# iSCSI を用いた大域 IP-SAN に対する小粒度アクセス性能に関する考察

喜連川 優
情報理工学系研究科電子情報学専攻

## 概要

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. 本研究では, 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の解析を行う. 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確認応答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.

## 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)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 Channel) 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 に大きな期待が集まっている. そこで我々は図の様に 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[1]. そして, 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解析を行い, そのターンアラウンドタイムについて考察する. 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れる. 本報告ではまず既存の 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 のアルゴリズムや遅延確認応答などの 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能を大きく向上できることを示す.

## 2. TCP のアルゴリズムの紹介

### 2.1. Nagle のアルゴリズム

Nagle のアルゴリズムは TCP に実装されているアルゴリズムの一つであり, 多数の微少なパケットが送信される非効率的な通信を回避するために"確認応答されていない MSS (Maximum Segment Size)未満の微少なパケットの送信を最大 1 個までとする". これにより多数の微少パケットが送信されることが回避されるが送信の遅延を招くこともあるため, 多くの TCP 実装では TCP_NODELAY オプションによりこれを無効化することも可能となっている.

### 2.2. 遅延確認応答

TCP ではデータ受信者は確認応答(Ack)を送信して正常に受信したことを伝える. しかし, 1 パケット毎に Ack を送信することは非効率的であり, 複数パケットの受信に対して Ack を送信することや送信データとともに Ack を送信するピギーバックが効率的である. よって TCP では単独のパケットを受信しても Ack を即時には送信しない遅延確認応答が実装されている. これにより単独のパケットに対する Ack 送信はタイムアウト時間まで保留される.

## 3. iSCSI アクセス性能測定

本章において, 大域環境における小粒度 iSCSI アクセスの性能について述べる.

## 3.1. 実験方法

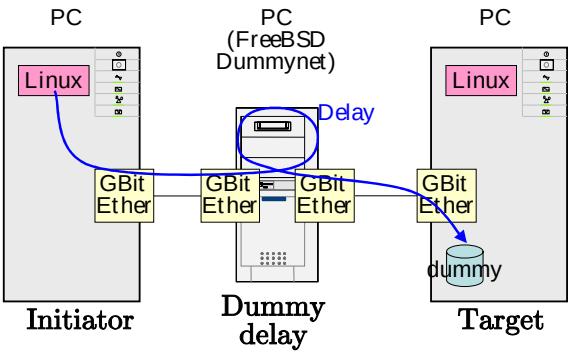


**図 1 実験環境**

図 1 のような実験環境を構築し，性能測定を行った．すなわち，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．接続は，途中に人工的な遅延装置として FreeBSD Dummynet を挟んでクロスケーブルで接続し仮想的な大域ネットワークを構築した．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．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)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我々が変更を施したもの(後述)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．同環境において，以下の実験を行いその性能を測定した．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．これにより純粋な TCP の振る舞いの影響を考察できる．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．

## 3.2. 性能測定結果

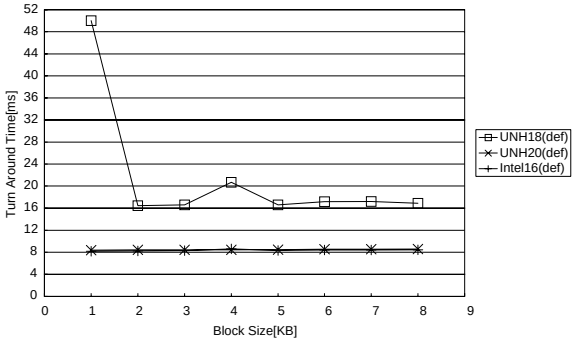


**図 2 性能測定結果 A(実装改変無し)**

前節の実験により，各実装の性能を測定し，図 2 の UNH18(def), UNH20(def), Intel16(def)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，同実装に対し我々が改変を行っていないものである．``(def)"は default を意味し後述する我々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る．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 において性能が

著しく劣る(約 6 倍)ことがあることが確認された.

## 4. 解析

### 4.1. 動作解析

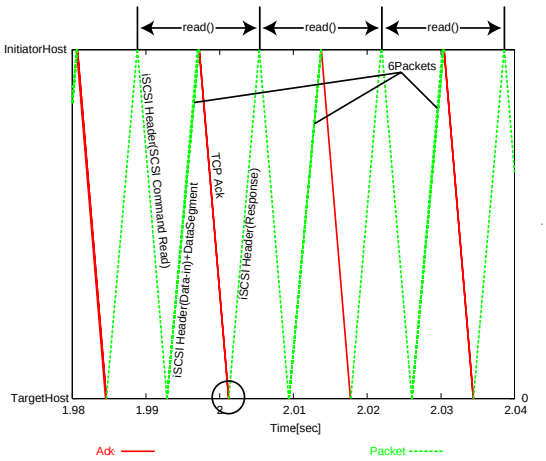


**図 3 UNH18 実装, 片道遅延時間 4ms, ブロックサイズ 8KB, パケットの転送図(60ms 間)**

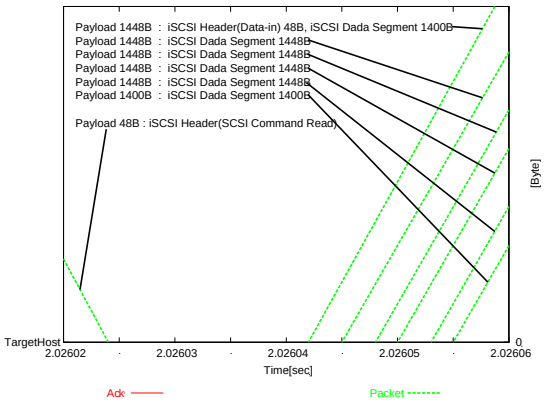


**図 4 UNH18 実装, 片道遅延時間 4ms, ブロックサイズ 8KB, パケットの転送図(0.04ms 間)**

本節では各iSCSI実装の振る舞いの解説を行う. 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 3 に示す. また同図の丸記号で示した部分の拡大を 図 4 に示す. 図 3 より, システムコール **read()** 1 回につき 2 往復を要していることが確認され, iSCSI Response の送信は TCP Ack の受信を待って行われていることが確認された. 図 4 は Target が iSCSI Read を受信し, iSCSI Data-in を返信する部分の拡大図である. 同図より, iSCSI Data-in PDU が MSS(本例では 1448 バイト)毎に分割されて送信されている様子が確認できる. また, 最後のパケットのサイズは MSS で分割した剰余となり MSS よりも小さいことも確認できる. Nagle のアルゴリズムが有効となっている同実装ではこの剰余パケットを既に送信しているため iSCSI Response(48 バイト) の送信を Ack 受信後まで保留することとなり結果として往復回数を 1 回増加させ, ターンアラウンドタイムを約 2 倍に増加させていることが確認された.

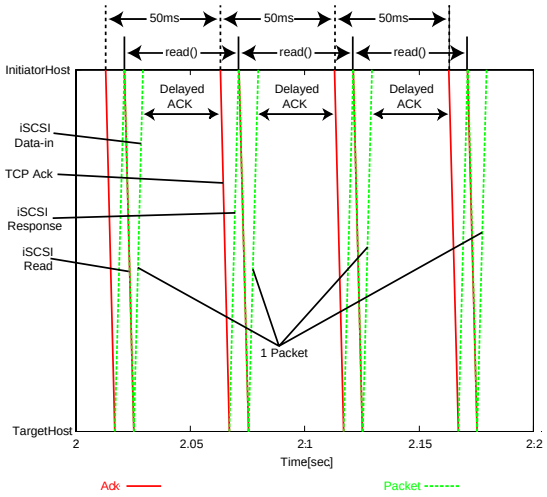


**図 5 UNH18 実装, 片道遅延時間 4ms, ブロックサイズ 1KB, パケットの転送図**

図 5 に性能が著しく劣るブロックサイズが 1KB におけるパケットの転送を示す. 同図より, Data-in PDU のサイズが MSS 未満となる 1KB の read では, 単独の TCP パケットの送信が発生しており, 遅延確認応答が動作していることが確認できる. これにより, Ack の送信が遅延され, iSCSI Response の送信が遅延され, 結果的にターンアラウンドタイムを著しく増加させていることが確認された.

同様の解析を Intel16 実装に対し行うことにより, 同実装では Nagle のアルゴリズムが無効化されており往復回数の増加が回避されていることが確認された. すなわち, iSCSI Data-in PDU の最後のパケット(MSS 未満の微少なパケット)送信の後に Ack の受信を待つことなく iSCSI Response PDU(2 個目の MSS 未満の微少なパケット) を連続して送信しており 1 回の **read()** に 1 往復のみ要している. また Ack の受

信を待たないため遅延確認応答が動作し Ack の受信が大きく遅延されてもそれが性能に影響を与えていない.
次に UNH20 実装に対し解析を行うことにより, 同実装は各 **read()**ごとに iSCSI Response を送信しておらず往復回数の増加が回避されていることが確認された. Data-in PDU の送信をもって 1 回の **read()** を終了するため iSCSI Response の送信ために Ack の受信を待つことがなく, 遅延確認の影響も受けていない.
この様に解析を行うことにより性能に大きな差(約 2 倍)が現れる理由や, 小さいブロックサイズ(1KB)において性能が著しく劣化する理由が確認された.

## 4.2. 参考実験

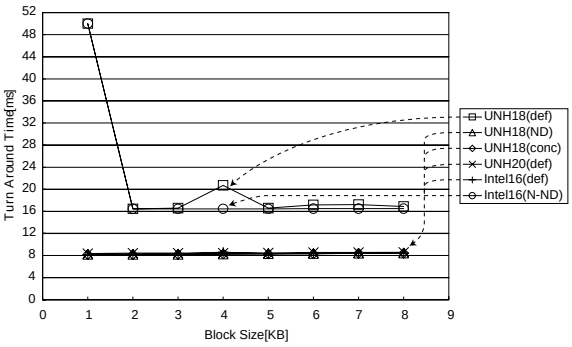


**図 6 性能測定結果 B(実装改変あり)**

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, TCP の振る舞いを考慮した iSCSI ドライバの実装が重要であることが確認された. 参考実験として, 既存の iSCSI 実装に対して以下の様な変更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)の変更を行ったもの. (2) UNH 18 iSCSI 実装に対し Data-in PDU と Response PDU の結合改変を行ったもの. (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)を削除したもの. 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 6 に示す. 同図より, UNH18(ND)および UNH 18(conc)のターンアラウンドタイムが約半分となること, Intel16(N-ND)のターンアラウンドタイムが約 2 倍になることが分かり, 前述の考察の様に TCP の振る舞いがターンアラウンドタイム性能に大きな影響を与えること及び考察の正しさが確認された.

## 5. おわりに

iSCSI アクセス解析システムを構築し既存の iSCSI 実装の動作を確認することにより, TCP に実装されているアルゴリズムの振る舞いが iSCSI 性能に大きな影響を与えることが確認された. iSCSI のドライバの実装には TCP の詳細な振る舞いを考慮しての実装が重要であるといえる. 今後は実ストレージデバイスを用いた iSCSI アクセスの動作の解析などを行っていく予定である.

## 参考文献

[1]山口実靖, 小口正人, 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, 2004 年 2 月(採録済み)