社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

iSCSI ストレージアクセスのトレースシステム

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

あらまし　本稿では iSCSI を用いた IP-SAN のアクセストレースシステムの提案と, それを用いた性能向上に関する考察について述べる. FC-SAN の欠点を補う SAN として TCP/IP と Ethernet を用いる IP-SAN や iSCSI が期待を集めるようになっているが, IP-SAN の欠点としては性能が FC-SAN より劣るとの指摘もあり IP-SAN の性能向上の実現が重要であると考えられている. iSCSI を用いた IP-SAN システムではサーバ計算機とストレージ機器が独立に個別の OS 等で稼働し, それらが TCP/IP ネットワークを用いて通信を行い協調して動作し全体のシステムを構築することとなる. よって, これらの統合的な解析の実現が重要である. 本稿では, 我々の実装した統合的なトレースシステムについて説明を行い, それを実際に高遅延環境における並列 iSCSI アクセスに適用し性能制限原因の発見および発見された問題の解決により並列 iSCSI アクセスの性能を向上させられることを示す.

Trace System of iSCSI Storage Access

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

† Center for Conceptual Processing of Multi-Media Informaition, IIS 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

Abstract　In this paper, we propose an IP-SAN access trace system and present performance improvement using the system. IP-SAN and iSCSI are expected to remedy problems of FC-based SAN. In IP-SAN systems using iSCSI, servers and storages work cooperatively by communicating with each other via TCP/IP, thus integrated analysis of servers and storages can be considered important. We explain our integrated trace system and show that the system can point out the cause of performance degradation.

1. はじめに

計算機システムの運用の大きな問題点の 1 個として, ストレージの管理費用の大きさが指摘されている. ストレージの運用には定期的なバックアップ等の作業が必要となり大きな管理費用が必要となる. この問題の解決策として SAN (Storage Area Network) の導入が提案された. SAN はストレージ専用の高速ネットワークであり, 1ヶ所で管理されているストレージに対して各計算機から SAN を用いて接続しストレージを使用する. ストレージを計算機の周辺機器として個別に管理するのではなく, SAN を用いてストレージを 1 箇所に集約することによりその管理費用は大幅に削減されることが期待される. この効果は高く評価をされており, 現在多くの企業において SAN が導入されている. しかし現在の FC (Fibre Channel) を用いる FC-SAN は普及に伴い, 以下のような欠点も明らかとなってきた. すなわち, (1)FC の管理技術を持つ技術者が少ない, (2)FC の導入費用は高い, (3)FC は接続距離に限界がある, (4)FC は相互接続性が必ずしも高くない, などの問題点も指摘されるようになり, これらの問題点を解決する SAN として TCP/IP と Ethernet を用いて構築する IP-SAN や, その標準的データ転送プロトコルである iSCSI に期待が集まるようになってきている. iSCSI を用いた IP-SAN では SCSI プロトコルを TCP/IP の中にカプセル化し Ethernet を用いて SCSI アクセスを転送する. よって, (1)TC/IP や Ethernet の管理技術を持つ技術者が多い, (2) 導入費用が低い, (3) 接続距離に限界がない, (4) 相互接続性が高い, などの利点が期待されている. 逆に iSCSI や IP-SAN の欠点としては, FC-SAN と比べて性能が劣ることなどが指摘されており, この問題の解決が最も重要であると言える. 特に, ネットワーク遅延による性能の劣化の問題が指摘されており [1] 本研究では高遅延環境における性能について考察をする.

iSCSI を用いた IP-SAN ではプロトコルスタックが SCSI

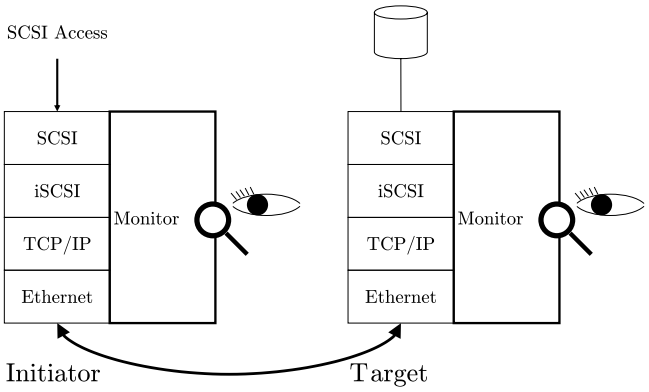

図 1　iSCSI プロトコルスタックと解析システム

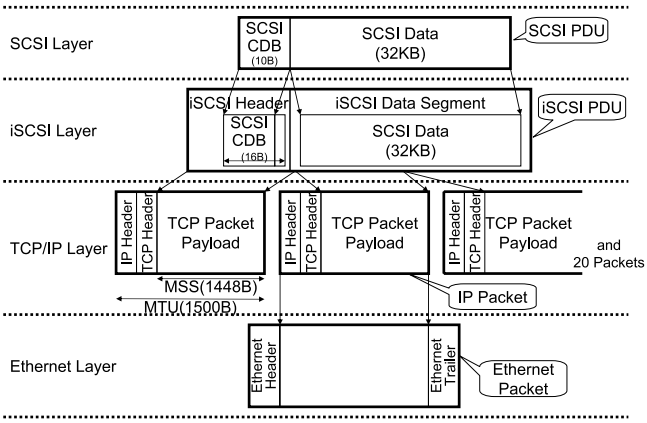

図 2　iSCSI プロトコルスタック各層の動作例

over iSCSI over TCP/IP over Ethernet という複雑な階層構造をとるが, End-to-End のストレージアクセスはこれら全層を通過してが行われるため, IP-SAN の性能を向上させるためにはこれら全層の振る舞いの把握が重要であると考えられる. さらに, サーバ計算機とストレージ機器が独立したシステムとして個別に動作しているが iSCSI アクセスは双方のプロトコルスタックを合成して構築されているためこれらの一方のみの解析ではシステム全体の振る舞いを把握することは困難であり, 双方を統合的に解析するシステムの実現が重要であると考えられる. 本稿では iSCSI ストレージアクセスを構成するこれら全層の解析が可能であり, サーバ計算機, ストレージ機器のトレース結果を統合的に処理することを可能とするトレースシステムを提案する. そしてそれを高遅延環境における並列 iSCSI アクセスに対し実際に適用し性能劣化原因の発見と性能向上が可能であることを示す.

本稿は, 以下のように構成されている. 第 2. 章において本稿で考察を行う iSCSI や IP-SAN の紹介を行う. そして, 第 3. 章において関連する研究の紹介を行う. 次に, 第 4. 章において本稿で提案する IP-SAN のトレースシステムの紹介を行い, 第 5. 章において提案したトレースシステムを並列 iSCSI アクセスに対し適用し当システムにより性能劣化原因の発見および性能の向上が可能であることを示す. 最後に, 第 6. 章において本稿のまとめを述べる.

2. iSCSI

2.1 IP-SAN と iSCSI

iSCSI は IP-SAN を主な応用として開発されたブロックレベルのデータ転送プロトコルであり, 2003 年 2 月に IETF [2] により標準化された [3]. iSCSI 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 TCP/IP ネットワークで接続された遠隔ストレージ機器に対する SCSI アクアセスを実現する. プロトコルスタックは図 1 の様に, SCSI over iSCSI over TCP/IP over Ethernet となる (同図内の “Monitor” に関しては, 第 4. 章において後述する). すなわち, サーバ計算機においてアプリケーションから発行された I/O 要求はサーバ計算機内において, ファイルシステムやブロックデバイスやキャラクタデバイス, SCSI 層, iSCSI 層, TCP/IP 層, Ethernet 層を経由してネットワークに送信され, ストレージ機器側では, Ethernet 層, TCP/IP 層, iSCSI 層, SCSI 層を経由し HDD デバイスにアクセスされることとなる. そして, これらの経路を逆方向に経由しサーバ計算機のアプリケーションに I/O 要求に対する応答が返されることとなる.

ストレージ機器 (ターゲット) における各層の振る舞いの様子を図 2 に示す. 同図は 32KB の “Data-in” iSCSI PDU の転送における例であり, MTU (Maximum Transfer Unit) 等は我々の実験環境における例である. SCSI 層で作成された SCSI データは iSCSI 層に渡され iSCSI PDU (Protocol Data Unit) の中にカプセル化される. iSCSI PDU は TCP/IP 層に渡され, TCP/IP 層はこれをセグメントサイズ毎に分割し各セグメント TCP/IP ヘッダを付加しこれらを Ethernet 層に渡す. これらを受け取った Ethernet 層が各パケットに Ethernet ヘッダとトレイラを付加し実際にネットワークで転送を行う. このように, iSCSI ストレージアクセスではこれら全層が動作し実行される. またこれら各層は必ずしも iSCSI における使用を想定して設計されていないため各層が独自に持つ機能が性能を低下させることもあり詳細な解析及び最適化が重要であると考えられる. 例えば, 我々は TCP の Nagle のアルゴリズム, 遅延確認応答が iSCSI 性能を大きく低下させること [4] や, TCP に実装されている輻輳制御機能と SCSI 層が独自に行うた同期処理の組み合わせにより性能が低下すること [5], [6] などを報告している.

3. 関 連 研 究

iSCSI を用いた IP-SAN の性能の評価に関する研究としては, 文献 [1], [7]〜[9] が上げられる. 文献 [1] は早期に SCSI over IP の性能評価を行った開拓的な研究である. 独自の SCSI over IP 実装を用い遅延のある環境におけるシーケンシャ/ランダムアクセスなどの基本性能の評価や, アプリケーション性能の評価を行っている. また, ファイルシステム等が性能に与える影響などについても考察を行っている. Sarkar らは文献 [7], [8] において低遅延環境における iSCSI アクセスの性能についての評価を行っている. 特に iSCSI 使用時における CPU 使用率の上昇と TOE(TCP Offload Engine) の使用について詳細な評価を

行っており, iSCSI 使用における CPU 使用率向上の程度の評価と現世代の TOE の貢献の限界を指摘している. 文献[9] は, IP 接続ストレージのアクセス手法として iSCSI と NFS の比較を行っている. 基本性能や応用性能の測定を行い iSCSI, NFS 双方の性能や CPU 使用率の評価を行っている. また, キャッシュの状態の性能への影響なども評価を行っている. 以上の研究は各種状況における iSCSI 性能の評価を行ったものであり, iSCSI の性能を知る上で有用な研究であると言えるがシステムの外部から負荷を与え性能を評価したものでありシステムの振る舞いについて考察を行ったものではない. よって, 性能が高い理由や低い理由, 性能の向上方法について十分な考察を行ったものではなく, 性能向上方法の提供を目指す本研究とは目的が異なる.

藤田らの文献[10] は, iSCSI ターゲットの内部の実装手法も考慮して iSCSI 性能の評価を行い, OS のカーネルに変更を加える手法や低レベルインターフェイスを使用する実装手法が性能において優れていることを指摘してる. ターゲット実装に着目し詳細な考察を行っている点において, システム全体の考察を目指す我々の研究と目的が同じでは無いが, ターゲットシステム実装の詳細な考察を行った既存の研究として価値が高いと思われる.

4. IP-SAN トレースシステム

本章では本稿で提案する “iSCSI ストレージアクセスのトレースシステム” について説明を行う. 前述のように iSCSI ストレージアクセスは個別に動作するサーバ計算機とストレージ機器が協調して実行される. サーバ計算機のアプリケーションから発行された I/O 命令はサーバ計算機において, ファイルシステム層やブロックデバイス層やキャラクタデバイス層, SCSI 層, iSCSI 層, TCP/IP 層, Ethernet 層を経由し, ストレージ機器において, Ethernet 層, TCP/IP 層, iSCSI 層, SCSI 層を経由し HDD デバイスにアクセスを行う. そして, この経路を逆方向に経由しアプリケーションに応答が返されることになる. そこで図 1 の様に iSCSI プロトコルスタック全層の振る舞いを観察できるモニタシステムを構築し, サーバ計算機とストレージ機器において個別に記録された iSCSI ストレージアクセスのトレース記録を統合的に解析することを可能とするトレースシステムを構築した. 実装には, オープンソースである Linux OS(ver. 2.4.18) とニューハンプシャー大学 InterOperability Lab[11] が提供する iSCSI reference implementation (ver. 1.5.02)[12] を用い (以下この iSCSI 実装を “UNH” と呼ぶ), これらのソースコードにモニタ用コードを適用することにより解析システムを実現させた.

図 3 に, 当システムを用いて iSCSI シーケンシャルアクセスのトレース結果を可視化した例を示す. 同図の縦軸は iSCSI アクセスのプロトコルスタックの遷移を表している. すなわち, 上から順に, (1) アプリケーションによるシステムコールの発行, (2)raw デバイス層, (3)SCSI 層, (4)iSCSI 層, (5)TCP/IP 層, (6)Ethernet によるパケットの転送, (7)TCP/IP 層, (8)iSCSI 層, (9)SCSI 層, (10)HDD デバイスアクセス, を表している.

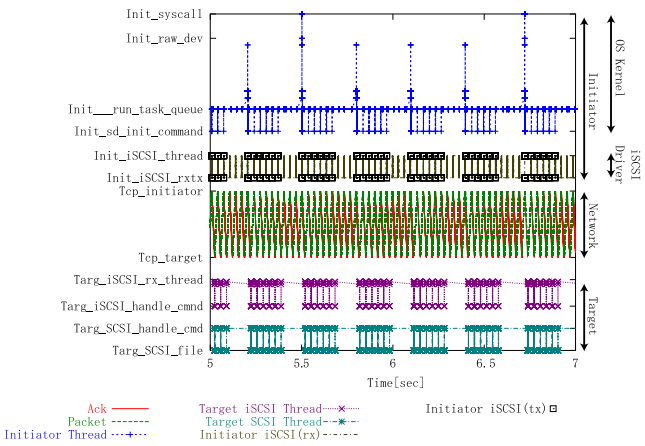

図 3 iSCSI アクセスのトレース

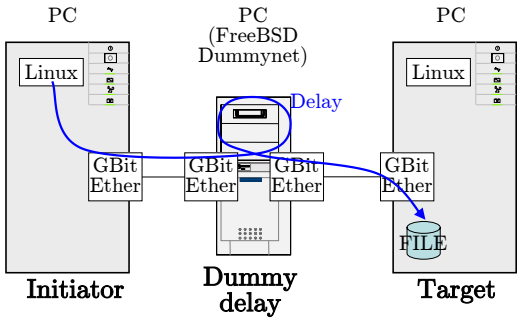

図 4 実験環境

(1)〜(5) が, サーバ計算機内における処理であり, (7)〜(10) がストレージ機器における処理である. 本トレース例では, ファイルシステムを用いずに raw デバイスを用いた. また使用した iSCSI Target は “ファイルモード” で動作させたため最下位層の HDD デバイスアクセスは実際はファイルアクセスの実行のトレースとなっている. 横軸が時間の経過を表しており, iSCSI ストレージアクセスの各処理における消費時間等を視覚的に確認することが可能となる. また, 大きなブロックサイズのシステムコールが各層で細分化されている様子や, 待ち状態にある処理の把握なども可能となる. 同図の例では 2MB のシステムコールが発行されており, これを raw デバイスが 512KB ごとの 4 要求に分割し, 4 要求が完了した時点で上位層にシステムコールの完了を通知していることや, raw デバイスから発行された 512KB の要求が 32KB の SCSI 命令に分割されていること, 細分化された要求を用いてネットワークに処理要求が送出されている様などを観察することが可能である.

5. 高遅延環境における並列アクセスのトレース

本章では, 前章で紹介を行った “iSCSI ストレージアクセスのトレースシステム” を実際に並列 iSCSI アクアセスに対し適用し, その有効性を示す. 具体的には高遅延環境において iSCSI 接続のストレージに対し複数プロセスから並列にショートブロック I/O 要求を発行し, 並列性を制限している層の発見が可能であることを示す.

5.1 並列アクセス実験

図 4 のように iSCSI イニシエータ (サーバ計算機) と iSCSI

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

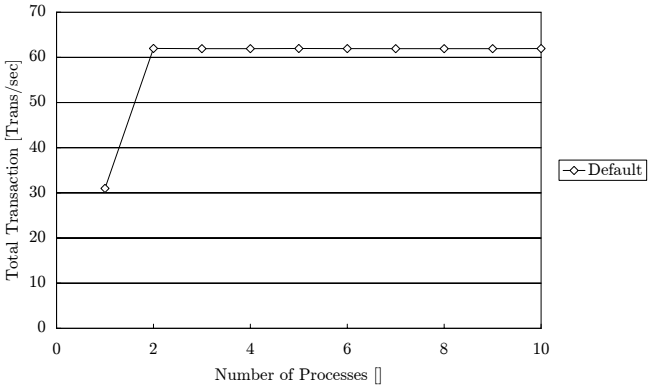

図 5　実験結果 A：初期状態における並列 I/O の合計性能

ターゲット (ストレージ機器) を Gigabit Ethernet で接続し IP-SAN 環境を構築し性能測定実験を行った．イニシエータとターゲットの間には人工的な遅延装置として FreeBSD Dummynet [13] を配置し擬似的な高遅延環境を実現した．iSCSI 実装としては，UNH 実装を用いた．イニシエータ，ターゲット，遅延装置はすべて PC 上に構築し，イニシエータとターゲットには Linux を，遅延装置 には FreeBSD をインストールした．イニシエータ，ターゲット PC の詳細は表 1 の通りであり，遅延装置の PC の詳細は表 2 の通りである．

本実験環境において以下ような並列ショートブロックアクセス実験を行った．サーバ計算機からストレージ機器に対して iSCSI 接続を確立し，その raw デバイスに対してシーケンシャルにショートブロックアクセス (raw デバイスに対してシステムコール “`read()`”) を行うベンチマークプログラムを作成し，同ベンチマークプログラムを複数プロセス同時に起動し全プロセスの合計性能を測定した．ブロックサイズは 512 バイトとし，片道遅延時間は 16m 秒とした．

5.2　実験結果

解析実行前の初期状態において上記の実験を行い図 5 の結果を得た．横軸が並列に動作させたベンチマークプロセスの数である．縦軸は合計性能を表し，単位時間における全プロセスの合計トランザクション数である (トランザクション数は “512 バイトのシステムコールの回数” を表す)．シングルプロセス時の性能は 31.0 [Trans/sec] であり，これは往復遅延時間 0.032 秒の逆数とほぼ等しく期待通りの性能が得られている．同様に，2 プロセス並列時の性能も 62.0 [Trans/sec] となり 2 プロセス合計でほぼ 2 倍の性能が得られていることが確認された．高遅延ショートブロックアクセス時にはネットワーク資源，CPU 資源ともに枯渇しないため，期待通りの性能が得られたと言える．これらに対し並列プロセス数を 3 以上に向上させても合計性能は 2 並列時と同程度となり，プロセス数の上昇に対する合計性能の向上は 2 プロセスで飽和する結果となった．これは，多段プロトコルスタックで構成される iSCSI ストレージアクセスのいずれかの層で並列動作数を 2 に制限していることが原因と考えられる．

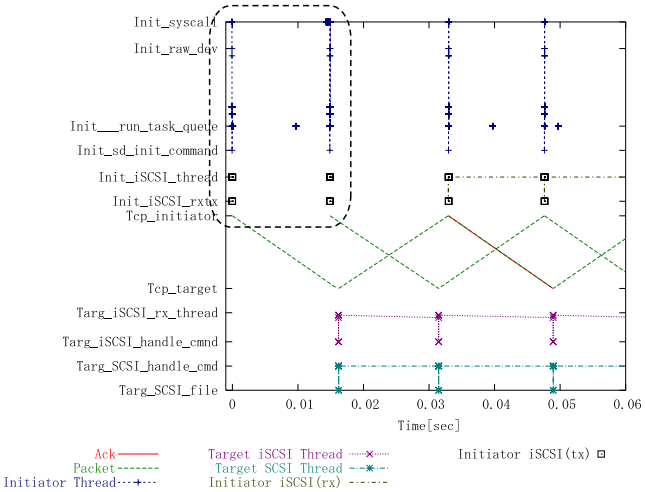

図 6　並列アクセスのトレース図 (default)：A

5.3　並列アクセスのトレース

前節の 3 プロセス並列アクセス実験のトレースを提案トレースシステムを用いて可視化すると図 6 の様になった．同図において “Initiator Thread” の線が切断されて表記してあるのは，スケジューラによるコンテキストスイッチが発生し同プロセスが一旦停止し処理が連続していないからである．同様に 0.010 秒，0.040 秒などにおいてトレースがプロットされているのもコンテキストスイッチが発生し同プロセスに CPU が割り与えられたが資源待ち状態であるためその直後に再度コンテキストスイッチを発生させ同プロセスが CPU 資源を解放し再度停止したことを表している．同図よりイニシエータ (サーバ計算機) からターゲット (ストレージ機器) に対し 1 往復時間に内に同時に 2 個の I/O 要求しか送出されていないことが確認できる．すなわち，並列度を制限している層はサーバ計算機側に存在すると予想される．次に，トレース結果をもとに，①イニシエータの SCSI 層において発行された SCSI Read 命令のアドレス，②各往復時間 (32ms) 内に発行されたシステムコール数，raw デバイスからの要求数，SCSI 層における SCSI 命令の数，③同時に処理中 (すでに要求されたが終了していない) にあるシステムコールの数，raw デバイスにおける要求の数，の時間変化を可視化し図 7 を得た．①は，3 プロセスそれぞれ I/O(A)，I/O(B)，

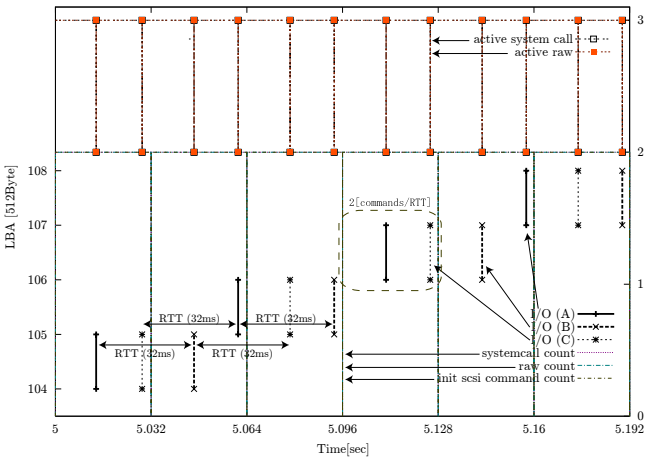

図 7　SCSI 命令, 並列発行要求数の推移 (default)

I/O(C) で記されており 1 往復時間内に 2 個の SCSI 命令しか発行されていないこと, ある命令発行の 1 往復時間後にその処理が終了し次の命令が発行されていることなどが確認できる. ②は “system call count”, “raw count”, “init SCSI command count” で表されておりそれぞれ各往復時間内において発行されたシステムコール数, raw デバイスからの要求数, SCSI 命令の発行数である. これらからも 1 往復時間内において各処理は 2 個ずつしか実行されていないことが確認できる. それに対し, ③処理中のシステムコール, raw デバイスによる要求の数 (同図における “active system call”, “active raw”) は常に 3 となっている. 図より処理途中のシステムコール, raw デバイス要求数は処理が終了した瞬間 (同図の例においては 5.014 秒, 5.029 秒など) に 2 に減少しその直後に次の要求が発行され処理途中要求数は再度 3 に上昇しているのが確認できる (ただし, “active system call” と “active raw” はほぼ同時刻に変化するため同図内では両線は重なって表示されている). 以上より, 常に 3 個のシステムコールが待ち状態にあるが, 1 往復時間で 2 要求ずつしか処理されていないことがトレースの解析により確認された.

次に並列度を制限している部分の巨視的な解析を行う. 図 6 左上波線部を拡大し可視化したものを 図 8 の左上に記す. そして, その波線部の拡大図を同図右下に記す. 同図では, 並列して動作している 3 本の I/O 要求を “I/O(A)”, “I/O(B)”, “I/O(C)” と別の線として記した. また, 図 6 同様にコンテキストスイッチが発生しプロセスが停止した時点でトレースの線は切断して記した.

図 8 左上より I/O(A), (B), (C) は順に時刻 0.000 秒, 0.015 秒, 0.015 秒にシステムコールを発行していることが確認できる. このことから, ターゲットからの応答を待たずに 1 往復時間 (32ms) 内に 3 個のシステムコールが発行されておりシステムコールの発行において並列性が制限されていないことが分かる. また, “I/O(A)” のトレースより “I/O(A)” の要求は raw デバイス層, SCSI 層, iSCSI 層, TCP/IP 層を経由し要求が実際にネットワークから送出されていることも確認できる. 次に, 同図右下より “I/O(C)” のシステムコールは時刻 0.01485 秒に発行され, 各層を経由し要求はネットワークに送出されている

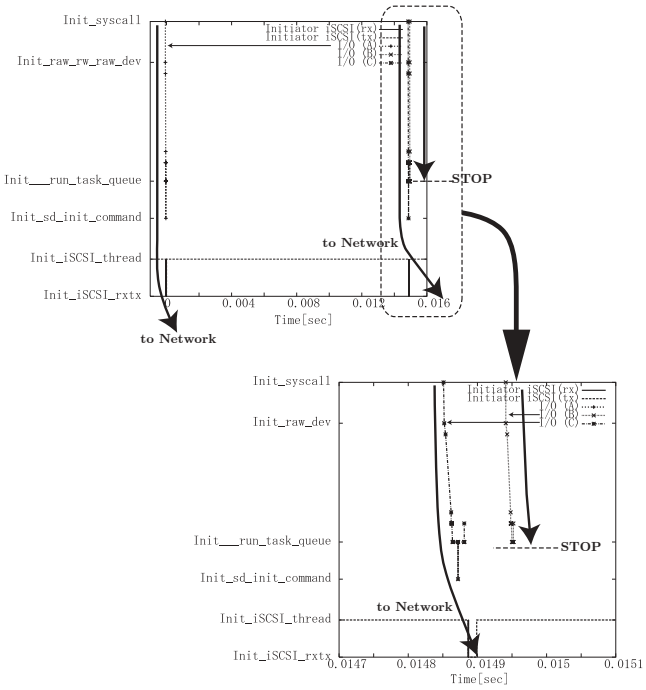

図 8　並列アクセスのトレース図 (default) : B

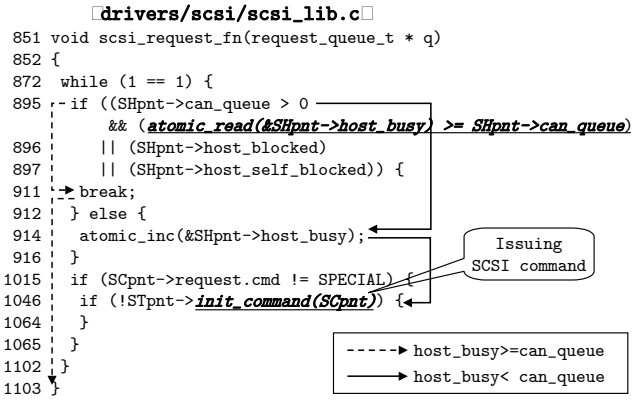

図 9　Linux SCSI 層のトレース : ”drivers/scsi/scsi_lib.c”

ことが確認できる. これに対し “I/O(B)” のシステムコールは時刻 0.01494 秒に発行され, raw デバイス層を経由し raw デバイスによる命令は発行されているが SCSI 層における SCSI 命令の発行に到っていないことが確認でき, SCSI 命令の同時発行上限が 2 となっていることが確認できる.

次に, 並列度制限箇所の微視的な解析を行う. SCSI 命令が即時に発行される最初の 2 要求 (I/O(A), I/O(C)) と命令の発行が拒否される 3 個目の要求 (I/O(B)) の分岐点は SCSI 層である図 9 の Linux カーネルの “`drivers/scsi/scsi_lib.c`” 部である. 同実装箇所は現在アクティブである命令数 `host_busy`(注1)と, 下位層 (本例では iSCSI ドライバ) が同時に受け付け可能なアクティブな SCSI 命令数 `can_queue`(注2)の比較部となっている.

(注1)：Linux “`drivers/scsi/hosts.h`” にて “commands actually active on low-level” と解説されている
(注2)：UNH iSCSI 実装の “`initiator/iscsi_initiator.c`” にて “max no. of simultaneously active SCSI commands driver can accept” と解説されている

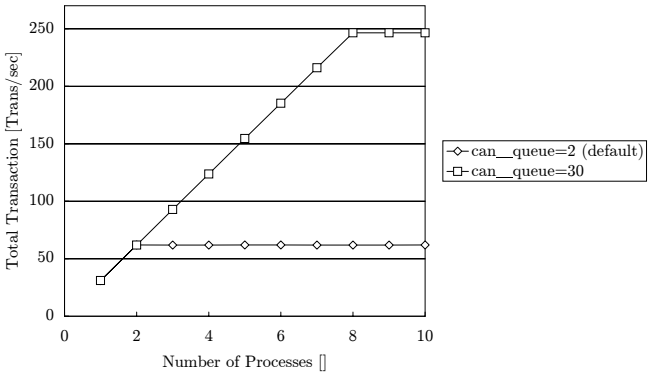

図 10　実験結果 B：並列 I/O の合計性能

UNH iSCSI 実装において “can_queue” の初期値は 2 となっており, アクティブ命令数 host_busy は 0 から開始される. 最初の 2 要求 (I/O(A),I/O(C)) のトレースでは host_busy はそれぞれ 0, 1 となっており図 9 における “host_busy<can_queue” に示される動作が記録された. すなわち, 同図 914 行目において host_busy をインクリメントし, 1046 行目において SCSI 命令が発行される動作が記録された. 3 個目の要求 (I/O(B)) においては host_busy が 2 であり “host_busy>=can_queue” に示される様に SCSI 命令が発行されない動作が記録された. 以上の微視的なトレース解析により, 並列度を制限し性能向上を妨げている原因は iSCSI ドライバにおける “can_queue” の値が不十分であることだと考えられる.

5.4　発見された問題点の回避と性能の向上

前節の解析から高遅延環境における並列アクセスの性能を向上させるには “can_queue” の値を増加させることが重要であると考えられる. これを 30 に増加させ第 5.1 節の実験を行い図 10 の実験結果を得た. 同図より, 全プロセスの合計性能は並列度 8 までは線形に増加させることが可能となり, 並列度 8 以上においては 4 倍の性能向上が実現させられた.

このように, 提案トレースシステムを用いてサーバ計算機上のアプリケーションからストレージ機器の HDD デバイスのアクセスまでの統合的なアクセストレース解析を行うことにより, iSCSI ストレージアクセスの性能劣化原因となっている点を的確に発見することが可能であり, 発見された問題の回避により性能を大きく向上させることが可能であることが確認された.

6. おわりに

本稿では, サーバ計算機とストレージ機器の分散協調システムとして動作する IP-SAN システムのアクセストレースシステムを提案し, その有効性の検証を行った. その結果, 提案システムは多段プロトコルスタックの中から性能劣化原因を的確に発見することが可能であり, その回避により多並列アクセス時に 4 倍の性能向上が実現され, 提案手法が IP-SAN システムの性能向上の実現に有効な手法であることが確認された.

本稿では解析対象としてファイルシステムやブロックデバイスを用いない例を選択し, ストレージ機器の HDD デバイスとしもてファイルモードを採用した. しかし, ファイルシステムやブロックデバイスのキャッシュヒットによるネットワークより上位層における要求の終了など性能に大きな影響を与える振る舞いの把握も重要であると考えられる. また, ショートブロックアクセス時には HDD デバイスアクセス時間の考察も重要であると考えられる. 今後はより実応用に近いシステムへのトレース解析の適用を考え, ファイルシステムや実 HDD デバイスを用いたシステムの解析を行っていく. そして, トレースシステムの適用に起因するオーバーヘッドなどについても考察していく. また, 本稿では “can_queue” 値の増加の後に並列度を 8 に制限している原因について言及していないが, この問題の解析についても述べていく予定である.

文　献

[1] 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.

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

[3] J. Satran et al. “Internet Small Computer Systems Interface (iSCSI)”. http://www.ietf.org/rfc/rfc3720.txt , April 2004.

[4] 喜連川優 山口実靖, 小口正人. “iSCSI における小粒度アクセスの特性解析”. In 電子情報通信学会第 *15* 回データ工学ワークショップ, March 2004.

[5] 喜連川優 山口実靖, 小口正人. “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.

[6] 喜連川優 山口実靖, 小口正人. “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.

[7] 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.

[8] 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.

[9] Peter Radkov, Li Yin, Pawan Goyal, Prasenjit Sarkar, and Prashant Shenoy. “A performance Comparison of NFS and iSCSI for IP-Networked Storage”. In *Proc. FAST 2004, USENIX Conference on File and Storage Technologies*, March 2004.

[10] 藤田智成 小河原成哲. “iSCSI ターゲットソフトウェアの解析”. In 先進的計算基盤システムシンポジウム *SACSIS 2004*, May 2004.

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

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

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