DEWS2004 4-A-03

iSCSI における小粒度アクセスの特性解析

山口　実靖† 　小口　正人†† 　喜連川　優†

† 東京大学生産技術研究所　〒 153-8505 目黒区駒場 4-6-1
†† お茶の水女大学理学部情報科学科 〒 112-8610 東京都文京区大塚 2-1-1
E-mail: †{sane,kitsure}@tkl.iis.u-tokyo.ac.jp, ††oguchi@computer.org

あらまし　規模拡張性の高さや導入コスト低さなどの利点を持つ SAN として IP-SAN や iSCSI が注目を集めている. 本稿では, TCP の振る舞いを考慮した ショートブロックによる iSCSI ストレージアクセスの特性解析について述べる. まず, 既存の iSCSI 実装を用いその性能を評価し, そのターンアラウンドタイム性能が必ずしも高く無いことや実装により性能が大きく異なる事を示す. 次に開発した iSCSI 解析システムを用い, 性能劣化が TCP の Nagle アルゴリズムや遅延 Ack に起因していることを示し, その回避方法とそれによる性能向上について述べる.

キーワード　iSCSI, ネットワークストレージ, IP-SAN

Analysis of iSCSI Storage Access with Short Blocks

Saneyasu YAMAGUCHI†, Masato OGUCHI††, and Masaru KITSUREGAWA†

† Institute of Industrial Science, University of Tokyo　4-6-1 KOMABA MEGURO-KU, TOKYO 153-8505, JAPAN
†† Department of Information Sciences Faculty of Science Ochanomizu University, 2-1-1 Otsuka Bunkyo-ku, Tokyo, 112-8610 Japan
E-mail: †{sane,kitsure}@tkl.iis.u-tokyo.ac.jp, ††oguchi@computer.org

Abstract　IP-SAN is expected as scalable and cost-effective SAN, and iSCSI is also expected data transfer protocol of IP-SAN. In this paper, authors describe detailed analysis of iSCSI storage access with short blocks. First, we show turn around time of short block iSCSI storage access with several iSCSI implementations. We found differences among implementations are significantly large. Second, we analyze these iSCSI implementations with our analysis system and show that performance degradations are caused by TCP Nagle algorithm and Delayed Ack.

Key words　iSCSI, Network Storage, IP-SAN

1. はじめに

超大容量のデータを高速に処理するためのシステムとして, SAN(Storage Area Network) [1] が注目を集めており, その実績は高い評価を得ている. しかし現世代の SAN は, FC(Fibre Channle) を用いた FC-SAN であり, FC の導入コストの高さ, FC 管理技術者の少なさ, FC の接続距離の限界, などの問題点も明らかとなってきている. これらの問題点を解決する SAN として, Ethernet と TCP/IP を用いた SAN である IP-SAN や, そのためのデータ転送プロトコルである iSCSI [2]〜[5] に大きな期待が集まっている. 著者らは, iSCSI 用いたストレージアクセスの性能向上手法として文献 [6] において高遅延ネットワーク環境における iSCSI を用いた連続的なデータ転送のスループット性能の向上させるにはブロックサイズの拡大がもっとも重要であることを述べ, 文献 [7] において, 大きなブロックサイズを用いた際にはローカルデバイス輻輳が発生し TCP が出力を制限してしまい高い性能が得られないことと, それの回避により性能がさらに向上されることを述べた. そして文献 [8] において, 図 1 の様に iSCSI システムを網羅的に観察することにより性能劣化原因の発見を可能とする iSCSI 解析システムを提案した.

本稿では, ショートブロックサイズによる小粒度の iSCSI ストレージアクセスの性能について述べる. ショートブロックアクセスは DBMS やファイルアクセスなどに用いられ, ターンアラウンドタイムの短縮が重要であると考えらる.

本稿ではまず既存の iSCSI 実装を紹介し, 小粒度の iSCSI アクセスの性能を紹介し, 実装によりその性能が大きく異なることを示す. そして, それら各実装の振る舞いに対する詳細な解析を紹介し, これらの性能差が Nagle のアルゴリズム [9] や遅延確認応答 [10] などの TCP の振る舞いに起因していること,

これらを回避することによりその性のを大きく向上できることを示す.

本稿は以下のように構成される. 第 2. 章で研究背景として, IP-SAN や iSCSI の重要性, 関連する既存の研究成果, 本稿で言及する TCP のアルゴリズムについて述べる. 第 3. 章において各実装における, iSCSI を用いた小粒度アクセスのターンアラウンドタイム性能について紹介し, 実装により性能が大きく異なることを示す. 第 4. 章において前章の実験の振る舞いの解析を述べ, 実装による性能の差の原因が TCP の振る舞いにあることを述べる. 最後に, 第 5. 章において本稿をまとめる.

2. 研究背景

2.1 ストレージの現状

ストレージシステムにおいては, データバックアップ等の管理作業が非常に重要となる. しかし, ストレージの管理に必要とされる人件費は, ストレージの導入コストの 6 倍〜10 倍以上 [11], [12] と言われ, ストレージの管理コストの高さが計算機システムの大きな問題となっている. ストレージ管理コストの削減方法の一つとして, SAN (Storage Area Network) の導入があり, その効果は高く評価されており, 既に多くの企業で採用されている. そして, これからも SAN 市場は拡大を続け世界の SAN 市場は 2006 年までに 1.4 倍以上に成長し, 日本国内に限れば 2 倍以上に成長すると予想されている [1], [13]. 特に, 新規の SAN の導入への投資の成長が著しく, 2002 年, 2003 年ともに約 1.5 倍の成長を遂げている [14]. このように SAN は広く活用されており, 今後さらにその重要性を増していくと考えられている. 現在, SAN の主流は FC を用いた FC-SAN であるが, 今後は IP-SAN も普及を始め, そのシェアが増えていくと予想されている [14].

2.2 iSCSI

SAN はストレージ専用の高速ネットワークであり, ストレージを一カ所に集約し各サーバ計算機は SAN を用いてこれに接続する. サーバ毎に管理されていたストレージを一カ所に集約して管理することにより, 管理コストは大幅に削減されると言われている. しかし現世代の SAN は FC を用いているため, FC の管理技術者が少ない, FC の接続距離には限界がある, FC の相互接続性は必ずしも高くない, FC の導入コストが高い, などの問題点も明らかとなってきており, これらの問題を解決する IP-SAN と iSCSI に期待が高まっている.

IP-SAN とは, Ethernet と TCP/IP を用いて構築する SAN であり, 管理可能技術者が多い, 接続距離に限界がない, 相互接続性が高い, 導入コストが低い, などの利点を持っている. IP-SAN 用のデータ転送プロトコルとしては SCSI プロトコルを TCP プロトコルの中にカプセル化し IP ネットワーク上で転送するブロックレベルのプロトコル iSCSI が代表的なプロトコルである. 多くの場合, 物理層, トランスポート層には Ethernet が用いられ, 代表的なプロトコルスタックは図 1 の様になる. iSCSI は, 2003 年 2 月に IETF [2], [3], [15] に正式に承認され, 現在各種 OS へのドライバ や HBA の提供が始まっている.

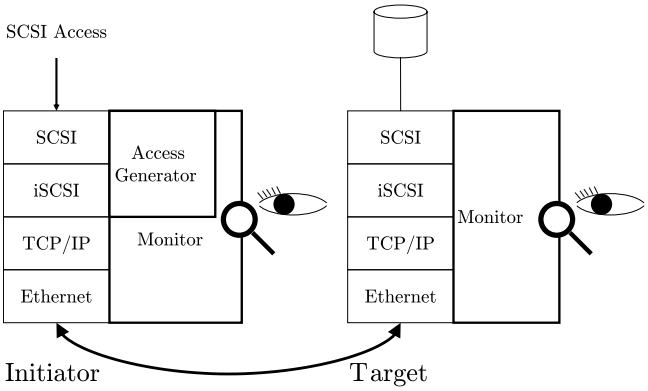

図 1　iSCSI プロトコルスタックと iSCSI 解析システム

2.3 本研究の位置づけ

iSCSI ストレージアクセスでは SCSI プロトコルを TCP/IP プロトコルの中にカプセル化してネットワークで転送する. しかし, SCSI プロトコルは必ずしも高遅延ネットワーク上の使用や, TCP の様に独自にフローコントロールを行うネットワーク上での使用を想定しておらず, 著者らはこれらの組み合わせにより生じる問題を考慮しなくては必ずしも高い性能は得られないと考える. ストレージアクセス手法としてはスループット性能が重要となるシーケンシャルアクセスや, ターンアラウンドタイム性能が重要となるショートブロックアクセスなどがあるが, 我々は高遅延環境下での連続データ転送時のスループット性能においては, SCSI 実装が作成するブロックサイズが十分に大きくなく高遅延環境下での性能が著しくて低下してしまうこと, この拡大によりその性能を大きく向上させることが可能であることを文献 [6] において示し, ブロックサイズの拡大にともなう SCSI プロトコルが発生させるバーストと TCP 実装のフローコントロールの組み合わせにより性能が低下してしまうことおよびその回避により iSCSI シーケンシャルアクセス性能が大きく向上できることを文献 [7] において示した. 本稿ではさらに高遅延環境下におけるショートブロックのターンアラウンドタイム性能について考察する.

また, ストレージの性能については既存の研究成果が多く発表されているが iSCSI プロトコルの振る舞いが性能に与える影響に関しては十分な考察がなされていないことに着目し, 対象を iSCSI に絞って考察を行う. 具体的には, ストレージデバイスの影響と iSCSI のプロトコルの影響を明確に分離するために iSCSI Target デバイスはメモリモードで動作させた. これは十分に高速であり性能に影響を与えないストレージと見なすことが可能である. これにより純粋な iSCSI プロトコルが性能に与える影響について考察を行うことが可能となる. 実ストレージに対し iSCSI を用いる際はさらにストレージデバイスについても考慮する必要があるが, ネットワーク遅延が十分に高い環境下では本稿で後述する往復回数の制御がターンアラウンドタイム性能に支配的になると考えられる. また, 同様に後述する遅延確認応答の動作による影響は数 10ms(第 2. 6) から数 100ms 秒程 [16] の規模となり, ストレージデバイスの動作と比べ十分に大きく, これもターンアラウンドタイム性能に対し支配的と

なると考える. 詳細については, 第 4.6 章で後述する.

2.4 関連研究

文献 [11] において, Ng らは独自の SCSI over IP 実装を用いて 8KB のブロックサイズにおけるシーケンシャルアクセスやランダムアクセスの性能を測定している. 同測定から, ランダムリードの性能がネットワーク遅延時間の増加にともなり単調に減少することが確認されているが, TCP の振る舞いによりこの性能が大きく変化することについては言及されていない.

文献 [17], [18] において, Sarkar らは CPU 使用率に着目し低遅延環境における iSCSI 性能について考察している. iSCSI の処理には多くの CPU 資源の消費が必要となること [17] や, その解決策としてのハードウェアによる TCP/IP 処理の効果 [18] について述べているが, TCP のフローコントロールアルゴリズムに起因する問題については言及しておらず, 本研究とは着目点が異なる.

2.5 Nagle のアルゴリズム

TCP/IP 通信においてペイロードのサイズが MSS(Maximum Segment Size) 未満の微少なパケット (一般にこれを “タイニーグラム” と呼ぶ) を送信することはパケットの数の増加, TCP/IP ヘッダの等のオーバーヘッドの増加を招く. この MSS 未満のサイズのパケットの問題への対処法として, Nagle のアルゴリズム [9] が, 提案されており既存の TCP 実装の多くで実装されている [16]. このアルゴリズムでは, Ack が未受信である微少パケット (以後これは “MSS 未満のパケット” を意味する) の送信を最大 1 個までとする. すなわち, 微少パケットをすでに 1 個送信しておりかつそのパケットに対する Ack が未受信である状態において MSS 未満のデータ送信要求が発生した場合は, 送信済みの微少パケットに対する Ack を受信するまで, あるいは送信要求データ量が MSS を越えるまで TCP 実装はその送出を保留することとなる. これは多数の微少パケットが送信されることを回避させるが送信の遅延を招くこともあため, 多くの TCP 実装では TCP_NODELAY オプションにより Nagle のアルゴリズムは無効化するこも可能となっている.

2.6 遅延確認応答

TCP はスライディングウィンドウシステムを用いており, データを受信した通信者は送信者に対して確認応答 (Ack) を送信し, 正常に受信がなされたことを知らせる. TCP の Ack 情報は TCP ヘッダの Ack フィールドに記載されているため, 全ての TCP パケットは Ack フィールドを持つ. よって, 受信者から送信者への送信希望データが存在するときはそのデータの送信の際に TCP ヘッダにおいて Ack 情報も併せて送信するの効率的となる (一般にこれを “ピギーバック” と呼ぶ). 送信希望データが存在しない場合は Ack 情報の伝達のために独立した TCP パケット (ペイロードを含まず TCP/IP ヘッダのみで構成されるパケット) を生成しこれを送信することとなる. TCP では “遅延確認応答” [10] が実装されており, 連続していない 1 個のパケットを受信しかつ受信者がピギーバック可能である逆向きの送信データを持たない場合に, 逆向きの送信要求が発生することを期待し Ack 送信を延期する [16]. 延期時間は定められておらず, TCP 実装に依存するが, 本稿の実験環境 (Linux

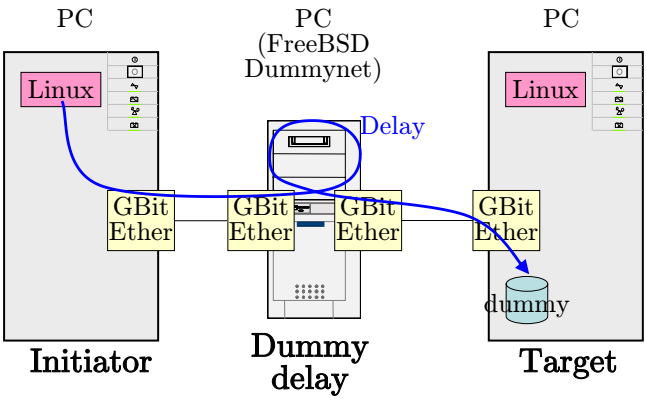

図 2 実験環境 1

表 1 性能評価実験環境 2 : 使用計算機

CPU	Pentium 4 2.80GHz
Main Memory	1GB
OS	Linux 2.4.18-3
Network Interface	Gigabit Ethernet Card Intel PRO/1000 XT Server Adapter

表 2 性能評価実験環境 3 : 使用計算機

CPU	Pentium4 1.5GHz
Main Memory	128MB
OS	FreeBSD 4.5-RELEASE
Network Interface	Gigabit Ethernet Card Intel PRO/1000 XT Server Adapter × 2

TCP/IP 実装を使用) では延期期限を “カーネルタイマの発生まで” としており, このタイマは 50ms 毎に発生する. カーネルタイマは OS の起動時刻から指定時間毎に発生し TCP 実装とは独立しているため, 50ms は延期時間の最大値となる.

3. iSCSI 実装の性能測定

本章において小粒度 iSCSI アクセスの性能を測定し, それを紹介する.

3.1 実験環境

性能評価実験は以下の環境で行った. 図 2 のように, iSCSI Initiator(サーバ) と iSCSI Target(ストレージ) を Gigabit Ethernet で接続して TCP/IP 接続を確立する. Ethernet の接続は, クロスケーブルで直結するか, あるいは途中に人工的な遅延装置として FreeBSD Dummynet [19] を挟んでクロスケーブルで接続をした. Initiator, Target, Dummynet はすべて PC 上に構築し, Initiator と Target には Linux を, 遅延装置には FreeBSD をインストールした. Initiator, Target の PC の詳細を表 1 に, 遅延装置の PC の詳細を表 2 に示す.

また, iSCSI の実装としては以下のものを用いた (1) ニューハンプシャー大学 InterOperability Laboratory(以下, “IOL” と呼ぶ) [20], [21] が配布する iSCSI 実装 (iSCSI draft 18 準拠のもの), (2) 同大学 IOL が配布する iSCSI 実装 (iSCSI draft 20 準拠のもの), (3)Intel 社が配布する iSCSI 実装 (draft 16 準拠), また, これらの実装に対し著者らが変更を施したもの (後

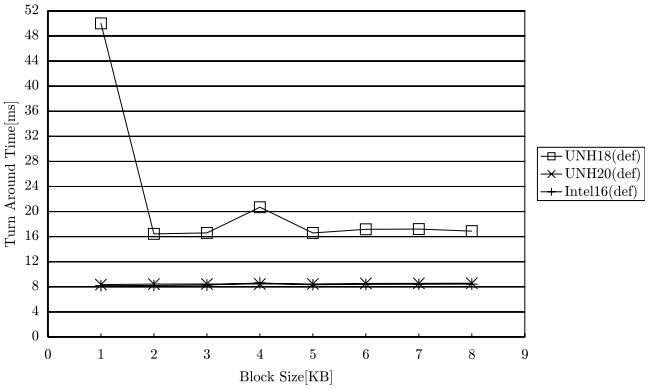

図 3　実験結果：片道遅延時間 4ms

述) を被実験実装として用いた. 以後, ニューハンプシャー大学の iSCSI 実装で draft 18 準拠であるものを “UNH 18” と, draft 20 準拠のものを draft 20 準拠のものを “UNH 20” と呼び, Intel の iSCSI(draft 16 準拠) を “Intel 16” を呼ぶ.

3.2　実 験 方 法

前節の実験環境により, 以下の実験を行い各実装の評価を行った. まず, Initiator 計算機 と Target 計算機において, iSCSI Initiator, iSCSI Target を起動させる. この際, iSCSI Target はメモリモードで起動させる. よって, iSCSI Target デバイスへのアクセスは物理的なディスクへのアクセスを伴わない. 次に, Initiator 計算機から Target 計算機に対し iSCSI 接続を確立させる (Initiator 計算機の OS において遠隔ディスクのマウントを行う). そして, 作成したベンチマークソフトウェアにより, iSCSI 接続のディスクの raw デバイスに対して, システムコール `read()` を連続して発行しその性能の平均を測定する.

3.3　性能測定結果

前節の実験により, 各実装の性能を測定し, 図 3 の結果を得た. 同図は, 片道遅延時間 4ms における各実装のターンアラウンドタイムを表している. “UNH18(def)” は UNH 18 実装を用いて測定したものであり, 同実装に対し著者らが改変を行っていないものである. “(def)” は default を意味し後述する著者らが改変を行ったものと区別するために “(def)” と記す. 同様に “UNH20(def)” は UNH 20 実装を用いて測定したものであり同実装に対して改変が行われていないもの, “Intel16(def)” は Intel 16 実装を用いて測定したものであり改変が改変が行われていないものである. 横軸はブロックサイズを表し, ベンチマークプログラムにおける システムコール `read()` の発行の際に引数として指定したサイズであり, 実際にネットワークで転送される iSCSI PDU での Read コマンドのブロックサイズもこれに等しい (システムコール時に大きいブロックサイズを指定しても実際に発行される iSCSI PDU における Read コマンドのブロックサイズがこれよりも小さいことがあるが [6], 本稿で述べる小粒度のアクセスにおいてこれは発生しない). 縦軸はターンアラウンドタイムを表し, システムコール `read()` が発行されてからそれが終了するまでの時間を表している.

同結果より, ターンアラウンドタイムは UNH18(def) において約 16ms であり (ただしブロックサイズ 1KB が例外として, 16ms から大きくはずれている) , UNH20(def), Intel(def) において約 8ms であることが確認された. すなわち, 本実験結果の例においてターンアラウンドタイムは実装の違いにより, 約 2 倍の性能差が現れること (1KB を除く), ブロックサイズ 1KB に 6 倍の性能差が現れることが確認された.

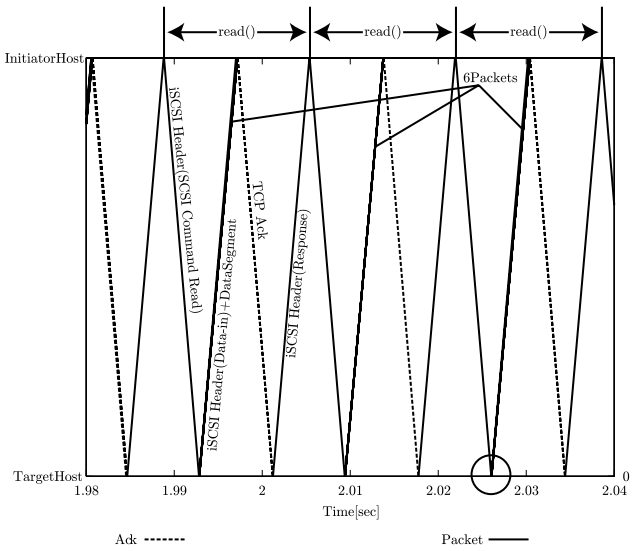

図 4　UNH 18 実装, 片道遅延時間 4ms Block Size 8KB, TCP パケットの移動の時間軸可視化 A(縮小)

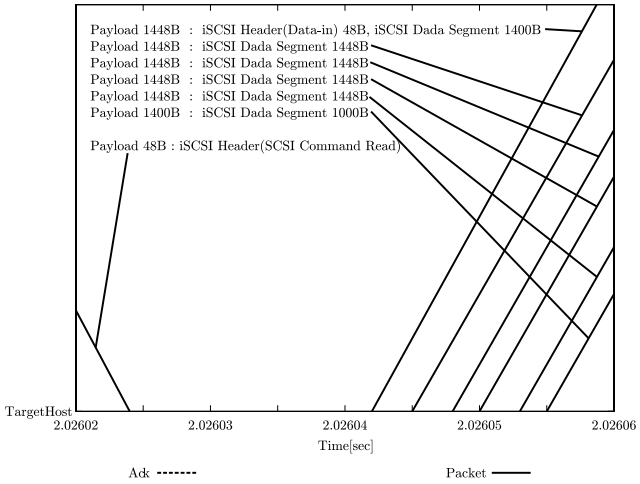

図 5　UNH 18 実装, 片道遅延時間 4ms Block Size 8KB, TCP パケットの移動の時間軸可視化 B(拡大)

4.　解　　析

次に, 著者らが開発した iSCSI 解析システム [8] を用いて前章の実験結果を解析し, 前述の性能差が現れる理由を示す.

4.1　UNH 18 の解析

本節では, UNH 18 実装の振る舞いの解析について述べる. 片道遅延時間 4ms において UNH18 実装を用いブロックサイズ 8KB の `read` を行ったときの TCP パケットの転送を可視化したものを図 4 に示す. また, 同図内の円において示した部分 (これは Target 計算機に SCSI Command Read の iSCSI PDU が到着し, Target 計算機が Data-in の iSCSI PDU を送り出す瞬間である) の拡大図を図 5 に示す.

まず, 高遅延環境下におけるショートブロックアクセスのターンアラウンドタイムの多くが “データの転送時間” ではなく, ネットワークの “遅延時間” に費やされていることが視覚的に確認できる. よって, ネットワークの遅延時間の短縮や往復回数の削減がターンアラウンドタイムの短縮には効果が大きいと考えられる. 次に, 図 4 より, iSCSI Target は SCSI Command Read の iSCSI PDU 受信の後まず iSCSI Data-in PDU を送信し, Initiator からの TCP Ack の受信の後に iSCSI Response を送信しているいることが確認された. すなわち, 1 回の iSCSI Read は iSCSI PDU SCSI Command Read (I→T), iSCSI PDU Data-in (T→I), TCP Ack (I→T), iSCSI PDU Response (T→I), により構成され Initiator - Target 間の 2 往復を要している (ただし, “I→T” は Initiator から Target 方向, “T→I” は Target から Initiator 方向を意味してる) . これにより, “ストレージデバイスの動作時間に対してネットワーク遅延時間十分に大きい” という仮定のものでは, ターンアラウンドタイムは, 4 × 片道遅延時間と同程度になると言える. また, 図 5 より, iSCSI Target ドライバにより, 8KB のデータ含む iSCSI Data-in PDU(48B の iSCSI Header と 8KB iSCSI Data Segment により構成され, 合計 10296B となる) が TCP 実装に渡され, これが MSS の 1448B 毎に分割され Ethernet により送信されている様が確認できる. 本実験システムの例においては第 1 層, 2 層に Etherent を用いているため MTU(Maximum Transmission Unit) が 1500B であり, TCP オプションのタイムスタンプオプションが有効となっている. IP ヘッダはオプションがされていない場合は 20B であり, TCP オプションも同様にオプションが付加されていない状態において 20B である. また, TCP のタイムスタンプオプションには 10B 必要となる. そして, TCP オプションの合計サイズは 4B の整数倍とするよう定められているため, パディングデータとして 1B の Nop オプションが 2 個付加され, 全 TCP オプションの合計サイズは 12B となる. 以上より, TCP/IP ヘッダとして, 52B が使用され, MSS は 1448B となる.

また, UNH 18 を用いて, ブロックサイズ 1KB で `read()` を行ったときにターンアラウンドタイムが非常に大きくなることが確認されている. UNH 18 実装を用いて, 片道遅延時間 4ms, ブロックサイズ 1KB の `read()` を行ったときのパケット転送を可視化したものを 図 6 に示す. 図 4 同様に, iSCSI Read PDU の送信は TCP Ack の受信を待ってから行われている. ただし, 図 6 おいては iSCSI PDU (SCSI Command Read) に要求されたデータサイズは 1KB であり, Initiator 計算機の TCP 実装が受け取る TCP パケットは 1 個となる (Data-in PDU は 48B の iSCSI Header と 1KB の Data Segment で構成され, これは MSS より小さい) . 第 2.6 節で前述の様に, 孤立した単数の TCP パケットを受信した Initiator 計算機の TCP 実装では遅延確認応答が動作し TCP 実装は Ack の送信を大きく遅らせる. 同図より 50ms 毎に発生するカーネルタイマの発生まで Ack の送信を遅らせており, それにより Target による iSCSI Response の送信が大きく遅れ, 結果としてターンアラウンドタイムが著しく低下していることが確認された.

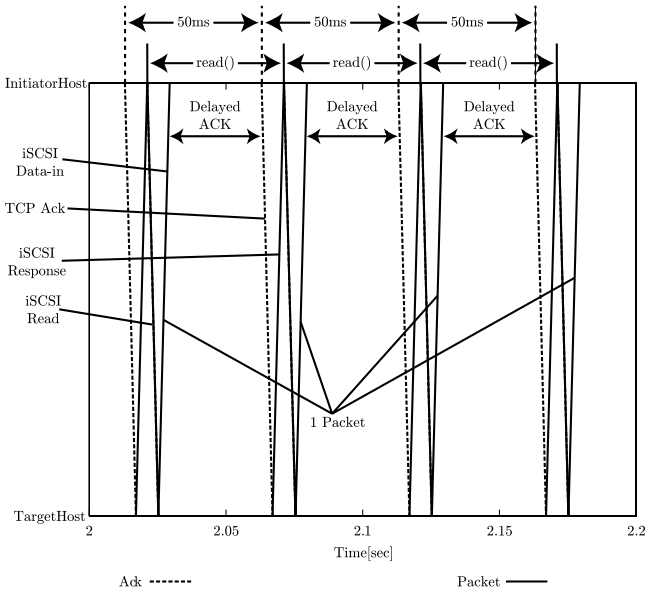

図 6　UNH 18 実装, 片道遅延時間 4ms Block Size 1KB, TCP パケットの移動の時間軸可視化

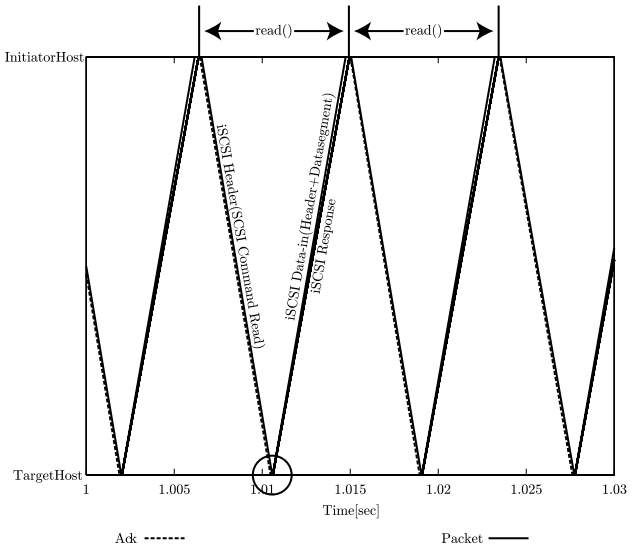

図 7　Intel 16 実装, 片道遅延時間 4ms Block Size 8KB, TCP パケットの移動の時間軸可視化 A(縮小)

遅延確認応答が動作しターンアラウンドタイムが著しく長くなる同様の現象はブロックサイズが 512B の時も発生することが確認されている. また往復回数が増加する現象と異なり, 遅延確認応答の動作によるターンアラウンドタイムの増加はネットワークの片道遅延時間に依存しておらず TCP 実装が固定的に持つタイマの長さに依存している. よって, 遅延確認応答の動作によるターンアラウンドタイムの増加率は片道遅延時間が短い程大きくなり, ブロックサイズ 1KB, クロスケーブル接続 (片道遅延時間 0.1ms 程度) の例において 100 倍以上となる. 遅延確認応答の動作と往復回数の増加を回避した場合のターンアラウンドタイムは 0.2ms 程度であり, 遅延確認応答が動作した場合のターンアラウンドタイムは 50ms 以上となる.

4.2　Intel 16 の解析

片道遅延時間 4ms において Intel 16 の実装を用いブロックサ

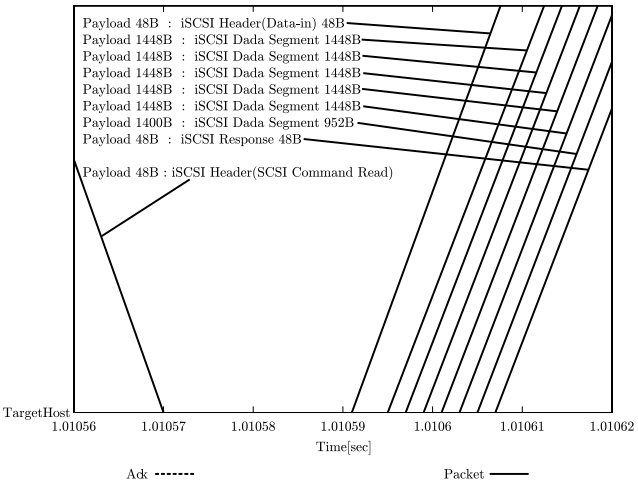

図 8　Intel 16 実装, 片道遅延時間 4ms Block Size 8KB, TCP パケットの移動の時間軸可視化 B(拡大)

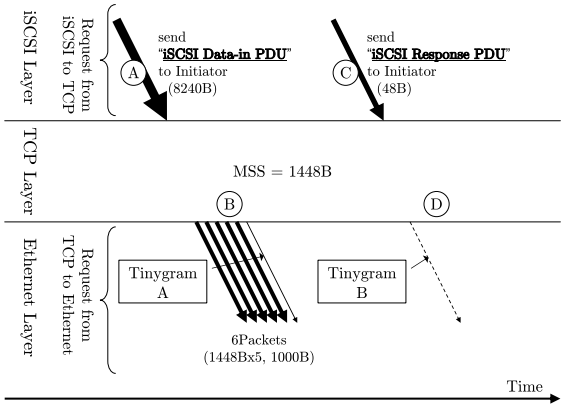

図 9　Nagle のアルゴリズム (UNH 18)

イズ 8KB の read の TCP パケットの転送を可視化したものを図 7 に示す. 同図で円において示した部分 (これは Target 計算機に SCSI Command Read の iSCSI PDU が到着し, Target 計算機が Data-in および Response の iSCSI PDU を送り出す瞬間である) の拡大図を図 8 に示す (ただし, 図 7 の円内の Ack は Target 到着時刻が 1.010470 であり図 8 内に記されていない) .

同 7 から UNH 18 実装と異なり Intel 16 実装では 1 回のシステムコール `read()` の完了に必要とされるネットワークの往復は 1 回である (ターンアラウンドタイムは 2 × 片道遅延時間と同程度) ことが確認でき, その結果 Intel 16 iSCSI 使用時のターンアラウンドタイムが UNH 18 使用時の約半分となることが確認できる. また, 図 8 より Intel 16 では iSCSI Data-in PDU の Header 部 (48B), iSCSI Data-in PDU の Datasegment 部 (8KB), iSCSI Response PDU が個別に送信要求されていることや (UNH 18 実装においては iSCSI Data-in PDU の Header 部と Datasegment 部は結合されて送出要求されていた), Initiator 計算機からの TCP Ack の受信を待つことなく iSCSI Response の送信を行っていることが確認できた. 両図より TCP Ack の受信を待たずに iSCSI Response を送信していることが UNH 18 実装と比べて 往復回数が 1 回少なくなっている原因であることが分かる.

4.3　UNH 20 の解析

UNH iSCSI 20 の振る舞いの解析を行うと, 図 7 同様に, システムコール `read()` 毎に必要とされるネットワークの往復回数は 1 回である. 結果として ターンアラウンドタイムは 2 × 片道遅延時間と同程度どなる. ただし, UNH 20 においては iSCSI Data-in PDU の後の iSCSI Response が省略されている.

4.4　iSCSI 実装の振る舞いの比較

本節において, 前述の各 iSCSI 実装の振る舞いの比較を行い, 実装により iSCSI 小粒度アクセスのターンアラウンドタイムが大きく異なることが TCP の Nagle のアルゴリズムに起因していることを示す. 前節により 1 回のシステムコール `read()` にネットワークの 2 往復を要する UNH 18 では, iSCSI Data-in PDU の送信の後の iSCSI Response PDU の送信は TCP Ack の受信を待つ必要があり, これにより往復回数が 1 回増加していることが確認された. UNH 18 iSCSI 実装では, TCP_NODELAY オプションは有効となっておらず, Ack 未受信の微少パケットを 2 個以上送信することを許さない. また, UNH 18 の iSCSI Target ドライバ実装は iSCSI Data-in PDU の送信と iSCSI Response PDU の送信を分離して TCP 実装に依頼する. これにより TCP 実装は iSCSI Data-in PDU を下位レイヤー (Ethernet ドライバ) に送出し, その後に iSCSI Response PDU を送出する. その結果, Nagle のアルゴリズムにより iSCSI Response の送信を TCP Ack の受信まで保留することとなる. 本例における Nagle のアルゴリズムの動作の様子を図 9 に示す. 同図は, Target 計算機において, iSCSI 実装が iSCSI Data-in PDU の送信要求及び iSCSI Response PDU の送信要求を TCP 実装に対して行う際の iSCSI 実装, TCP/IP 実装, Ethernet およびそのドライバ実装の振る舞いの模式図である. `read()` のサイズは, 8KB である. 同図のようにまず iSCSI Target 実装が iSCSI Data-in PDU の送信要求を TCP 実装に対して行う. 8KB の `read()` の場合 iSCSI Data-in PDU は 48B の iSCSI Header と 8KB の Data Segment で構成され, そのサイズは 8240B となる. Data-in PDU を受け取った TCP 実装はこれを MSS 毎に分割し, それらに TCP/IP ヘッダを付加し, 下位層である Ethernet 層に送信する. 前述の様に, 本実験環境における MSS は 1448B であるため, 8240B の Data-in PDU は 5 個の 1448B と 1 個の 1000B の合計 6 個のセグメントに分割され, それぞれに TCP/IP ヘッダが付加される. ここで, 最後のセグメント (図中の “Tinygram A”) は 1000B であるため MSS 未満の微少パケットとなり, Nagle のアルゴリズムが有効である場合は TCP 実装はこれ以降は微少パケットの転送を依頼されてもその送出を保留する. 次に, iSCSI Target 実装が iSCSI Response PDU の送信要求を TCP 実装に対して行う. iSCSI Response PDU は, iSCSI Header のみにより構成されており通常 48B である. よって, iSCSI Response PDU の送信は, 2 個目の微少パケットの送信

(図中の “Tinygram B”) を招くこととなり Nagle のアルゴリズムが有効である場合は, この送信は TCP Ack の受信まで保留されることとなる. 具体的には, iSCSI Response PDU の送信は iSCSI Data-in PDU の最終セグメントである “Tinygram A” に対する TCP Ack を Initiator 計算機の TCP 実装から受信する時刻以降となり, ネットワークの 1 往復 (I→T iSCSI Data-in, T→I TCP Ack) 時間分 iSCSI Response の送信が保留されることとなる. ネットワーク遅延が十分に大きく, 遅延時間がターンアラウンドタイムに対して支配的である環境ではターンアラウンドタイムを 2 倍に増加させることとなる.

4.5 TCP オプションの変更と性能への影響

以上の様に, 小粒度の iSCSI アクセスのターンアラウンドタイムの低減には, Nagle のアルゴリズムが有効である状態における MSS 未満の微少パケットの連続作成の回避や, 確認応答が必要な状況における遅延確認応答の動作の回避が重要であることが分かった. そこで, 参考実験とした既存の iSCSI 実装に対し以下の様な改変を施し, その性能を測定した.

（1） UNH 18 iSCSI 実装に対し Nagle のアルゴリズム無効化 (TCP_NODELAY の有効化) の変更を行った.

（2） UNH 18 iSCSI 実装に対しパケット結合改変を行った.

（3） Intel 16 iSCSI 実装から Nagle のアルゴリズム無効化 (TCP_NODELAY の有効化) を削除した.

まず, 改変 (1) は, UNH 18 が `read()` 1 回に付きネットワークの 2 往復が要されているが Nagle のアルゴリズムの無効化によりこの軽減が可能であるかを確認すことを目的とする. 測定結果においては, これを UNH 18(ND) と記す. 改変 (2) は, Nagle のアルゴリズムが動作し iSCSI Response の送信が延期されてしまう原因である iSCSI 実装による TCP 実装への微少パケットの連続送信要求を回避することを目的とする. iSCSI 層と TCP/IP 層の間に結合層を追加し, この層が iSCSI 層からの要求を一旦受け取り iSCSI Data-in PDU と iSCSI Response を結合して 1 個の要求として TCP 層に依頼する. この変更は, Nagle のアルゴリズムによる iSCSI Response の延期の問題を回避することが可能であり, かつ Nagle のアルゴリズムを無効化 (TCP_NODELAY オプションの有効化) をする必要が無くなる. Nagle のアルゴリズムを無効化し微少パケットを待たずに送信することはターンアラウンドタイムの短縮に効果的であるが, パケット数の増加を招くためネットワーク負荷等を考慮した場合は好ましくない. 測定結果においては, これを UNH 18(conc) と記す. 改変 (3) は, 確認のために Intel 16 iSCSI 実装に対し TCP_NODELAY オプションの無効化 (Nagle のアルゴリズムの有効化) を施し, その性能を確認した. 測定結果においては, これを Intel16(N-ND) と記す.

以上の実装を加えた測定結果を図 10 に示す. 同図より, UNH 18 iSCSI 実装に対し Nagle のアルゴリズム無効化 (TCP_NODELAY の有効化) や, 結合層の追加を行うことによりターンアラウンドタイム性能は大きく向上 (時間において約半減) することが確認され, 逆に Intel 16 実装から TCP_NODELAY オプションを削除することによりその性能が大きく劣化してしまうことも確認された. よって, Nagle のアルゴリズムが有効である状態において iSCSI Read に対し iSCSI Data-in と iSCSI Response を分割して送信することは, 往復回数の増加 (2 往復となる) を招きターンアラウンドタイム性能の劣化を招く. 往復回数増加の回避には, パケット数の増加が大きな問題とならない環境においては Nagle のアルゴリズムの無効化が, パケット数の増加が問題となる環境においては Data-in PDU と Response PDU の一括送信や, iSCSI Response による同期の回数の削減などが効果的であると考えられる.

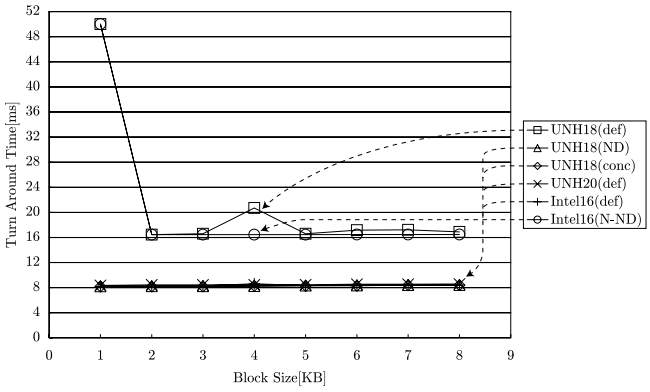

図 10 実験結果 B : 片道遅延時間 4ms

4.6 実際のストレージデバイスの性能と影響

本稿では, TCP/IP の振る舞いによる影響をそれ以外の要素の影響と明確に分離するために iSCSI Target はメモリモードで計測を行った. 分離して考察したことにより, 実デバイスの動作と併せての考察が容易になることと, 実デバイス使用時の性能に関する考察を示す.

本稿で述べた計測は全て Target をメモリモードで動作させているためストレージデバイスからの読み込み動作は十分に短い時間で終了し, ネットワーク遅延時間が支配的である環境における考察として行ってきた. 実ストレージのアクセス時間はストレージ製品に強く依存してしまうが, 例として文献 [11] で性能が紹介されているストレージ “IBM DDYS” を用いて考察を行う. 同文献によれば, ローカル接続の同ストレージのランダムリードアクセスのターンアラウンドタイムは 7.50ms である. また, 同文献で紹介されている独自の SCSI over IP 実装を用いての, リモートディスクに対するターンアラウンドタイムは片道遅延時間 0ms において 8.24 ms, 遅延 1ms において 10.09 ms, 遅延 2ms において 12.21 ms, 遅延 4ms において 16.29 ms, 遅延 8ms において 24.46 ms である. 同例のように, ストレージの動作を考慮したターンアラウンドタイムは “遅延時間 0 におけるターンアラウンドタイム+2×片道遅延時時間” でモデル化することが可能であり, ストレージの動作時間は約 8ms である.

次に, 性能に影響を与える各要素の時間規模を以下にまとめる. まず, ストレージデバイスの応答時間は 10ms 弱であり, 製品に依存する. 遅延確認応答の保留時間 数 10ms から数 100ms 程度であり, TCP/IP 実装に依存する (これは 500ms 以下でならないとされている [16]). ネットワークの片道遅延時間は, LAN などにおいて 1ms 未満, 国内のネットワークにおい

て 10ms 以上程度, 国際回線において 100ms 以上程度と予想される.

よって, ネットワークが 1ms 未満の状況においてはストレージデバイスの応答時間が全体の性能に対して支配的である (往復回数の削減の影響は小さい) と考えられ, この場合はストレージデバイスの高速化やキャッシュ等によるとストレージ読み込みの回避などの最適化を施すことが好ましいと思われる. また, 遅延確認応答の動作はその他の要素の動作時間 (10ms 未満) と比べて著しく大きいためこれの回避は重要であると思われる. 次に, ネットワークの遅延時間が 16ms 以上の様な環境においては, ネットワークの往復回数などが性能決定の主要因となり本稿で述べたような考察が性能に大きく影響を与えると期待される. 本稿で述べたネットワーク遅延が 4ms 程度である環境は, 上記の中間に位置し (ストレージデバイスの動作が 8ms 程度であり, ネットワークの往復時間も同様に 8ms) , TCP/IP の振る舞いを考慮した最適化は重要な要素の一つと言える. 遅延確認応答の動作はその他の要素と比べ小さくなく, これの回避も重要であると考えられる.

ただし, ストレージデバイスの高速化と TCP/IP を考慮した iSCSI ドライバの実装は独立して行うことが可能であるため両立させることが望ましい.

5. ま と め

本稿では, ネットワーク遅延の大きい環境におけるショートブロックサイズ iSCSI アクセスの性能について述べた. シーケンシャルアクセスと異なり, ショートブロックサイズのアクセスにおいてはそのターンアラウンドタイム性能が重要となる. 一般にネットワークが物理的にもつラウンドトリップタイムよりもターンアラウンドタイムを短くすることができず, ラウンドトリップタイムに近づけることが理想と言える. しかし, TCP の振る舞いを考慮せずに iSCSI ドライバの実装を行うとネットワークの往復回数を増やすことや, 遅延確認応答の動作によるターンアラウンドタイム性能の著しい低下を招くことがあり, TCP 実装の動作に対する考察が重要であると言える.

本稿の例に置いては, Nagle のアルゴリズムを有効にした状態において Data-in PDU と Response PDU の送出要求を個別に TCP に対して行うと, TCP 実装が Response PDU の送出を保留してしまい, ネットワーク往復回数が 2 回となり, ターンアラウンドタイムが約 2 倍となった. また, Nagle のアルゴリズムが有効になっている状態において, 微少 (MSS 未満のサイズ) な `read()` を行うと, 孤立した TCP パケット の受信を発生させ, それによる 遅延確認応答の動作を招くこととなりターンアラウンドタイムを著しく増加させてしまうことが確認された.

このように, SCSI プロトコルを TCP プロトコルの中にカプセル化して送信する iSCSI においては, その性能向上のためには TCP プロトコルの振る舞いを十分に考慮することが重要であると言える.

今後は, 実ハードディスクデバイスを用いての性能の考察, TOE(TCP Offload Engine) を用いての性能の考察などを進めていく予定である.

文 献

[1] 喜連川優. “ストレージネットワーキング”. オーム社出版局, July July 2002.

[2] Julian Satran et al. “iSCSI”. `http://www.ietf.org/internet-drafts/draft-ietf-ips-iscsi-20.txt` , January 2003.

[3] “IETF IPS”. `http://www.ietf.org/html.charters/ips-charter.html` .

[4] Storage Networking Industry Assocation. `http://www.snia.org/` .

[5] SNIA-J. `http://www.snia-j.org/` .

[6] 山口実靖, 小口正人, and 喜連川優. “高遅延広帯域ネットワーク環境下における iSCSI プロトコルを用いたシーケンシャルストレージアクセスの性能評価ならびにその性能向上手法に関する考察”. In 電子情報通信学会第 *14* 回データ工学ワークショップ, March 2003.

[7] 山口実靖, 小口正人, and 喜連川優. “バースト性を考慮した高遅延ネットワーク環境下における iSCSI シーケンシャルアクセスの性能向上に関する考察”. In 夏のワークショップ *DBWS 2003* 電子情報通信学会技術研究報告データ工学 信学技報 *Vol.103 No.190*, July 2003.

[8] 山口実靖, 小口正人, and 喜連川優. “iSCSI 解析システムの構築と高遅延環境におけるシーケンシャルアクセスの性能向上に関する考察”. 電子情報通信学会論文誌 *D-1*, 87, February 2004.

[9] John Nagle. RFC 896 : “Congestion Control in IP/TCP Internetworks”. `http://www.ietf.org/rfc/rfc0896.txt` , January 1984.

[10] R. Braden. RFC 1122 : “Requirements for Internet Hosts”. `http://www.ietf.org/rfc/rfc01122.txt` , October 1989.

[11] Wee Teck Ng et al. “Performance Evaluation and Improving of Sequential Storage Access using iSCSI Protocol in Long-delayed High throughtput Network”. In *Proc. of IEICE The 14th Data Engineering Workshop*, March 2003.

[12] F. Neema and D. Waid. “Data Storage Trend”. In *UNIX Review, 17(7)*, June 1999.

[13] EMC. “情報資源の活用新方程式”, 2003. p. 4.

[14] “北米の SAN 市場最新情報とシスコの戦略”. NETWORKERS 2003 Breakout Session 113, October 2003.

[15] IETF Home Page. `http://www.ietf.org/` .

[16] W. Richard Stevens. *“TCP/IP Illustrated, Volume 1:”*. Addison-Wesley, 1994.

[17] Prasenjit Sarkar and Kaladhar Voruganti. “IP Storage: The Challenge Ahead”. In *Proc. of Tenth NASA Goddard Conference on Mass Storage Systems and Technologies*, April 2002.

[18] Prasenjit Sarkar, Sandeep Uttamchandani, and Kaladhar Voruganti. “Storage over IP: When Does Hardware Support help?”. In *Proc. FAST 2003, USENIX Conference on File and Storage Technologies*, March 2003.

[19] L. Rizzo. dummynet. `http://info.iet.unipi.it/ luigi/ip_dummynet/` .

[20] University of new hampshire interoperability lab. `http://www.iol.unh.edu/` .

[21] iSCSI reference implementation. `http://www.iol.unh.edu/consortiums/iscsi/downloads.html` .