Vol.2009-OS-112 No.13
2009/8/6

# iSCSI 遠隔ストレージアクセスにおける パケット送信処理のカーネルモニタを用いた動作解析

比嘉玲華[†1] 松原幸助[†2] 岡廻隆生[†2]
山口実靖[†3] 小口正人[†1]

iSCSI は複雑な階層構造を持つため，性能を向上させるためには複数の層にまたがる最適化を行う必要がある．iSCSI パラメータの最適化等を行ったところ，一定の性能向上は達成できたものの，高遅延環境においてはまだなお大きな性能低下が確認された．そこで本研究においては，パケット解析および輻輳ウィンドウ解析，ソケットバッファ解析，カーネルのログ解析を行い，その結果に基づいて，iSCSI 遠隔ストレージアクセスにおける性能劣化の原因を検討する．特に遠隔 iSCSI アクセス時に観察される断続的なパケット送出に関して，そのカーネル内部における振舞を詳細に解析し，原因についての考察を行う．

# Analysis of process of transmission of iSCSI remote storage access with kernel monitor

Reika HIGA,[†1] Kosuke MATSUBARA,[†2]
Takao OKAMAWARI,[†2] Saneyasu YAMAGUCHI[†3]
and Masato OGUCHI[†1]

iSCSI has a complex hierarchical structure, SCSI over TCP/IP over Ethernet. Therefore, for the purpose of getting the better performance of iSCSI, optimization through multiple layers is required. iSCSI remote storage access has been optimized with iSCSI parameters. However, in the case of long latency, drastic performance deterioration has still been observed. Thus, in this paper, we have monitored packets, parameters in the kernels including TCP congestion window and socket buffer. Based on the results, we have analyzed the factor of iSCSI performance deterioration.

## 1. はじめに

コンピュータシステムにおける処理データ量の増大に伴い，効率的にストレージを管理したいという要望が高まっている．そこで SAN (Storage Area Network) が登場し，広く用いられるようになった．現在主流として用いられているのは，ファイバチャネルを用いた FC-SAN であるが，構築や管理が非常に高価であるため，より経済的でコストパフォーマンスの高い SAN が望まれている．そこで，次世代 SAN として期待されているのが IP ネットワークを用いた IP-SAN である．iSCSI はその IP-SAN の代表的なプロトコルであり，SCSI コマンドを TCP/IP パケットでカプセル化する規格である[1)2)]．iSCSI を用いることにより広域環境における IP-SAN を低コストで構築でき，遠隔地のデータセンタなどにデータをバックアップすることが容易となるため，ストレージのアウトソーシングといったサービスへの利用が期待されている．

しかし現状において iSCSI は，複雑な階層構成のプロトコルスタックで処理されており，バースト的なデータ転送も多いことから，通常のソケット通信と比較して，特に高遅延環境においては性能の劣化が著しい[3)]．さらに下位基盤の TCP/IP 層が提供できる限界性能を超えることはできない．従って iSCSI を用いたストレージアクセスにおいては，iSCSI 層だけではなく複数の層にまたがる制御を施すことによる性能向上が期待される．既存研究において，iSCSI パラメータの変更等複数の層にまたがる最適化を行った結果，RTT32ms においてデフォルト時よりも約 4 倍の性能向上が得られた[4)]．しかし，高遅延環境においては，なお性能低下が著しいことも確認された．また，4MB のブロックサイズでバースト的通信が行なわれる iSCSI write アクセスを実行しプロセス中の各処理時間を測定することにより，高遅延環境下で性能が劣化する原因を解析したところ，データブロックの送出にかかるデータ転送時間がボトルネックになっていることが確認された．

そこで本研究では，パケット解析，輻輳ウィンドウ解析，ソケットバッファの解析を行うことで，さらなる iSCSI 遠隔ストレージにおける性能劣化の原因の解明を進める．

また，遠隔バックアップを行う場合，データの書き込み量と読込み量とを比較すると，圧倒的にデータの書き込み量のほうが多いこと，遠隔ストレージ側では標準的なシステムのみを用いることができ，カスタマイズできない可能性があることが想定されるため，本研究に

†1 お茶の水女子大学
†2 ソフトバンクテレコム株式会社
†3 工学院大学

おいては，iSCSI シーケンシャルライトアクセスの性能向上に焦点を絞り，Initiator 側における解析を行う．

本稿の構成は以下の通りである．2 節で研究背景を述べ，3 節で本実験システム，実験ツールの概要を述べる．4 節から 7 節において iSCSI リモートアクセスにおいて性能が劇的に低下してしまうことの理由を詳細に解析し，最後に 8 節でまとめる．

## 2. 研究背景

### 2.1 iSCSI

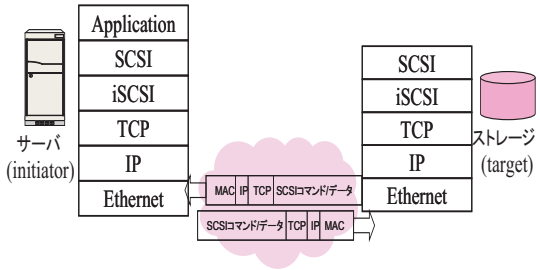


**図 1** iSCSI

IP-SAN の代表的なプロトコルに iSCSI がある．iSCSI は SCSI コマンドを TCP/IP パケットでカプセル化する規格で，iSCSI により SAN を IP 機器だけで構築することが可能となる．IP 機器だけで構築できるということにより，相互接続性が高い，接続距離に制限を受けない，比較的安価に構築可能である，管理が容易である，といったメリットがある．このことにより，広域ネットワークへの適用，具体的には，データセンタなどへのデータのバックアップといった適用に期待されている．

一方で次のようなデメリットも抱えている．iSCSI は図 1 のように複雑な階層構成をとることになり，下位のプロトコルの限界性能を超えることはできない．また，iSCSI には長距離アクセスの実現が期待されているが，広帯域な回線を用いた場合には遅延帯域積の問題も存在する．そこで iSCSI 遠隔ストレージアクセスには複数の層にまたがる適切な制御が求められている．

### 2.2 TCP 送信処理

iSCSI は下位のトランスポート層に TCP を用いる．Linux OS のカーネル内部の TCP 実装において，送信データを格納するソケットバッファは図 2 のように，送信キューにつながれていて処理されるのを待つが，送信されても確認応答 (ACK) を受信するまでは解放されない．[5)]

送信キューは sock 構造体の sk_write_queue メンバで，次に送り出すデータのソケットバッファを指すのが sk_send_head ポインタである．このうち，キューの先頭から sk_send_head の手前までのソケットバッファは，「送信されたが確認応答がまだないために解放できない部分」である（再送キュー：状況によっては再送される）．sk_send_head から先には，これから送信するデータのソケットバッファがつながれている．セグメントを送信したら，sk_send_head をずらしていく．

本研究においては，キューの長さ（waiting ACK の状態から tail まで）について第 7 節にて議論している．

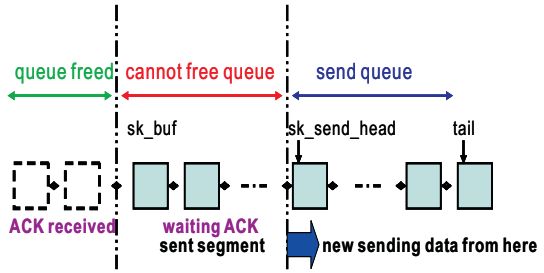


**図 2** 送信キュー

### 2.3 iSCSI リモートストレージアクセス時における性能低下

#### 2.3.1 複数レイヤにまたがる iSCSI 最適化

iSCSI は複雑な階層構造をとなっている．そこで，図 3 のように，複数レイヤにまたがる最適化を行った[4)]．SCSI/iSCSI 層においては，iSCSI パラメータ最適化を行い，その結果，RTT32ms の場合に約 4 倍のスループットの向上が確認できた．

ただし SCSI/iSCSI 層の最適化による性能向上はウィンドウサイズにより制限される可能性があるため，次に TCP/IP 層における最適化を試みた．具体的にはスループットと関係の深い輻輳ウィドウの値を決める輻輳ウィンドウ制御アルゴリズムを変更して性能測定を行った．その結果，アルゴリズムごとのスループットの違いがソケット通信時には見られたが，iSCSI 利用時には見られなかった．すなわち，本実験環境においては TCP 輻輳制御ア

ルゴリズムの変更は，iSCSI 性能最適化には影響を与えないと言える．この原因としては，TCP の輻輳ウィンドウの違いによる性能向上分が，iSCSI のブロックアクセスのシーケンスに吸収され消えてしまっていると考えられる．

最後に Ethernet 層における最適化として NIC のパラメータを変更し iSCSI 通信を行ったところ，RTT = 32ms の場合において約 5%の性能向上が確認できた．

このように iSCSI アクセス時の最適化によって約 4 倍の性能向上が得られた．この結果を，図 4 に示す．また，比較としてソケット通信およびローカルディスクアクセススループットも測定した．ソケット通信の測定には Iperf[6] を，ローカルディスク，iSCSI の測定には bonnie++[7] を使用した．本実験においては，通信の妨げにならない程度の十分なウィンドウサイズを想定し，広告ウィンドウを設定した．また，iSCSI Target を起動する際に使用するコマンドにおいて，デフォルト状態では広告ウィンドウを 1MB に設定するようになっているが，本実験においては十分な大きさではないため，コマンドを書き換えて，iSCSI 起動時の広告ウィンドウも十分な大きさになるように設定を変更した．

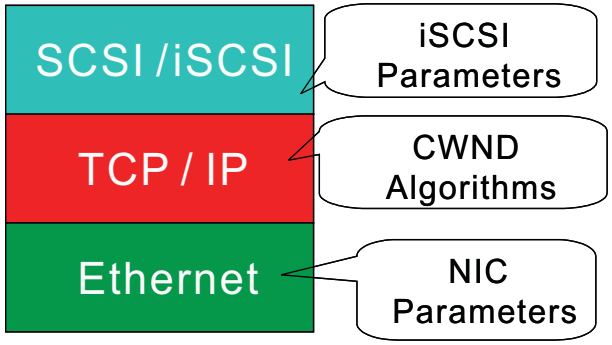


**図 3** 複数レイヤにまたがる最適化

#### 2.3.2 高遅延環境下での性能低下

複数の層ににまたがる最適化を行なった後の結果を図 4 に示す．図 4 からわかるように，ソケット通信の場合は高遅延環境においてもスループットを保っているのに対して，iSCSI 通信の場合は低遅延環境で保たれていたスループットが高遅延環境において著しく性能低下している．複数の層にまたがる最適化を行なうことでデフォルト状態と比較して RTT32ms において約 4 倍の性能向上が達成されたが，高遅延環境下での性能の劇的な低下は解消されていない．

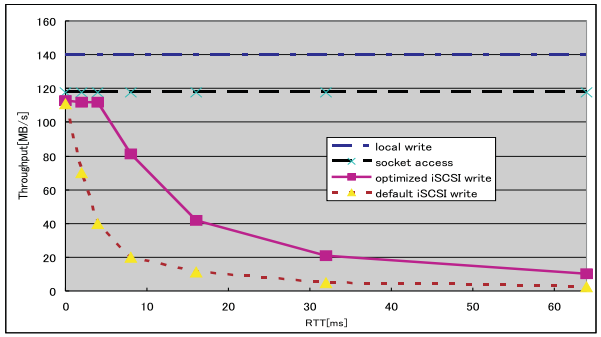


**図 4** スループット比較

そこで，本稿においては，さまざまな角度から解析を行うことで性能低下の原因を詳細に調べる．

## 3. 実験システム

### 3.1 プロトコルアナライザ

高遅延環境において性能が著しく劣化する原因を解明するため，本研究ではまず，ネットワーク上を往来するパケットを調べる．ネットワークからキャプチャしたトラフィックを直接高速アクセス可能な HDD に書き込む大容量ネットワークアナライザである ClearSight 社の Network Recorder[8] を設置し,iSCSI アクセス時のパケットキャプチャを行った．

### 3.2 カーネルモニタツール

本実験では次に，TCP カーネルの振舞をモニタするツールを構築した．図 5 に示すように，カーネル内部の TCP ソースにモニタ関数を挿入しカーネルを再コンパイルした．これによりモニタできるようになった値には，輻輳ウィンドウ，ソケットバッファのキュー長の他，各種エラーイベント (Local device congestion，重複 ACK，SACK 受信，タイムアウト検出) の発生タイミングなどがある．

### 3.3 実験システム

本研究において,Initiator と Target 間は GigabitEthernet で接続し,TCP/IP コネクションを確立した.Target のストレージには SAS ディスクを用い RAID コントローラによる RAID0 構成で接続した. 使用した実装システムと実験環境を図 6 および表 1 に示す.

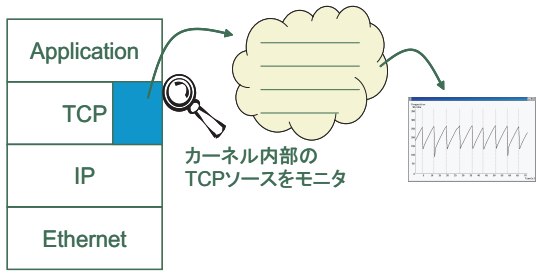


**図 5** カーネルモニタツール

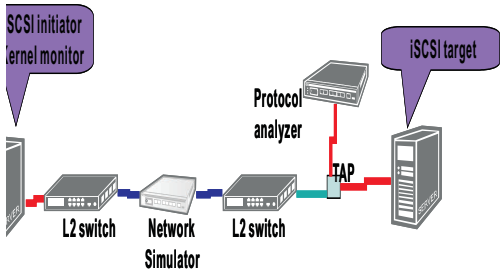


**図 6** 実装システム概要

## 4. iSCSI シーケンシャルライトアクセスモデル

iSCSI アクセス時の複数の層にまたがる最適化を行なった上で，iSCSI write アクセスモデルを構築しその解析を行なった．その結果，高遅延環境における iSCSI アクセスが理論値よりも低下する理由は，以下のように，RTT によらずに一定であるはずのデータブロックの送出にかかるデータ転送時間が RTT に比例する値になっていることが原因であるとわかった．

### 4.1 iSCSI write アクセスモデル

高遅延環境において性能が低下する原因となるボトルネックを以下のように調べた．

まず，dd コマンドを用いて実行される iSCSI ブロックアクセスのパケットをプロトコルアナライザを用いて解析したところ，様々な大きさの複数のパケットが入り混じって飛んでいたため，モデル化の検証に当たっては sg_dd コマンドを使用した．sg_dd コマンドは，dd コマンドと文法的に互換性があるが，dd コマンドとは異なり，iSCSI アクセスに

**表 1** 実験環境

| | |
|---|---|
| OS | Red Hat Enterprise Linux 2.618-8.e.15 |
| CPU | Quad Core Intel Xeon 1.6GHZ |
| Main Memory | 2GB |
| NIC | Intel PRO/1000PT Server Adaptor on PCI Express |
| HDD | 73GB SAS × 2(RAID0) |
| RAID Controller | SAS5/iR |
| iSCSI | Initiator : open-iscsi-2.0-865 |
| | Target : iSCSI Enterprise Target(IET)-0.4.15 |
| Network Analyzer | ClearSight Network Recorder |
| Network Simulator | ANUE |

おいて SCSI レベルで指定したブロックサイズによるアクセスが可能となるコマンドである[9]．カーネルを再構築したことで sg_dd コマンドを用いたアクセス時に最大 4MB 程度のブロックサイズでのアクセスが可能となった．それに伴い，iSCSI パラメータの設定を FirstBurstLength,MaxBurstLength ともに 4,194,304 とした．

4096KB のブロックサイズで write アクセスを実行したときのプロセスは図 7 のようになる．このとき Ta とは Initiator 側における最初のパケット送出から最後のパケット送出までのデータ転送時間，Tb は Target 側で書き込みが終了し Initiator へ書き込みが終了したことを知らせるまでの時間，Tc は次の write が実行されるまでの時間である．遅延装置で設定した遅延時間ごとに Ta，Tb，Tc，RTT を測定することにより，高遅延環境下で性能が劣化する原因を解析する．2048KB，4096KB のブロックサイズで write アクセスを実行した．このときの RTT は 0ms，2ms，5ms，10ms，20ms，50ms とした．

### 4.2 解析結果

Ta，Tb，Tc，RTT をアナライザを用いて測定した結果，Tb と Tc はほぼ定数であること，RTT は遅延装置で設定した値とほぼ等しいということが確認された．しかし，Ta は図 8 に示すように RTT に比例する値で，RTT の増大と共に増加していた．すなわち高遅延環境における iSCSI アクセスが理論値よりも低下する理由は，RTT によらずに一定であるはずのデータ転送時間が RTT に比例する値になっていることが原因であるとわかった．

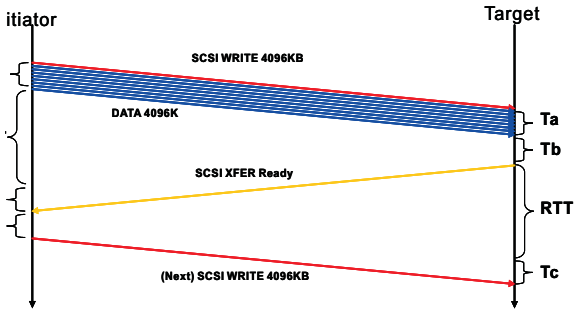


**図 7** iSCSI write アクセス実行図

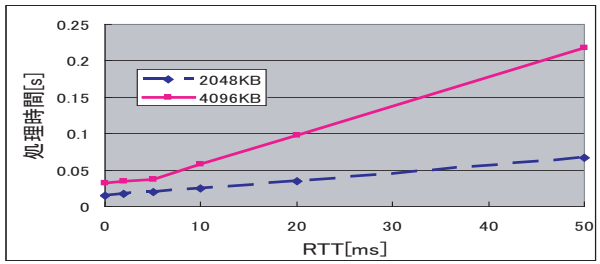


**図 8** データ転送時間 Ta の測定結果

## 5. アナライザを用いた用いたパケット解析

前節に述べた解析結果より，高遅延環境における性能低下の原因が，データ転送処理にあることがわかった．そこで，本節ではアナライザを用いてどのようなパケットがやりとりされているのかを詳細に調べる．

### 5.1 Initiator における送出パケット解析

RTT20ms，ブロックサイズ 4MB の iSCSI アクセスを実行した際の Initiator 側から Target 側に向かって送出されたパケットをアナライザを用いて解析を行った．そのときのパケット解析結果を図 9 に示す．グラフは横軸が時刻、縦軸がパケット番号を表している．パケットと比較するため，write10 コマンドと dataout コマンドの送出タイミングを上部に並べて示した．図 9 からわかることは，write10 コマンドの後に dataout コマンド 15 個が繰り返され，write10 パケットの後には 4MB のパケットが繰り返されているということ，SCSI レベルで 4MB と指定して送信しているはずのパケットが 4 つに分かれて送信されていること．つまり，パケット送信が断続的に行なわれているということである．このような振舞は，通常のソケット通信の場合には見られない．

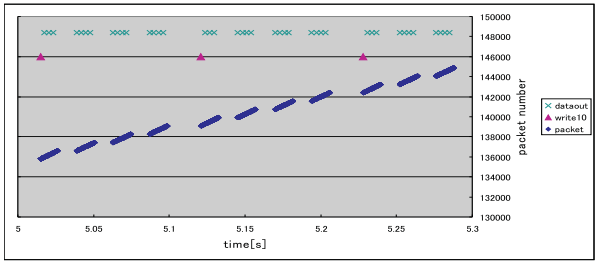


**図 9** RTT20ms におけるパケット解析

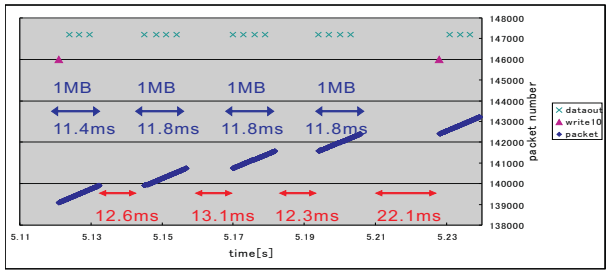


**図 10** 拡大した RTT20ms におけるパケット解析

図 9 のうちの一周期を拡大したものを図 10 に示す．

図 10 より，短い時間に連続してパケットが送信された後，突然パケットの送出が止まっていること，一定時間の後に再びパケットの連続送信が行われていること，パケットの送出数は約 730 個であること，それらパケットの送出開始から次の送出再開までの間隔は RTT にほぼ等しい約 20ms であることがわかる．また，RTT を変化させて同じ実験を行なったところ，RTT = 50ms，80ms においてもパケットの送出開始から次の送出再開までの間隔は RTT にほぼ等しい値になったことから，一連のパケット送出間隔は RTT にほぼ等しいと考えられる．

### 5.2 TCP ACK パケットの解析

送出再開の前後にはどのような現象が起こっているのかを解明するために，送出再開の直前のパケットを詳しく調べたところ，Target 側から Initiator 側への TCP ACK のみが存在していた．また，write10 コマンドで送出が再開される場合は全ての ACK が帰ってきて初めてパケット送出再開が行なわれていることが確認された．

### 5.3 アナライザを用いたパケット解析の考察

RTT = 20ms，ブロックサイズ 4MB の iSCSI アクセスを実行時のパケットをアナライザを用いてキャプチャし詳細な解析を行なった結果からは，短い時間に連続してパケットが送信された後突然パケットの送出が止まっていること，それらのパケットの送出間隔は RTT に等しい約 20ms であること，また，送信再開の前後には TCP ACK のみが受信されていることがわかった．これらの結果から，パケットの送信の断続性の原因としては，輻輳ウィンドウ切れ，すなわちウィンドウを使い切った可能性がまず最初に考えられる．そこで，次節でカーネルモニタを用いてこの時の輻輳ウィンドウの値を調べた．

## 6. カーネルモニタを用いた輻輳ウィンドウ解析

### 6.1 輻輳ウィンドウ解析

スループットと輻輳ウィンドウには密接な関係があることが知られている．そこで，Initiator 側でカーネルモニタと tcpdump コマンドを使って輻輳ウィンドウの値とパケット送出量の関係を調べた．RTT = 20ms，ブロックサイズ 4MB の iSCSI アクセスを実行したときの輻輳ウィンドウとパケット解析の結果を図 11 に示す．RTT = 20ms において 4MB を非同期に送信するには，輻輳ウィンドウは約 3000 が必要であるが，図 11 に示されたように，輻輳ウィンドウは約 1200 であり，4MB を送信するには十分な大きさではない．

### 6.2 輻輳ウィンドウ解析の考察

図 11 を一見すると，輻輳ウィンドウが十分な値でないことから輻輳ウィンドウ切れがパケットの送出停止の原因として考えられるが，図 10 と図 11 を合わせてみると，それが原因ではないことが推測される．すなわち図 10 からパケットの一周期あたりの送出量は約 730 であったが，輻輳ウィンドウの値は 1200 であることが図 11 により確認された．従って，パケットの送出停止は輻輳ウィンドウを使い切ったことが原因ではないことになる．もし，輻輳ウィンドウが原因でパケットの送出停止が起こったのなら，輻輳ウィンドウ 1200 の分だけのパケット，つまり約 1.8MB のパケットを送出することが可能だが，最大でも 1MB の大きさしか送出されていないからである．さらに，輻輳ウィンドウが本当に余っているかオ

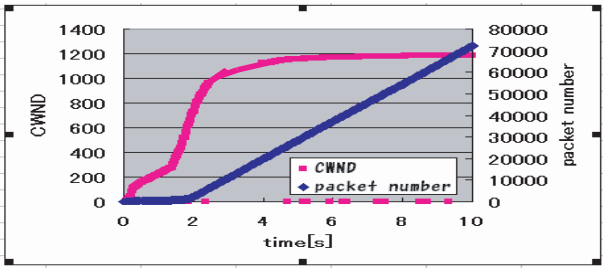


**図 11** RTT20ms における輻輳ウィンドウ解析

ンザフライ状態のパケットを調べることで確認した[10]．その結果，実際にネットワーク上を飛んでいるパケットは最大で約 1MB であり，輻輳ウィンドウを使い果たしていない状態であることが確認された．

やはり，輻輳ウィンドウが余っているにも関わらず iSCSI 通信中に送信パケットの断続性が見られるということであった．

## 7. TCP ソケットバッファ解析

前記の解析結果により，パケット送信断続の原因は，広告ウィンドウ，輻輳ウィンドウの両者ではないことがわかった．そこで，本節においては送信処理におけるソケットバッファの振舞を解析していく．

### 7.1 iSCSI 通信における詳細なキュー解析

iSCSI 通信におけるキューの振舞について詳細に解析していく．

RTT を 20ms，アクセスブロックサイズを 4MB，広告ウィンドウを通信の妨げにならない十分な値である 4MB に設定した．iSCSI 通信を測定するときには sg_dd コマンドを実行した．図 12 から図 14 では，Initiator 側で tcpdump コマンドを実行したときのパケットの送出と，カーネルモニタを用いて記録したキューの長さを比較したものである．横軸を時間，縦軸をキューの長さ，第二縦軸をパケットの送出番号とする．このとき ACK の縦軸は意味を持たずタイミングのみの表記とする．

RTT20ms において iSCSI 通信時の定常状態以降のキューの変化を表したのが図 12 である．この箇所は 4MB の iSCSI 通信が行なわれた時における一周期のものである．

次に，図 12 における 6.254 秒から約 6.264 秒まで，つまり write10 コマンドが発行されてから最初のパケットの断続が生じるまでの箇所を拡大したのが図 13 である．ACK が戻っ

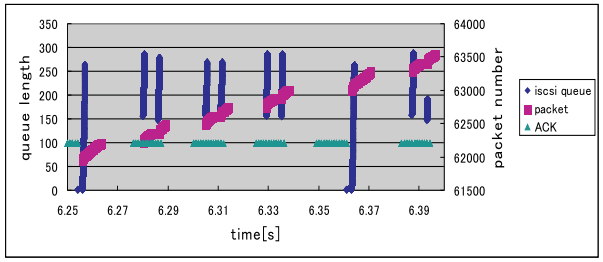


**図 12** RTT20ms における iSCSI キュー長変化拡大 no.1

てきたことがトリガとなりキューの成長，パケットの送出が生じている．このとき，キューの成長の停止のあとにパケットの送信停止が生じていた．

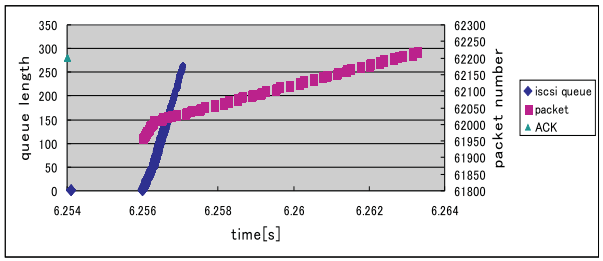


**図 13** RTT20ms における iSCSI キュー長変化拡大 no.2

次に，約 6.263 秒から約 6.283 秒までを拡大したのが図 14 である．つまり直前のパケット送出停止から次のパケット送出再停止が生じるまでの箇所を拡大したのが図 14 である．ACK が戻ってきたことがトリガとなり，キューの成長とパケットの送出再開が生じている．

以上のグラフからわかる振舞として，パケットの送出停止が起こった後に，ターゲットからの ACK が帰ってきたことでキューが解放され，キューに空きが生じた．キューが成長したことでパケットが送信可能になり送出されるが，その後，キューが限界になったのでパケットの送出が停止した．そしてまた RTT 後に，ターゲットからの ACK が帰ってきたことで，キューに空きが生じパケットが送信可能になるという繰り返しが起こっていると推測される．

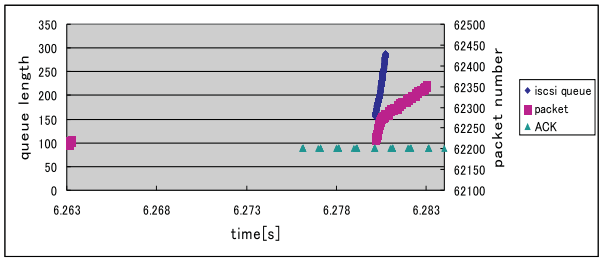


**図 14** RTT20ms における iSCSI キュー長変化拡大 no.3

### 7.2 iSCSI 通信とソケット通信地におけるキュー長比較

ソケット通信においては，高遅延環境においても高い性能を維持していたにも関わらず，iSCSI 通信においては高遅延環境になると性能の劇的な低下が確認されている．そこで，iSCSI 通信時，ソケット通信時におけるソケットバッファのキューの振舞をカーネルモニタを用いて比較する．RTT を 20ms，アクセスブロックサイズを 4MB，広告ウィンドウを通信の妨げにならない程度の十分な値に設定した．ソケット通信を測定するときには Iperf を実行し，iSCSI 通信を測定するときには sg_dd コマンドを実行した．図 15 と図 16 は，Initiator 側で tcpdump コマンドを実行したときのパケットの送出と，カーネルモニタを用いて記録したキューの長さを比較したものである．横軸を時間，縦軸をキューの長さ，第二縦軸をパケットの送出番号とする．

#### 7.2.1 RTT20ms におけるキュー長比較

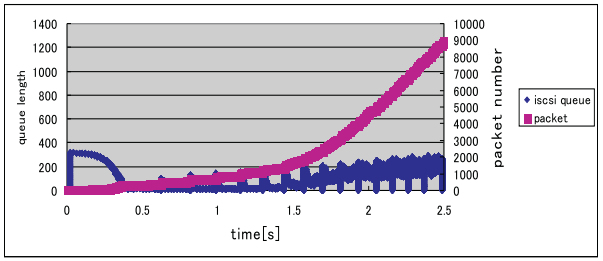


**図 15** iSCSI 通信キュー長変化

図 15 からわかるように，iSCSI 通信においてはキューの最大値は約 300 であり，パケッ

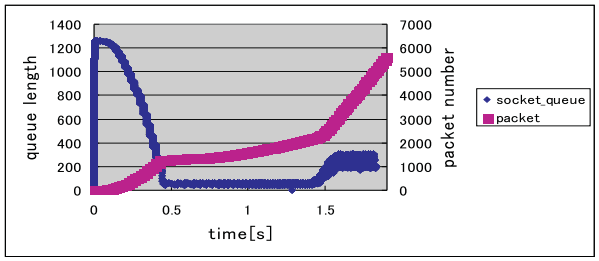


**図 16** ソケット通信キュー長変化

トの送出と似たようなタイミングで 0 から 300 を推移している．しかし，ソケット通信においての最大値は約 1300 であり，パケットの送出が一定になった後は 200 から 300 を推移していることが図 16 からわかる．このように，iSCSI 通信とソケット通信ではキューの振舞に明らかなる違いが確認された．

### 7.2.2 カーネルモニタによるログ追跡

iSCSI 通信とソケット通信の両者の両者においてどのようにカーネルの処理がされているのかを，カーネルモニタのログを詳しく追っていったところ，通過する場所は違うものの、両者ともに timeout が生じていることがわかった．ソケット通信においては，図 17 における 1415 行目の schedule() を通過し，iSCSI 通信においては 1439 行目の schedule() を通過する際にタイムアウト待ちに入り RTT 後の ACK で再開されている．

両者ともにタイムアウト待ちに入っているのにも関わらず，ソケット通信においては性能は維持されたままで，iSCSI 通信においては性能低下を引き起こしている．その原因としては，図 16 からわかるように，ソケット通信の場合はソケットバッファのキューに十分なデータが保持されているからパケット送出は途切れないということが考えられる．

### 7.3 キュー解析における考察

iSCSI 通信において，ACK が帰ってきたことでキューの解放が行なわれ，キューの再成長と送信の開始が始まるが，割り当てられたキューが消費され，パケット送信の停止が生じ，再度 ACK が帰ってきたことで，キューに空きが生じパケットが送信可能になるという繰り返しが起こっていることがわかった．このような振舞は通常のソケット通信においては観測されない．

また，ソケット通信と iSCSI 通信の高遅延環境におけるキュー長の振舞が明らかに異なることが確認された．ソケット通信においては，通信の開始時に大きな値までキューが成長

```
1400 fastcall signed long __sched schedule_timeout(signed long timeout)
1401 {
1402 struct timer_list timer;
1403 unsigned long expire;
1404
1405 switch (timeout)
1406 {
1407 case MAX_SCHEDULE_TIMEOUT:
1415 schedule();                          Socket access
1416 goto out;
1417 default:
1425 if (timeout < 0)
1426 {
1427 printk(KERN_ERR "schedule_timeout: wrong timeout "
1428 "value %lx from %p\n", timeout,
1429 __builtin_return_address(0));
1430 current->state = TASK_RUNNING;
1431 goto out;
1432 }
1433 }
1435 expire = timeout + jiffies;
1436
1437 setup_timer(&timer, process_timeout, (unsigned long)current);
1438 __mod_timer(&timer, expire);
1439 schedule();                          iSCSIaccess
1440 del_singleshot_timer_sync(&timer);
1441
1442 timeout = expire - jiffies;
1443
1444 out:
```

**図 17** kernel ソースコード

しパケットは送出され続けるのに対して，iSCSI 通信においてはキューが約 300 で頭打ちになり，パケット送出が断続的になってしまっている．この振舞が，ソケット通信においては高遅延環境においてもほぼスループットを維持するのに対して，iSCSI 通信においてはスループットの劇的な低下が生じる原因になっていると考えられる．つまり，iSCSI 通信においては TCP 層で使用可能な（確保されている）メモリ領域の大きさががソケット通信時よりも小さい値となっており，バースト的な iSCSI 通信を行なったときにはキューを使い果たした状態になるため，送信の断続性が生じているということとなる．

## 8. まとめと今後の課題

既存研究では，iSCSI 遠隔ストレージアクセスの性能を高めるために，複数の層にまたがる最適化を行った．その結果，最適化 iSCSI はデフォルト状態の iSCSI と比較して約 4 倍の性能向上を達成することが出来た．しかし，なお高遅延環境における性能の低下が著しいことから，本研究では高遅延環境における性能低下の原因をより深く調べるために，iSCSI ブロックアクセスのモデル化を行い解析した結果，高遅延環境における性能低下の原因がデータ転送時間であることが判明した．

また，ネットワーク上を飛来するパケット解析とカーネル内部の輻輳ウィンドウ解析，ソケットバッファ解析を行った．その結果，パケットの送出は断続的であり，送信開始の前後

には TCP ACK のみが受信されたことが確認された．このことから，iSCSI の送信において停止・開始の制御は TCP レベルで行われていることが推測され，輻輳ウィンドウを使い切ったことがパケット送出停止の原因である可能性が考えられた．しかしさらに調べた結果，パケットの 1 周期の送出量はその輻輳ウィンドウの値を使い切る量ではなかったことがわかった．

パケット送信断続の原因が広告ウィンドウでも輻輳ウィンドウでもないことから，パケット送信処理をより詳細に調べるためにソケットバッファを調べたところ，iSCSI 通信時とソケット通信時においてキュー長の振舞に明らかなる違いが確認された．そのことより，iSCSI 通信においては使用可能な（確保されている）メモリ領域の大きさががソケット通信時よりも小さい値となっており，バースト的な iSCSI 通信を行なったときにはキューが枯渇し，パケット送信の断続性が発生したと考えられる．

従って，iSCSI の送信において停止・開始の制御は TCP レベルで行われているが，その制御は輻輳ウィンドウの値だけによるものではなく，ソケットバッファの割り当て大きさが原因になっていると推測される．

これらの結果は，Linux カーネルの実装に依存する問題であるかのようにも考えられるが，しかし，Windows 環境での iSCSI 実装において高遅延環境における iSCSI ストレージアクセスを行なった場合も，同様の性能の劇的な低下が確認されることが知られていることから，パケット送信の断続性の振舞は Linux カーネルの実装に依存した振る舞いではなく iSCSI を用いた際に広く共通する問題である可能性が考えられる．

今後の課題としては，本稿では Linux カーネルに限った実験を行なっているが，Linux 環境において解析を進めた後，Windows 環境においても性能測定をすることで Windows 環境における iSCSI の振舞も調べていく．Linux 環境における更なる解析として．具体的には，ソケットバッファの容量はどこで決められているのかを特定し，ソケットバッファを大きくする方法を検討し制御することで，原因の特定とシステムの性能改善を実現したい．

## 参 考 文 献

1) iSCSI Specification,
http://www.ietf.org/rfc/rfc3720.txt?number=3270
2) SCSI Specification,
http://www.danbbs.dk/˜dino/SCSI/
3) 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.216-231，2004 年 2 月
4) 比嘉玲華, 松原幸助, 岡廻隆生, 山口実靖, 小口正人：”iSCSI 遠隔ストレージアクセスの複数レイヤにまたがる最適化” インターネットコンファレンス 2008，pp.120，沖縄，2008 年 10 月．
5) 高橋浩和，小田逸郎，山幡為佐久 : ”Linux カーネル 2.6 解読室”，ソフトバンク　クリエイティブ，2006 年 11 月
6) http://dast.nlanr.net/Projects/Iperf/
7) http://www.textuality.com/bonnie/intro.html
8) http://www.toyo.co.jp/clearsight/product/analyzer.html
9) http://sg.torque.net/sg/sg3_utils.html/
10) 比嘉玲華, 松原幸助, 岡廻隆生, 山口実靖, 小口正人：”パケット解析と輻輳ウィンドウ解析による遠隔 iSCSI アクセスの断続的パケット送出に関する考察”, DEIM2009，E1-1，掛川，2009 年 3 月