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アクセス時のTCP輻輳ウィンドウ制御を用いたシステム性能向上手法の一検討

豊田真智子† 山口 実靖†† 小口 正人†

† お茶の水女子大学 人間文化研究科 〒112–8610 東京都文京区大塚 2–1–1
†† 東京大学生産技術研究所 〒153–8505 東京都目黒区駒場 4–6–1
E-mail: †{machiko,oguchi}@ogl.is.ocha.ac.jp, ††sane@tkl.iis.u-tokyo.ac.jp

あらまし 既存のネットワークインフラを利用し，低コストで構築可能な IP-SAN の技術として，Ethernet と TCP/IP を用いて構築する iSCSI が注目を集めている．iSCSI を用いて遠隔ストレージにアクセスする際には，スループットが重要な性能評価基準となるが，スループットは TCP 層のパラメータである輻輳ウィンドウと密接に関連している．本稿ではスループットのばらつきを抑制するために，輻輳ウィンドウを動的にコントロールする手法を提案する．iSCSI ネットワーク上で提案手法を用いてストレージアクセスを行うことで，スループットが安定し，効率のよい通信が行われていることを確認できた．
キーワード iSCSI，輻輳ウィンドウ，動的コントロール，遠隔ストレージ

A Study of Performance Improvement by Controlling TCP Congestion Window on iSCSI Access

Machiko TOYODA†, Saneyasu YAMAGUCHI††, and Masato OGUCHI†

† Humanities and Sciences, Ochanomizu University
Otsuka 2–1–1, Bunkyo-ku, Tokyo, 112–8610 Japan
†† Institute of Industrial Science, The University of Tokyo
Komaba 4–6–1, Meguro-ku, Tokyo, 153–8505 Japan
E-mail: †{machiko,oguchi}@ogl.is.ocha.ac.jp, ††sane@tkl.iis.u-tokyo.ac.jp

Abstract The iSCSI protorol based on TCP/IP and Ethernet is becoming increasingly important as IP-SAN technology, which is configured with low cost using existing network infrastructure. Throughput is an important performance indicator in accessing remote storage with the iSCSI, and throughput has a close relationship with the size of Congestion Window of TCP parameter. In this paper, we propose an idea of dynamic Congestion Window control nethod to balance throughput unevenness. Accessing remote storage on iSCSI communication using the proposed model, we confirmed that throughput become stable and efficient communication is achieved.
Key words iSCSI, Congestion Window, dynamic control, remote storage

1. はじめに

近年，計算機システムは，WWW システムなどの大量のデータを蓄積するアプリケーションの登場により，処理するデータ量が増大している．システム全体のデータ容量が増大すると，データのバックアップや，容量増加に対応するためのストレージ増設，障害対応などの運用にかかる負担が問題になる．これまでストレージは DAS(Direct Attached Storage) と呼ばれる方法で管理されてきた．これはストレージを直接接続されているサーバごとに管理するという技術である．しかしこの手法では，各々のストレージの使用に無駄があり，資源を有効に活用できないといった問題点が指摘されている．

そこで，複数のサーバがストレージ装置を共有する，スト

レージ統合が考えられた．代表的な技術である SAN(Storage Area Network) は，サーバとストレージ間を接続する高速のネットワークであり，現在 Ethernet と TCP/IP を用いて構築する IP-SAN が期待されている．IP-SAN の中で，高速な SAN 実現の基盤として最も重要な技術の 1 つと注目されている iSCSI は，IP ネットワークを介して SCSI コマンドを送るため，イントラネット経由でのデータ転送が容易になり，遠距離のストレージアクセスも可能となる．

iSCSI を用いて遠隔ストレージアクセスを行った場合，TCP/IP のみで構築されたネットワークと比較してスループットの低下が問題となる [1]．またスループットは，通信を行う上で重要な階層となる TCP 層のパラメータの 1 つである輻輳ウィンドウと密接に関係しており，輻輳ウィンドウの低下時にはスループットも低下し，効率の良い通信が行えていないのが現状である [2]．

そこで本稿では，Linux OS の内部で輻輳ウィンドウの値をモニタし，輻輳ウィンドウを動的にコントロールして通信を行う手法を提案する．これは，Target でモニタした輻輳ウィンドウが低下した際に，Initaitor へ低下を通知し，以降の Target へのアクセスのブロックサイズを変更するというものである．この手法を実装した Initiator，Target を使用して，iSCSI プロトコルで遠隔ストレージアクセスを行った．コントロール前にはある値まで増加すると急激に低下し，再度増加をするという規則的な鋸状の変化を繰り返していた輻輳ウィンドウが，コントロール後には最終的に一定の値をとるようになった．それに伴い，不安定に変化していたスループットも，輻輳ウィンドウが一定値をとるとほぼ一定値となり，安定した通信でストレージアクセスが行われている様子が確認された．さらに，2 種類の実験環境を用いることで，異なる環境においてもこの手法を適用できることが示された．

本稿は以下のように構成される．まず 2 章で研究背景を紹介する．3 章では輻輳ウィンドウのコントロール手法を概説する．4 章でコントロール手法を実装したマシンでの実験を行い，その結果について議論する．最後に 5 章でまとめを述べる．

2. 研究背景

2.1 iSCSI

2003 年 2 月に IETF により承認されたばかりの新しい技術である iSCSI は，SCSI コマンドをカプセル化して TCP/IP ネットワーク上に転送する [3] [4]．iSCSI では，ストレージ機能提供側を Target，ストレージ要求側を Initiator と呼ぶ．Initiator に対応する機器には，ネットワークインターフェイスと iSCSI 対応のデバイスドライバを用意するだけで，あたかも直接接続されているかのように遠隔地にあるストレージデバイスにアクセスすることができる．また，見かけ上従来の SCSI と変わらないため，ディスクに直接読み書きするといった，特殊なディスクアクセスにも対応できる．iSCSI の階層モデルを図 1 に示す．送信側では，SCSI 層からの SCSI コマンドやレスポンス，データをカプセル化処理して iSCSI PDU を作成し，TCP/IP が提供する TCP コネクションを介して送信を行う．受信側では，TCP コネクションを介して受信した iSCSI PDU を処理して SCSI コマンドやレスポンス，データを抜き出し，それを SCSI 層に引き渡す．

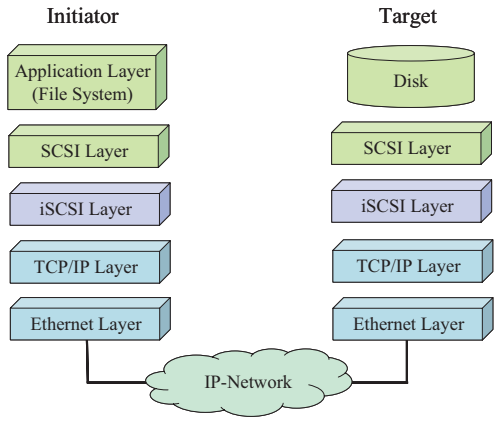

図 1　iSCSI プロトコルスタック

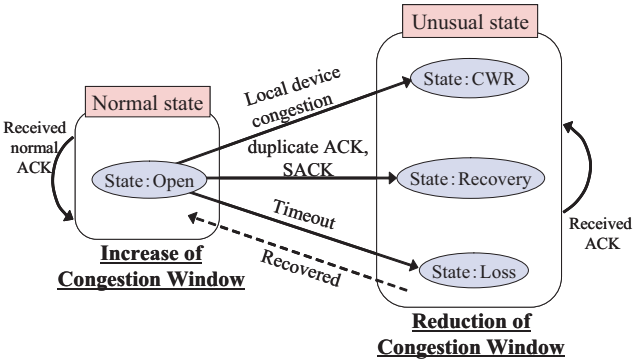

図 2　Linux TCP 状態機械遷移図

iSCSI についての研究も，徐々に行われ始めている [5] [6]．しかし，システム内部の TCP 層の振る舞いについて触れ，TCP パラメータ値がシステム性能に関係することを取り上げている研究は，まだ行われてはいない．

2.2 Linux TCP 実装

本研究においては，Linux OS の内部で TCP パラメータのモニタを行った．Linux OS の TCP 実装は，状態機械として実装されている．パケット受信時には，状態機械の状態によりその処理が異なる．その様子を図 2 に示す．

通信時の状態が正常であれば，ACK 受信ごとに輻輳ウィンドウは増加するが，エラーが検出されると異常と判断され，輻輳ウィンドウが低下する．輻輳ウィンドウが低下する原因としては，下位層であるデバイスドライバのバッファが溢れることによる，Local Congestion を検出した場合（CWR），重複 ACK，SACK を受信した場合（Recovery），

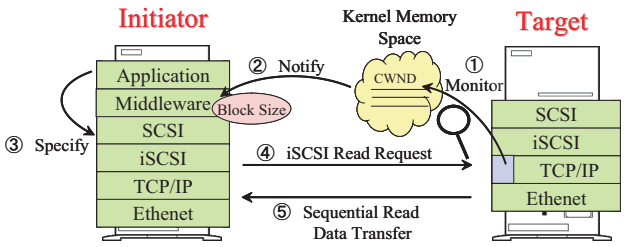

図 3　輻輳ウィンドウコントロール手法の概念図

タイムアウトを検出した場合（Loss）が挙げられる．また，Linux の TCP 実装は，通信中に一度設定された輻輳ウィンドウは，そのウィンドウの値を使い切らない限りは変化しないという特徴を持ち，この時スループットは，安定したほぼ一定の値をとる．しかし，輻輳ウィンドウが変化する場合はスループットも変化を示し，性能が不安定となる [2]．

3.　輻輳ウィンドウコントロール手法

本章では，本稿で提案する輻輳ウィンドウのコントロール手法とその概要を示す（図 3）．

本実験では，通信時に TCP 実装が行っているフロー制御で管理されている TCP パラメータをモニタし，可視化することでその状態を把握している．Linux の TCP のソースコード内のモニタしたい箇所にモニタ関数を挿入し，TCP パラメータを外部からアクセス可能なカーネルメモリ空間に記録するように実装を行い，カーネルを再構築する．これにより，Linux カーネル内部の TCP 実装で管理されている TCP パラメータを，カーネルメモリ空間にアクセスするための特殊ファイルを読み出すことにより確認が可能となる [7]．

この TCP パラメータモニタの仕組みを Target に実装し，Target から輻輳ウィンドウ変化の通知を受けて，アプリケーションによるストレージアクセスのブロックサイズを調節する機能をミドルウェアとして実装して，Initiator から Target へ iSCSI プロトコルを用いたシーケンシャルリードアクセスを行った．この時，TCP パラメータである輻輳ウィンドウを取り出し，その変化を観察した．本実験では，輻輳ウィンドウ低下原因として CWR，Loss が確認されているが，シーケンシャルリードアクセスにおいて，周期的な間隔で最も頻繁に検出されるのが CWR であることから，CWR を検出した場合のみ輻輳ウィンドウをコントロールするものとした．

上記記述に基づいた輻輳ウィンドウコントロール手法は以下の通りである．

（1）　Target で輻輳ウィンドウをモニタし，変化を観察する

（2）　CWR が検出され，輻輳ウィンドウが低下した場合，Initiator に輻輳ウィンドウの低下を通知する

（3）　通知を受けた Initiator では，ミドルウェアが調節を行い，アプリケーションがブロックサイズを再指定する

（4）　Initiator から Target にシーケンシャルリードコマンドが送信される

（5）　Target から Initiator に要求されたブロックサイズでデータが転送される

（6）　CWR を検出するたびにこの処理を繰り返す

この手法による輻輳ウィンドウコントロールにより，やがて輻輳ウィンドウが一定値となるブロックサイズに落ち着く．この時スループットもほぼ一定の値をとるため性能が安定し，効率の良いストレージアクセスを行うことができる．

表 1　使用計算機

CPU	Intel Xeon 2.4GHz
Main Memory	512MB DDR SDRAM
OS	Initiator, Target： Linux2.4.18-3 Dummynet ： FreeBSD 4.5 - RELEASE
NIC	machine1 : Intel PRO/1000MT Server Adapter Intel 82545 EM machine2 : Intel PRO/1000XT Server Adapter Intel 82544 EI

4.　性能測定実験

本章では，前章で述べた手法により輻輳ウィンドウがコントロールされ，スループットが安定する様子を評価するために，iSCSI ストレージアクセスの性能測定実験を行う．

4.1　実験環境

本実験は以下の環境で行った．Initiator と Target 間は Gigabit Ethernet で接続し，TCP/IP 接続を確立した．サーバとストレージ間が離れている場合のストレージアクセスを想定した実験を行うため，Ethernet の接続途中に人工的な遅延装置として FreeBSD Dummynet [8] を挟み，片道遅延時間を“ 2ms ”，往復“ 4ms ”に設定し，クロスケーブルで接続した．Initiator，Target，Dummynet はすべて PC 上に構築し，Initiator，Target には Linux を，Dummynet には FreeBSD をインストールした．実験で生じるストレージデバイスの影響をなくすために，Target はメモリモードで動作させ，ディスクアクセスを伴わないようにした．これにより，ストレージは無限に高速であり性能に影響を与えないとみなすことができる．実験で使用した計算機の環境を表 1 に示す．なお，実験では，NIC の異なる 2 組の Initiator，Target マシンで測定を行っており，Intel PRO/1000 MT Server Adapter の NIC を使用したマシンをマシン 1，Intel PRO/1000 XT Server Adapter の NIC を使用したマシンをマシン 2 とする．

また，本実験で用いた iSCSI 実装において，Target にはニューハンプシャー大学 InterOperability Lab が提供する UNH IOL reference implementation ver.3 on iSCSI Draft

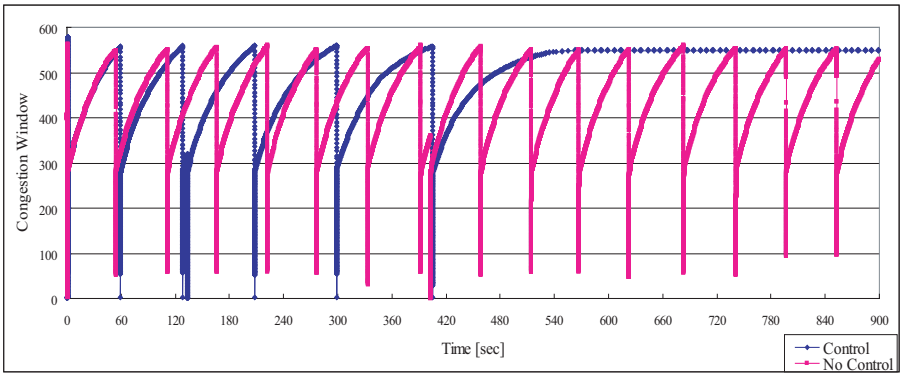

図 4　コントロールした場合としない場合における輻輳ウィンドウの比較

18 [9] を用いた．この UNH 実装では，大きなブロックサイズで read コマンドを発行しても，要求したブロックサイズより小さなブロックサイズに分割してデータを送信する [10]．本実験ではこの実装による性能測定への影響を避けるために，Initiator には UNH 実装を用いず，UNH 実装の Initiator と同等の機能を持ち，かつ大きなブロックサイズのデータ転送を行う自作 Initiator を用いて実験を行った．この自作 Initiator は通常のユーザ空間のアプリケーションとして動作し，iSCSI Target と TCP/IP コネクションを確立して iSCSI プロトコルで通信を行うものである．

本実験においては，Initiator から Target へシーケンシャルリードアクセスを行い，その時のスループット，輻輳ウィンドウの値の測定を行った．また，Target から Initiator に輻輳ウィンドウ低下通知があった場合に Initaitor で再指定するブロックサイズは，その時アクセスしていたブロックサイズより 50KB ずつ低下させる実装とした．本章で示す図は，すべて初期値として 1024KB のブロックサイズで実験を行った場合の結果とする．

4.2　マシン 1 における測定結果

前節の実験により得られた結果として，マシン 1 における測定結果を示す．まず，輻輳ウィンドウをコントロールした場合とコントロールしない場合の時間変化のグラフが図 4 である．同図より，輻輳ウィンドウをコントロールしない場合には，輻輳ウィンドウが増加しては低い値にリセットされるという鋸状の変化を繰り返す．これは，輻輳ウィンドウが増加していくと，ある時点でデバイスドライバのバッファ溢れによる Local Congestion エラーが起き，これを検出した Linux の TCP 実装が，輻輳ウィンドウの値を非常に小さな値にリセットするという動作が繰り返されているためである．これに対し輻輳ウィンドウをコントロールした場合には，低下するまでの時間が徐々に増加し，最終的に一定値となり，コントロールを実現できている様子が確認できる．

次に輻輳ウィンドウをコントロールしない場合とした場合のスループットと輻輳ウィンドウの変化を図 5，図 6 に示す．

輻輳ウィンドウをコントロールしない場合の図 5 においては，輻輳ウィンドウが低下を示す時，スループットも一時低下を示す様子が確認される．一方，輻輳ウィンドウをコントロールした場合の図 6 においては，輻輳ウィンドウが変化を示す時はスループットも変化を示すが，徐々に輻輳ウィンドウが低下する時間間隔が広がり，最終的に一定値となった時にはスループットも安定したほぼ一定値を取っていることがわかる．

4.3　マシン 2 における測定結果

前節と同様，マシン 2 において，輻輳ウィンドウをコントロールしない場合とコントロールした場合のスループットと輻輳ウィンドウの変化を図 7，図 8 に示す．

輻輳ウィンドウをコントロールしない図 7 においては，マシン 1 と同様，輻輳ウィンドウが低下を示すとき，スループットも一時低下する様子が確認される．しかし輻輳ウィンドウをコントロールした場合は，図 8 で確認されるように，マシン 1 と NIC が異なるマシン 2 においても，輻輳ウィンドウをコントロールしたストレージアクセスが実現された．マシン 1 とは異なり，マシン 2 ではコントロールした後のスループットがやや不安定であるが，これは NIC の性能の限界によるものと考えられる．マシン 2 より性能が良い NIC を使用したマシン 1 では，このような変化は見られない．

4.4　考　　察

実験結果より，マシン 1，マシン 2 のどちらの場合も，輻輳ウィンドウが低下すると一度に送信できるパケット数が減少するため，一時的にスループットが減少することがわかった．通常のストレージアクセスにおいても，輻輳ウィンドウは実験で確認された鋸型の変化を繰り返すため，通信中のスループットは安定せず，不安定なものとなっている．提案した実装を使用すると，iSCSI を用いたストレージアクセスの際の輻輳ウィンドウをコントロールし，スループットもほぼ一定に保ったままでストレージアクセスを行うことができる．また，異なる実験環境であっても，提案手法

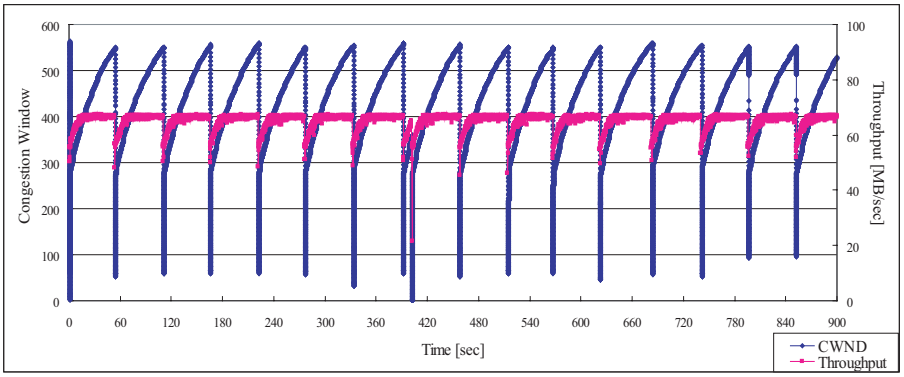

図 5　輻輳ウィンドウをコントロールしない場合のスループットと輻輳ウィンドウの時間変化：マシン 1

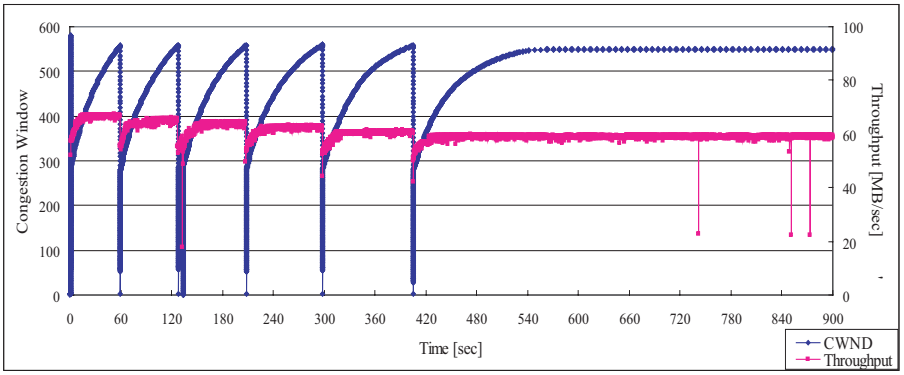

図 6　輻輳ウィンドウをコントロールした場合のスループットと輻輳ウィンドウの時間変化：マシン 1

は適用できることが確認できた．

本実験で用いたマシン 1，マシン 2 では NIC の相違から性能も異なり，輻輳ウィンドウが一定となる場合のブロックサイズの値が異なる．この値は，マシン 1 では 780KB であり，マシン 2 では 490KB であった．つまり，アプリケーションからこのブロックサイズ以下でリードリクエストを発行した場合に，輻輳ウィンドウが一定値となり，スループットも安定することになる．本実験ではブロックサイズの初期値を 1024KB として測定を行ったため，マシン 2 では，輻輳ウィンドウが一定値となるまでマシン 1 の倍以上の時間がかかってしまう．通常ストレージアクセスは，Initiator と Target 間で大きなブロックサイズで実行されることが予想される．そのため，現在のコントロール手法では，より大きなブロックサイズでアクセスすると輻輳ウィンドウが一定値となるまでに長い時間がかかる．より早く，Initiator から一定値となるブロックサイズでアクセスするために，Initiator が再指定するブロックサイズを決定するアルゴリズムを検討する必要がある．

iSCSI は，その構成の特徴から，遠隔地にストレージを配置した長距離接続の環境において，その効力を発揮するものであると期待されている．このような高遅延環境においては，パケットを送信してから ACK が返信されるまでの時間が長くなり，何もせずに待っているだけという無駄な時間が増えてしまう．そのため，一度に送信できるパケットを増やすことが待ち時間を短縮でき，効率よく通信を行うためには重要である．高遅延環境において，輻輳ウィンドウをコントロールしなかった場合，定期的に起こる Local Congestion エラーにより，輻輳ウィンドウの値は非常に小さくなり，その間の通信は少数のパケット送信後，長い遅延時間を経て ACK を受け取るまで送信側は待つだけの時間が続くため，スループットは劇的に低下する．本実験環境では，片道遅延が 2ms，往復でも 4ms という小さな遅延であるため，現状ではマシン 1，マシン 2 のどちらにおいても，ストレージアクセス開始直後のスループットの方が，コントロール後，輻輳ウィンドウが一定値となった時のスループットより良いという結果となっているが，上記のような理由から，高遅延環境においては，提案手法を用いた方がコントロールをせずにストレージアクセスを行った場合よりもスループットの向上が見込めると考えられる．

5. ま と め

本稿では，iSCSI ストレージアクセス時に確認される，輻輳ウィンドウの鋸状の変化とスループットの低下を抑えるため，輻輳ウィンドウをコントロールする手法を提案した．輻輳ウィンドウは，Target の Linux TCP ソースコード内に独自の関数を挿入することでカーネル外部で読み出し可

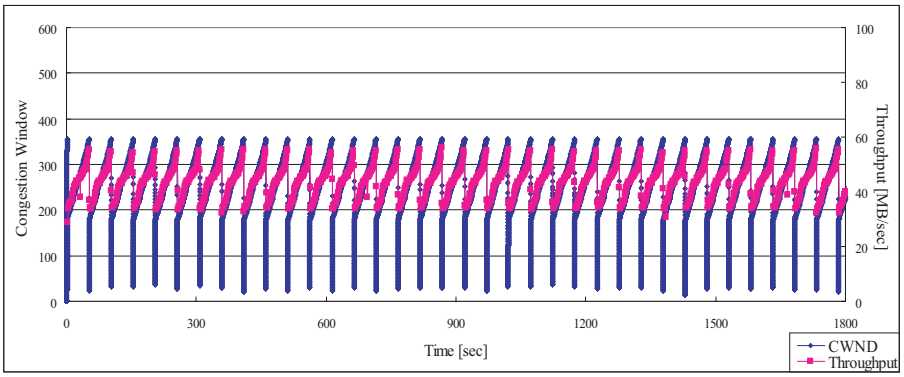

図 7　輻輳ウィンドウをコントロールしない場合のスループットと輻輳ウィンドウの時間変化：マシン 2

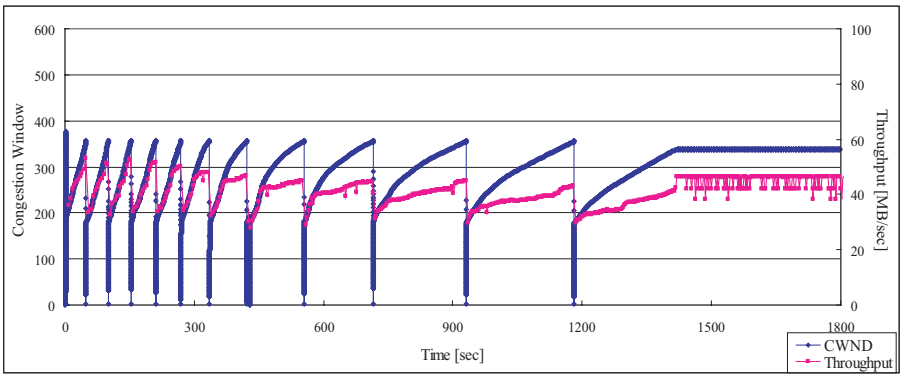

図 8　輻輳ウィンドウをコントロールした場合のスループットと輻輳ウィンドウの時間変化：マシン 2

能で，この輻輳ウィンドウをモニタすることで状態を把握する．モニタした輻輳ウィンドウの低下時に，Target から Initiator 通知することによりリードリクエストのブロックサイズを変更し，輻輳ウィンドウのコントロールを実現している．この手法を実装した Initiator，Target を用いて，iSCSI シーケンシャルリードアクセスを行った．通信初期では，増加しては低い値にリセットされるという変化を繰り返していた輻輳ウィンドウが，徐々に低下するまでの時間間隔が広がり，最終的に一定値となることが確認された．また，輻輳ウィンドウの低下時に同様に低下していたスループットも，輻輳ウィンドウ一定値後はほぼ一定値となり，安定した通信が行われていることが確認された．今後は，Initiator での輻輳ウィンドウ低下のアルゴリズムを検討し，また，高遅延環境に本手法を適用して，その性能を評価する．

謝　　辞

本研究は，一部，文部科学省科学研究費特定領域研究課題番号 13224014 によるものである．

文　　献

[1] 山口実靖，小口正人，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 ”，電子情報通信学会論文誌 Vol.J87-D-I, No.2，pp.216-231，February 2004.

[2] 豊田真智子，山口実靖，小口正人：“ iSCSI ストレージアクセス時における TCP 輻輳ウィンドウとシステム性能の関連性評価 ”，FIT2004 第 3 回情報技術フォーラム，B-004，pp107-109，September 2004.

[3] iSCSI Specification, http://www.ietf.org/rfc/rfc3720.txt?number=3720

[4] SCSI Specification, http://www.danbbs.dk/ dino/SCSI/

[5] P. Sarkar, S. Uttamchandani, and K. Voruganti：“ Storage over IP: When Does Hardware Supporthelp? ”, *Proc. FAST 2003*，*USENIX Conference on File and Storage Technologies*，pp.231-244，January 2003.

[6] P. Radkov, L. Yin, P. Goyal, P. Sarkar and P. Shenoy, “ Performance Comparison of NFS and iSCSI for IP-Networked Storage ”, *Proc. FAST 2002, USENIX Conference on File and Storage Technologies*，pp.101-114，March 2004.

[7] 豊田真智子，山口実靖，小口正人：“ iSCSI ストレージアクセス時の TCP フロー制御のリアルタイム可視化 ”，電子情報通信学会，B-16-9，p618，March 2004.

[8] L. Rizzo：“ dummynet ”, http://info.iet.unipi.it/luigi/ip_dummynet/

[9] InterOperability Lab：“ Univ, of New Hampshire ”, http://www.iol.uhn.edu/consortiums/iscsi/

[10] 山口実靖，小口正人，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 ”，電子情報通信学会第 14 回データ工学ワークショップ DEWS2003，4-B-02，March 2003.