to be implemented with fewer functionalities of ECL and used for only providing an access point to the storage device.

A facility called *fast buffers* (fbufs) was introduced by Peter Druschel and Larry L. Peterson [14]. It was an operating system facility implemented for I/O buffer management and data transfer across protection domain boundaries on shared memory machines. Although, the main goal of this idea was to provide high bandwidth to user level processes, this was achieved by implementing a new facility in the operating system. This is unlike the ECL and quanta approach that achieves the similar goal by reducing the overheads in the protocol stacks. Further, ECL and quanta approach are specifically designed to improve the performance of the storage networks instead for the general networks.

David D. Clark and David L. Tennenhouse [15] provide guidelines for designing of new generation of protocols. Our protocol design efforts are in line with two important guidelines presented in [15]. They are, the data manipulation costs are more compared to transfer control operations and the application data units are the natural pipelining units.

In section 2, we provide the architectural goals of the cross layer design. Section 3 discusses the encryption mechanism details. We present the ECL details in section 4. Section 5 provides the test results for the emulation of ECL using Hyperscsi. Finally we present the conclusion in section 6.

2. Cross layer architectural goals

CLA (Cross Layer Architecture) enables the combining of the features of the SCSI/iSCSI/RDMA /DDP/Framing/TCP/IPsec into a single layer called the ECL. This obviates layering overheads and enables the ease of operation of Storage Area Networks.

CLA has been designed with the following architectural goals:

1. Minimize data handling and processing by dividing data units into fixed size blocks or quanta.

2. Provide security on the wire and on the storage unit.

3. Provide the features of the iSCSI protocol.

4. Provide application level buffering by incorporating the features of the DDP protocol. (Direct Data Placement)

5. Provide the transport layer mechanisms.

3. Encryption Mechanism

In order to include a security mechanism as a part of a single layer, ECL, a new security mechanism is proposed. The new scheme for storage security mainly deals with reducing the high overheads of authentication and encryption. This is done by avoiding encryption and decryption of data on both client and server sides. The new scheme does both encryption and decryption on the client side. The new scheme includes several techniques, which improves the efficiency of encryption and decryption:

1. The data is stored in the encrypted form on the server:

2. Encryption and Decryption is never performed on the server and hence there is no key management in the server.

3. In addition to encryption, the computation of parities, packetization, and error-corrective information is done at the time the file is stored and never recomputed again.

4. The majority of the computation load for providing security and error correction is performed at the client end.

5. Authentication, key management is decoupled from data security.

These requirements are met by implementing a new security mechanism. The data to be stored is encrypted by the client and preformatted. This data unit is called the Encrypted Data Unit (EDU). The EDU is stored on the servers *as is.* The EDU is transported to be stored on servers using ECL headers. The combination of EDUs and the ECL headers are called quanta. The keys used to encrypt the data are generated using AES encryption algorithm. The keys belonging to a particular file is stored in a file and encrypted. This file is stored in a centralized key server. A valid client can access the file on servers stored in the quanta form. The clients fetch the keys from the key server and perform decryption. Figure 1 shows the network architecture that would be used to implement the security mechanism.

Several parameters that are used for read and write operation, encryption, and decryption are included in the ECL header. This is discussed in the next section.

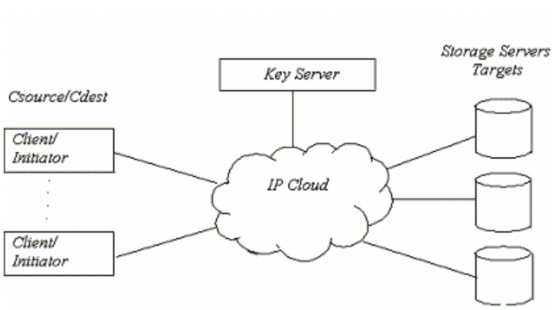

Figure 1. System Setup for data storage centric network

4. Effective Cross Layer

A single layer satisfying the specified objectives is defined as the ECL. Figure 2 shows the iWARP stack for iSCSI along with the security layer, IPsec. Figure 3 shows the corresponding ECL. In the following paragraphs we identify several common features in different layers. The necessary and common functionalities are then retained and incorporated into the ECL and the rest are excluded. These functionalities are designed such that they are scalable over a WAN.

4.1. ISCSI functionalities of the ECL

iSCSI is an adaptation of SCSI for accessing data over the Internet. Hence most of the SCSI functionalities that are part of iSCSI are retained in ECL [5]. Headers and data digests in iSCSI were not inherited from SCSI. These were added for error correction in the iSCSI protocol. Hence header and the data digests are excluded from the ECL. Instead, a single checksum for the entire quanta traversing the channel are utilized.

4.2. Copy avoidance functionalities of the ECL

Most of the iWARP functionalities are retained in the ECL. Messages framed using MPA protocol is obviated by defining a constant MTU in the SAN. Further, the quanta size is always chosen less than or equal to the minimum MTU.

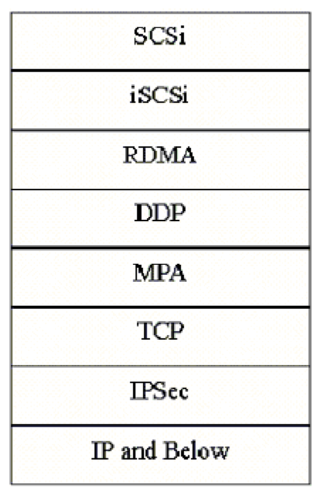

Fig 2. iWARP suite for iSCSI

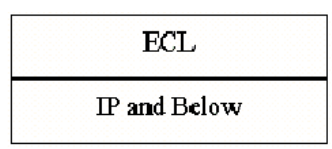

Fig 3. Effective Cross Layer

This obviates the necessity of fragmentation and thus there is no need for markers. The minimum MTU over a path between a source and the destination can be discovered, before any transmission occurs.

The DDP protocol requires the *tags* or the addresses of buffers at the destination. These tags when available at the source enable *direct data placement*. As a consequence the quanta do not subject themselves to reassembly buffering and kernel copies. Hence they are directly placed at the appropriate SCSI buffers from the NIC.

4.3. Transport functionalities of the ECL

Sequencing [6] information is retained from the iSCSI headers. Thus the sequencing information at the TCP layer can be excluded. Congestion and flow control [6] algorithms are implemented in TCP and they are based on *sliding windows* [6]. The windows are based on sequence numbers. ECL has sequencing information similar to TCP. Hence congestion and flow control mechanisms similar to that of TCP can be used.

Source and destination ports [6] identification is due to a requirement of socket level de-multiplexing. Instead direct data placement provides buffer addressing information that can be used to place the data directly into application buffers. Thus direct data placement alleviates the need for source and destination port information.

The data checksum is computed during the encryption process. Therefore there is a need only for a single quanta header checksum. Message length information that is a part of UDP can be excluded because it is also a part of the iSCSI functionality.

4.4. Security considerations in ECL

The encryption should be done only when a write or an update operation is done. During a read operation, only decryption is required. These operations are indicated in the iSCSI functionality. Hence it is not necessary to add it into the ECL separately. Authentication must be performed before every transaction. Authentication is done by the login mechanism. iSCSI login mechanism that are retained can be employed for this purpose.

The final ECL header structure that combines the features mentioned above is shown in figure 4.

5. Emulation of ECL by HYPERSCSI

In order to evaluate the performance improvements with the use of ECL, tests were conducted using Hyperscsi [7]. Hyperscsi was used because it emulates a simple version of the ECL. In the current form, Hyperscsi is equivalent to SCSI packets placed directly over Ethernet that does not involve copies, reassembly buffering, and functionality redundancies of layers.

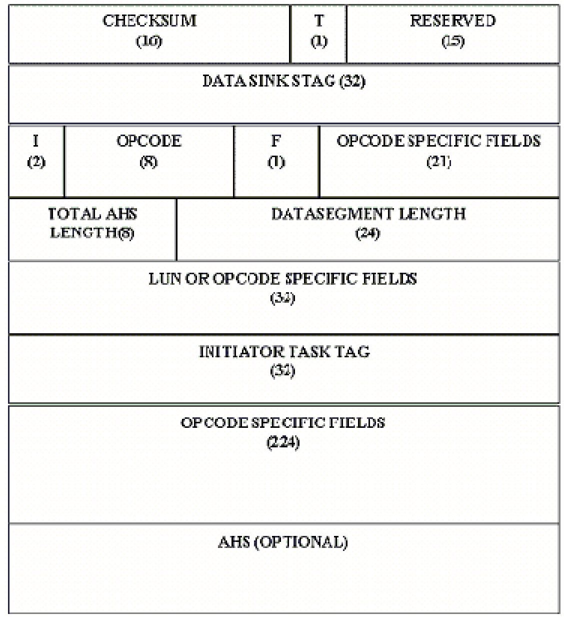

Fig 4. ECL header for WRITE operations

The test setup for ECL throughput characterization consists of two Dell Power Edge-Linux boxes (initiator and the target) with Pentium 866 MHz processors connected end to end through a Gigabit Ethernet link (82546 Dual port GBE). Linux Kernel 2.4.20-18.7 was used in the end systems. Throughput was measured using the bonnie++1.03 tool [16]. Tcpdump and Ethereal [17] was used to see the packet dumps on the end systems. iSCSI Reference 0.18 v10 [18] was used for comparison with iSCSI on TCP. 40 GB QUANTUM hard disks with ULTRA-160 SCSI bus were used at the target. Figure 5 shows the throughput snapshot tests indicating throughput improvements with the ECL.

Several read operations were performed in three different scenarios. First, read operations were performed using only the SCSI protocol. Second was performed using Hyperscsi and the third repeated with UNH iSCSI v10. The results represent an arithmetic mean of 10 trials for a fixed MTU size of 16000. The read throughput achieved using Hyperscsi was close to direct access, whereas, the latest version of the UNH iSCSI code achieves only 219.6 Mb/s as opposed to 358 Mb/s and 363 Mb/s achieved by Hyperscsi and local disk access respectively. The improvement of 63% denotes the extent of betterment in throughput that can be obtained by getting rid of system copies, reassembly buffering and multiplicity of functionalities in the iWARP stack through the ECL emulated by Hyperscsi.

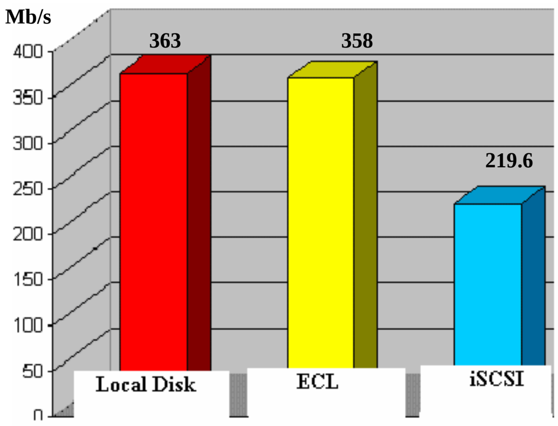

Fig 5. Read performance through the ECL

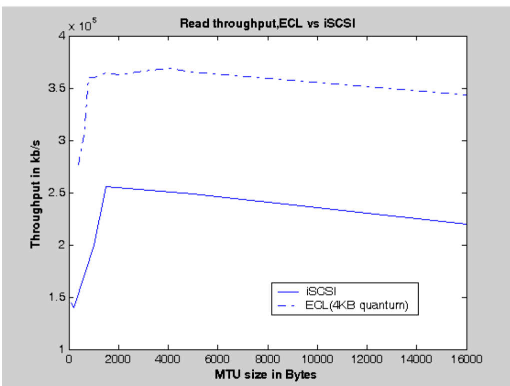

Fig 6. Bulk read throughput results

We next investigate the throughput characteristics of the ECL for reads and writes and justify the quantum data lengths equal to the path MTU.

The ECL needs to accommodate very large windows when it is operating over a network with large round trip times. The required changes are made to the Hyperscsi code to accommodate infinitely large windows. (Hyperscsi at present can accommodate only 32 segments in a window interval). 4K quanta are used for ECL. Fragmentation of quantum is allowed for MTU values less than 4K. For MTU values greater than 4K, each quantum is contained within a single Ethernet packet along with the headers.

Figure 6 compares the iSCSI throughput vs. the ECL throughput for reads for various possible MTU sizes in the Gigabit Ethernet platform. The read throughput shows a constant improvement of about 63% in the favor of ECL.

Figure 7 compares the iSCSI throughput vs. the ECL throughput for writes for various possible MTU sizes in the Gigabit Ethernet platform. The write throughput is less than the read throughput for both the iSCSI and the ECL cases, since writes are more expensive than reads. For large MTU sizes, the ECL throughput approaches the iSCSI throughput since we speculate that non-TCP/IP overheads for large MTU values in an ECL environment approach the TCP/IP overheads for large MTU values in the TCP/IP environment.

Figure 8 shows the read throughput variations for 4K quantum ECL. The throughput peaks at MTU values equal to the quantum-sized blocks. We allow for fragmentation in our tests for MTU sizes less than the quantum sizes. For MTU sizes greater than the quantum sizes, the throughput gradually decays since the per packet non-TCP/IP overheads begin to dominate. This is due to the Ethernet per packet processing overheads. For MTU sizes less than the quantum sizes, the fragmentation overheads exceed the Ethernet per packet overheads. The optimal values are thus reached for MTU sizes almost equal to the quantum sizes. The trend is also seen for writes as shown in figure 9.

An increase in quanta size increases the throughput. The write throughput for 5K path MTU and various quanta sizes are depicted in figure 10.

The tests conducted conclude the throughput characteristics of the ECL and justifies the Quantum size selections equal to MTU sizes.

6. Conclusions

Existing iSCSI storage systems exhibit low performance. Additional protocols have been proposed for improving the performance. We presented the argument that this would lead to further decrease in performance. This is due to excessive processing redundancy and several protocol layers. Further, use of protocols designed for non-storage specific requirements result in poor storage network architectures. In order to solve these problems we proposed data handling in the form of fixed data units called quanta. A new Effective Cross Layer was proposed that combines the necessary features of security, iSCSI, direct data placement, and TCP. A new security mechanism is proposed that emphasizes burden of computation on clients. Throughput improvements over the existing iSCSI are indicated by using the Hyperscsi protocol for emulation. The characteristics of the ECL throughput are noted.

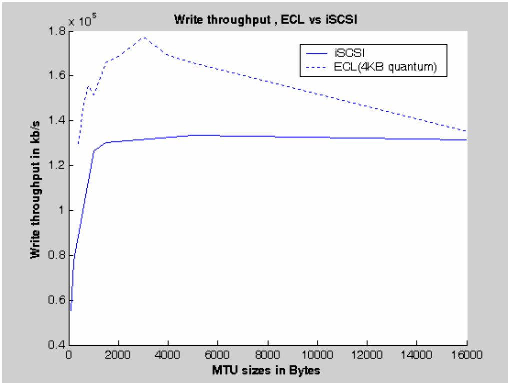

Fig 7. Bulk write throughput results

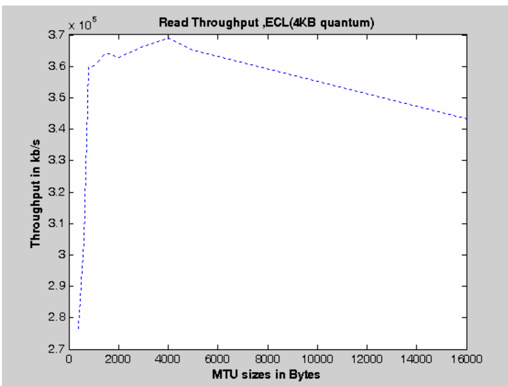

Fig 8. ECL bulk read throughput results

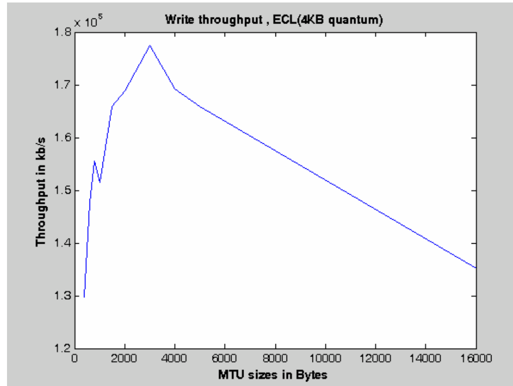

Fig 9. ECL Bulk write throughput results

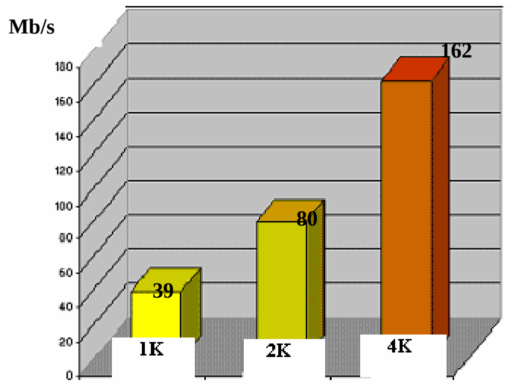

Fig 10. Write Throughput for 1K, 2K and 4K-quantum sizes- in Mb/s for 5K path MTU

References

[1] Hemal Shah, James Pinkerton, Renato Recio, and Paul Culley, *Direct Data Placement over Reliable Transports*, IETF internet draft, draft-shah-iwarpddp-00.txt (Work in progress), October 2002

[2] R.Recio, P.Culley, D.Garcia, and J. Hilland, *RDMA Protocol Specification*, IETF internet draft, draft-recio-iwarp-rdmap-00.txt, October 2002

[3] P.Culley, U.Elzur, R.Recio, and S.Bailey et. al, *Marker PDU Aligned Framing for TCP specification*, IETF internet draft, draft-culley-iwarp-mpa-03.txt, June 2003

[4] S Kent, *IP Encapsulating Security Payload (ESP)*, IETF internet draft, draft-ietf-ipsec-esp-v3-06.txt, July 2003

[5] Julian Satran, Costa Sapuntzakis, Mallikarjun Chandalapaka and Efri Zeidner, *iSCSI*, IETF internet draft, draft-ietf-ips-iSCSI-20.txt

[6] R Stevens, *TCP/IP illustrated*, *Volume 2*, Addison-Wesley, November 2001

[7] **http://www.nst.dsi.a-star.edu.sg/mcsa/**

[8] Yingping Lu and David H.C.Du, *Performance Study of iSCSI-Based Storage Subsystems*, IEEE communications magazine, August 2003

[9] Yongdae Kim, Fabio Maino, Maithili Narasimhma, and Gene Tsudik, *Secure Group Key Management for Storage Area Networks*, IEEE communications magazine, August 2003

[10] Shuang-Yi Tang, Ying-Ping Lu, and David H.C. Du, *Performance Study of Software-Based iSCSI Security*, Proceedings of the First International IEEE Security in Storage Workshop, SISW'02, 2003

[11] StephenAiken, Dirk Grunwald, Andrew R.Pleszkun, and Jesse Willeke, *Performance Analysis of the iSCSI protocol*, IEEE MSST, 2003

[12] **http://www.rdmaconsortium.org**

[13] Rodney Van Meter, Gregory G. Finn, and Steve Hotz, *VISA: Netstation's Virtual Internet SCSI Adapter*, Proceedings of the eighth international conference on Architectural Support for Programming

Languages and Operating Systems**,** Pages 71 - 80 , October 1998

[14] Peter Druschel and Larry L. Peterson, *A High-Bandwidth Cross-Domain Transfer Facility,* Proceedings of the fourteenth ACM symposium on Operating Systems Principles**,** volume 27 issue 5, December 1993

[15] David D. Clark and David L. Tennenhouse, *Architectural Considerations for a New Generation of Protocols*, ACM SIGCOMM Computer Communication Review, Proceedings of the ACM symposium on Communications architectures & protocols, Volume 20 Issue 4 , August 1990

[16] **http://aixpdslib.seas.ucla.edu/packages/bonnie++.html**

[17] **http://www.ethereal.com/**

[18] **http://sourceforge.net/projects/unh-iscsi/**