長距離 TCP 高速化機構の開発

下見淳一郎、河合　純、下國　治、陣崎　明 †

中村　誠、稲葉真理、平木　敬 ‡

† 株式会社　富士通研究所　ユビキタスシステム研究センター
〒211-8588　川崎市中原区上小田中 4-1-1
E-mail: {shitami,jkawai,osamus,zinzin }@labs.fujitsu.com
‡ 東京大学　情報理工学研究科
〒113-0033　東京都文京区本郷 7-3-1
E-mail: {makoto,mary,hiraki}@is.s.u-tokyo.ac.jp

Development of a Long Distance TCP Accelerator

Junichiro Shitami, Jun Kawai, Osamu Shimokuni, Akira Jinzaki †
Makoto Nakamura, Mary Inaba, Kei Hiraki ‡

† Ubiquitous System Research Center, Fujitsu Laboratories Ltd.
211-8588 Kamikodanaka 4-1-1, Nakahara-ku, Kawasaki, Kanagawa, Japan
E-mail: {shitami,jkawai,osamus,zinzin }@ labs.fujitsu.com
‡ Department of Information Science, University of Tokyo
113-0033 Hongo 7-3-1, Bunkyo-ku, Tokyo, Japan
E-mail: {makoto,mary,hiraki}@is.s.u-tokyo.ac.jp

長距離 TCP でネットワーク帯域を 100%活用できない問題を解決するために Comet TCP 技術を開発した。Comet TCP は独自の高速高信頼プロトコルである LFT（Long Fat Tunnel）を用いてエンド・エンドの TCP 通信を透過的に遠隔中継することで TCP 通信を高速化する。これを実現するため LFT では Window サイズを固定とし帯域制御によって輻輳制御を行う方式を実装した。本論文は Comet TCP 方式、基本性能、長距離 TCP 高速化ゲートウェイ装置「Comet TCP Accelerator」の実現と評価結果を示す。Comet TCP Accelerator は帯域 800Mbps、RTT 200ms の長距離回線を用い、セッション時間 120 秒で 700Mbps 以上の短時間 TCP 性能を実現可能である。

1. はじめに

ネットワーク遅延が大きくなると TCP（Transmission Control Protocol [1]）の通信性能が極度に低下することはよく知られている。具体的には日本とアメリカ東海岸を Gigabit Ethernet で接続した場合（Round Trip Time: RTT 200ms）の典型的な実効 TCP 性能はパケットロスが全くない場合でも 4Mbps 程度である。東京・大阪間（RTT 20ms 程度）でも 40Mbps 程度と回線帯域の 4%しか利用できない。

かつては長距離回線が比較的低速であったため、この特性は顕在化しなかった。しかし、近年著しい光ファイバネットワークの広帯域化、長距離化に伴ってインターネット活用の大きな問題となってきた。例えば企業が「ビジネスの継続性（Business Continuity）」を確保するために、災害などによるデータ喪失に備えて重要データの遠距離バックアップシステム（Disaster Recovery System）が必須となっているが、これらのシステムでは国際はもちろん日本国内のバックアップにおいても転送性能問題が生じている。

そこで数年前から帯域距離積（Bandwidth Distance Product）を指標とした長距離 TCP 高速化が盛んに研究開発されている。その中で東京大学 Data Reservoir [2] は 2003 年 11 月にアメリカ、フェニックスで開催された Super Computing 2003（SC2003）で、フェニックスと東京を接続した 24000Km（大平洋 1.5 往復）、合計 8.2Gbps のネットワークを用い、システム全体で 7Gbps を越える iSCSI ディスク間転送を実現した [3] [1]。

Data Reservoir では高速転送を実現するために複数の技術を開発したが、そのひとつが「Comet TCP」である。本論文は Comet TCP に焦点をあて、Comet TCP の実用化をめざして開発した長距離 TCP 高速化ゲートウェイ装置「Comet TCP Accelerator」の実装と評価結果を示す。Comet TCP Accelerator は帯域 800Mbps、RTT 200ms の長距離回線を用い、120 秒のセッション継続時間で 700Mbps 以上の単一 TCP セッション性能を実現可能である。

[1] 本研究の一部は文部科学省科学技術振興調整費先導的研究基盤整備「科学技術研究向け超高速ネットワーク基盤整備」によって行った。

2. 長距離 TCP 高速化

2.1. 長距離 TCP の問題

TCP は多様なリンクや装置の集合体であるインターネットで、より多くの通信が相互に調和しつつ動作することを目指して設計されたプロトコルである。TCP において Window サイズは受信側からの到達確認応答（ACK: Acknowledgement）なしに送信するデータサイズである。送信者は受信者からの ACK 受信に従って Window サイズを増やしていく（Slow Start）が、パケット喪失を検出すると Window サイズを半分に落とし、通信開始時よりもさらにゆっくりと Window サイズを増加させる（Congestion Avoidance）。このメカニズムはパケット喪失や応答遅延をネットワーク輻輳や通信相手負荷増大の兆しとみなし、Window サイズを制御することで輻輳状態からの回復を試みる自動調整機構であるが、ネットワーク遅延に比例して ACK の応答遅延が増加すると輻輳制御が効きすぎるため性能低下を招く。解析[4,5]も行われているが複雑な問題である。なお、長距離 TCP でもセッション継続時間を長くすればピーク性能は向上する。しかし、利用者にとっては通信開始から終了までの平均性能が重要なので、セッション継続時間を長くとった時の「ピーク TCP 性能」とセッション継続時間を制限した「短時間 TCP 性能」を区別すべきである。本論文は基本的に短時間 TCP 性能を重視する。

2.2. 長距離 TCP 性能

長距離 TCP 性能を把握することを目的として、通信遅延およびパケットロス率と短時間 TCP 性能の関係を実験環境で定量評価した（図 1）。

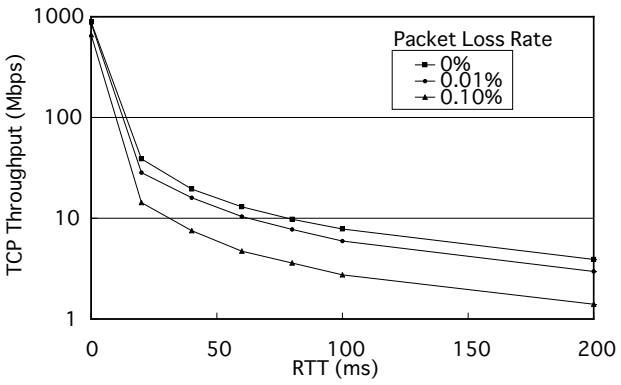

図 1 長距離 TCP の性能

実験環境では計算機性能が制限にならないよう十分高速な IA64 サーバおよびオンボードの Gigabit Ethernet

を用い、RedHat Linux 9 を標準のままで使用した。回線遅延装置としては自作の Comet Delay を使用した。Comet Delay は遅延を 1ms 単位、パケットロスを 0.01%単位で設定可能で、1000Base-T ネットワークに透過的に挿入して回線遅延を制御することができる。Comet Delay を用いて得た結果は実際の日米回線での実験結果と符合しており、実験環境として評価に耐えると考えている。TCP 性能の測定には iperf[6]を用い、セッション継続時間を 120 秒とした。一般に OS の各種パラメータをチューニングすることで TCP 性能をある程度向上させることができるが、120 秒短時間 TCP 性能ではほとんど差を生じない。

評価結果によれば、日本とアメリカ東海岸（RTT 200ms 程度）で TCP 性能は 4Mbps 程度、日本国内であっても東京と大阪（RTT 20ms 程度）で 40Mbps 程度しか出ないことがわかる。数十分程度の非常に長い時間をかければピーク TCP 性能をのばすことが可能だが、短時間では性能を発揮することができない。さらにパケット喪失があると Window サイズが半減する上に Congestion Avoidance にはいるため帯域の成長は期待できない。一般に輻輳のない専用インターネットであっても一定の確率でパケット喪失する可能性は残るため、TCP を用いて長距離で単一セッションの性能を向上させるのには限界があると考えられる。

地球規模の超高速インターネットで、単一のアプリケーションが 1Gbps 以上の帯域を独占的に使って通信できる状況では TCP の協調的な動作がかえって制限となっているといえよう。

2.3. 長距離 TCP 高速化技術の動向

長距離 TCP の性能向上を目的とした様々な方式が提案、実験されている。

まず考えられるのは TCP 通信を並列使用して性能を出す方法である。Data Reservoir はディスク転送を複数の iSCSI 通信に分割し、並列に動作させることで総合性能を向上させた。SC2003 では単一ファイルシステムの転送に 128 セッションの並列 TCP を用い、セッション間のトラフィック平均化技術[7]、パケット送出におけるパケット間ギャップ制御の技術[8]によって、20 分程度の短時間で合計 7Gbps 近いピーク転送性能を達成している（図 2）。

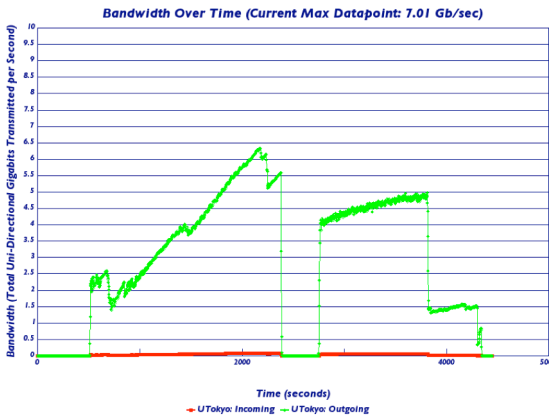

図 2　並列 TCP の転送性能（SC2003 公式結果[9]）

次に考えられるのは抜本的に単一 TCP セッションの性能向上を狙う方法で、大別して TCP プロトコルの精神を守りつつ改善する方法と全く異なる高速高信頼プロトコルに切り替える方法がある。

第一の方法は TCP の Window 制御を改善し、Window サイズの成長を早くするものである。FAST TCP[10]、High speed TCP[11]、Scalable TCP[12]などはこの範疇に属する。これらのアプローチは従来の TCP と同じく、他の通信との輻輳に対して共存が可能である。一方、性能の改善も限定的で、遅延が大きくなると性能が低下するという性質も従来の TCP と同じままである。

第二の方法は TCP に束縛されない新しい長距離高性能通信プロトコルを新規開発する方法で、IETF で Performance Enhancing Proxy（PEP）[13]として関連技術の調査結果が報告されている。

新たなプロトコルを用いて高速化を実現する場合、新プロトコルは TCP と相互接続性をもたないので、アプリケーションの TCP 通信を新プロトコルにつなぎ込む必要が生じる。PEP 方式では長距離インターネットの両端に特殊なゲートウェイを設け、端点の TCP をゲートウェイで終端し、ゲートウェイ間は新プロトコルを用いて通信する。本論の Comet TCP は PEP 方式で高速化を実現している。

現在市販されている長距離 TCP 高速化装置である NetEX HyperIP[14]、Mentat Sky-X[15]等はゲートウェイ装置間を独自のプロトコルで通信するという意味で PEP 方式を採用していると考えられる。

具体例として SC2003 における Data Reservoir の Comet TCP 使用性能を示す（図 3）。

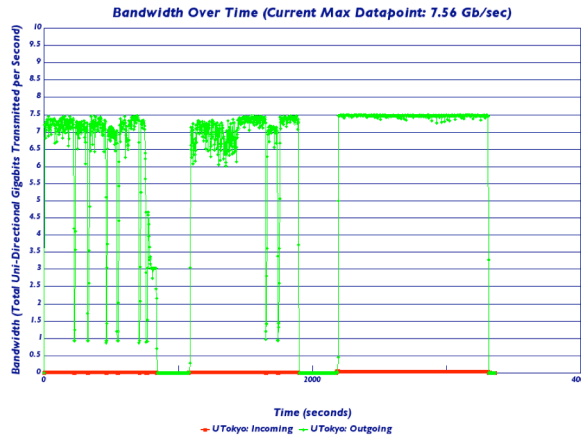

図 3　Comet TCP の転送性能（SC2003 公式結果 [9]）

TCP でないため当然ではあるが、立ち上がりのよい通信を実現可能で TCP アプリケーションからみた短時間 TCP 性能を向上させる効果があることがわかる。計測の前半は使用ネットワークで経路のフラップがあり、大量のパケットロスが発生したが、迅速に復旧している。

PEP 方式では TCP にとらわれず長距離高信頼通信に適したゲートウェイ装置間プロトコルを採用できるため、高速化が容易である反面、他の通信と輻輳した場合に脅威となりうる点が問題である。

3.　Comet TCP 方式

3.1.　基本方式

Comet TCP は帯域保証回線での使用を想定し、PEP 方式によって TCP アプリケーション間の長距離 TCP 性能を向上させる方式である（図 3）。

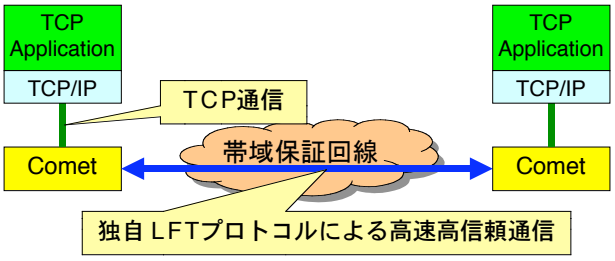

図 4　Comet TCP 方式

Comet は TCP アプリケーションの通信パケットを途中で TCP アプリケーションからみて透過的に終端し、独自プロトコルで遠隔の Comet と通信する。遠隔 Comet は独自プロトコルから TCP への再乗り換えを行い、あたかも TCP アプリケーション同士が直接通信しているようにみせる。すなわち Comet TCP 方式では TCP アプリケーションは TCP を用いていると信じて通信を行う。TCP アプリケーションと Comet の間はネットワーク的に近く遅延が小さいので TCP 性能は高速となる。長距離ネットワークの両側に配置された Comet 間は独自プロトコルである LFT（Long Fat Tunnel）プロトコルによって高速高信頼通信を実現する。LFT は性能を重視して帯域保証のあるネットワークを前提とし、指定帯域を活用して最大限のスループットを得る。

他の通信との競合や帯域の指定間違いなど、指定された帯域分の通信ができない場合もあるが、Comet TCP はネットワーク状況に応じて使用帯域を調整し輻輳を回避するようになっている。

Comet TCP の実現で鍵となるのは、TCP と LFT の中継、LFT による高速高信頼通信である。中継に関しては、Comet TCP の全ての機能を Comet i-NIC（Intelligent Network Interface Card）とデバイスドライバで実装することで様々な適用形態に対応できるようにした。

3.2.　中継モデル

TCP を中継するには適用する状況によって複数の方法が考えられる。今回、Comet TCP を実装するにあたって、以下の 3 つのモデルを考えた。Data Reservoir のようにホスト計算機を対向で接続する Host model、中継ゲートウェイによって接続することで複数のホスト計算機間の TCP を同時にサポートするものとして、アプリケーションゲートウェイで中継する Proxy model と IP ルーティングで中継する Router model である。

Host model は Comet TCP を使った通信をするホストに直接 Comet i-NIC を組み込むもので、ネットワークの変更を必要としないのが利点であるが、ホストの OS に対応したドライバが必要となる欠点がある。実際の利用に当たっては、Linux マシンの PCI バスに Comet i-NIC を装着し、デバイスドライバを組込むだけで Comet TCP を使用することができる。アプリケーション、TCP 他のシステムソフトウェアを一切変更する必要はない。

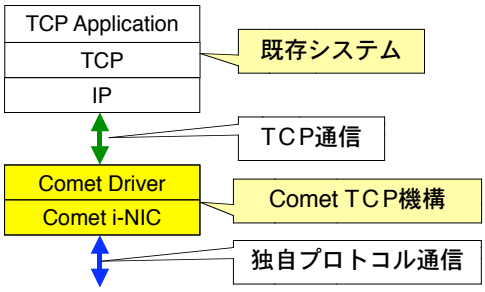

図 5　Host model

実際 Data Reservoir においてもソフトウェアを全く変更することなく、ネットワークインタフェースを切り替えるだけで Comet TCP／通常 TCP を使い分けることができた。

Proxy model は遠距離ネットワークの両側に配置されたゲートウェイでアプリケーションゲートウェイを使用して通信を中継するもので、アプリケーションは通常の TCP 通信でアプリケーションゲートウェイに接続し、アプリケーションゲートウェイ間が Comet TCP により高速化される。アプリケーションゲートウェイとしては一般の proxy ソフトウェアが使用可能である（図 6）。

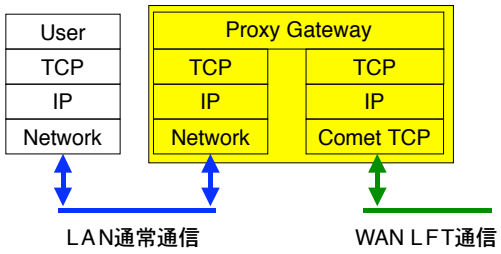

図 6　Proxy model

反対側には同じ構成の Comet TCP Accelerator があり、再度 Proxy を行うことでユーザの通信を中継する。Proxy を行うアプリケーションゲートウェイは delegated[16,17]などの Proxy プログラムをそのまま使用可能である。

Proxy model は http proxy には非常に親和性があり、例えば日本からアメリカの Web を高速アクセスするような場合、効果を発揮する。またデータをアプリケーション層で処理するため、データを圧縮／伸長するなど特定の加工を行う処理や Fiber Channel や SCSI など非 IP 通信を TCP 中継する処理を容易に実現できる。問題としては、中継するポートの全てにアプリケーションゲートウェイが必要である点、複数の計算機を中継する場合にはアプリケーションゲートウェイに ftp などプロトコル依存の処理が必要となる点、Comet TCP の機構だけでは任意の計算機の間の通信を中継するのが困難な点がある。特に最後は問題で、http など一部のプロトコルを中継する場合を除いて、任意の計算機を中継するためには Comet TCP 間でエンド・エンドのソース・デスティネーションアドレスを交換しなければならない。

Router model は Comet TCP が IP 層に位置して透過的に TCP を処理するという特性を生かして、ゲートウェイを IP ルータとして動作させ、Comet TCP で TCP 高速化処理を行う方式である（図 7）。

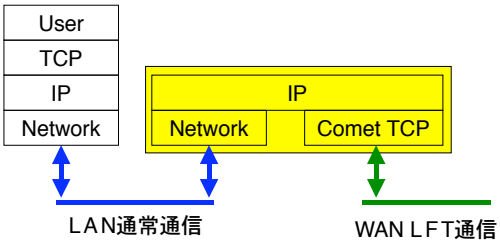

図 7　Router model

Comet TCP からみれば同じマシンの上位層からのパケットでも他のマシンからのパケットが IP 層でルーティングされたパケットでも違いはないため、Comet TCP 自体に特別な変更を加えることなく Proxy model と同様に Router model を実現可能である。

Router model ではエンド・エンドのパケットがそのまま流れるため、Proxy model と異なり任意の通信に対処可能である。またゲートウェイ装置としてもアプリケーションゲートウェイを動作させる必要がないためプロセッサ負荷が小さい。問題としては IP 層での処理であるため、Proxy model で可能なデータの内容に依存した処理は不可能であること、Fibre Channel など非 IP ネットワークの中継には対応できないことがあげられる。

3.3.　帯域制限制御

ネットワークで必ずパケットロスが発生する以上、信頼性のある通信を実現するためには本質的に ACK による到達確認が必須と考えられる。TCP は ACK がなくても送り出せるデータ量を Window サイズによって管理しているが、この方法では性能がのびないことはすでに議論したとおりである。そこで Comet TCP では Window サイズを常に十分大きくとるかわり、使用可能帯域を規定し、その帯域におさまる範囲で最大のスループットを実現する方式を採用した。言い換える

と「Window サイズ制御」から「帯域制限制御」への転換を行った。

TCP は ACK 受信によって利用可能な帯域を探りつつ流量制限を行うのに対して Comet TCP は規定の帯域を信用して帯域の範囲内で「積極的な通信」を行う。ここで「積極的な通信」とは、Window サイズを制限せず、規定帯域の範囲でパケットを冗送する（同じパケットを複数送信する）ことである。規定帯域は専用回線の場合はあらかじめ設定してもよいし、パケットロス率を交換して自動調整することも可能である。

3.4. LFT プロトコル

「帯域制限制御」を実現するためのプロトコルとして LFT を開発した。LFT は IPSec ESP トンネルモードを用いた Point to Point Tunnel プロトコルで、現在の実装は IPv4 のみ、暗号あり／なしに対応する。再送方式はごく普通の Go Back N 方式を採用した。Negative Acknowledgement、Selective Acknowledgement は採用していない。

Comet TCP の LFT 実装において、ある時点で送信すべきパケットは「本来送信すべきデータパケット」、「冗送パケット」、「LFT 再送パケット」、「LFT ACK パケット」の四種類である。Comet TCP はこれらのパケットに優先順位をつけ、指定された帯域の範囲内で送信する。例えば「本来送信すべきデータパケット」、「LFT 再送パケット」の両方がない場合は ACK されていないデータを「冗送パケット」として繰り返し送信する。これによって受信側が必要なパケットを得る確率は向上する。一見「冗送」はトラフィックとして「無駄」であるが、余っている利用可能帯域を有効活用するという意味では無駄ではない。むしろ余りを使わないほうが無駄ということもできる。重要なのは帯域制限（Shaping）を正確かつ効率的に行うことである。

当然ながらこのような「積極的な通信」は TCP など控えめな通信者にとっては脅威となりうる。Comet TCP の実装では精度の高い帯域制御を行うことが必須である。

LFT は IPSec ESP[18]の些細な変更で実装した。LFT の制御を行うため ESP のシーケンス番号フィールド（32bit）を制御コードとして使用した。LFT の再送制御コマンド、シーケンス番号、ACK 番号に用いる。LFT シーケンス番号は 24bit である。

単に高速高信頼通信を独自プロトコルで実現しようと考えるならば、IP、UDP などを用いればよい。しかし LFT の設計においては敢えて IPSec に準拠した VPN トンネル技術を採用した。これは、iSCSI で Single DES による暗号化を規定しているように、今後の高速遠距離通信においてセキュリティが重要であるという基本認識に基づく。IPSec に準拠することでデータ暗号化や AH（Authentication Header）認証によってパケットの盗聴、改竄、リプレイ攻撃、SYN 攻撃、RST 攻撃に自然に対処可能となる。但し現在の実装では Comet i-NIC のハードウェア的制限から認証には対応していない。問題は ESP カプセリングのため MTU が小さくなる点であるが、TCP 性能上の顕著な影響はなかった。

3.5. 実装

Comet TCP はネットワークプロセッサ Comet NP を搭載した Comet i-NIC[19]を用いて実装した。図 8 に Comet i-NIC の外形を、図 9 に構成を示す。

図 8 Comet i-NIC

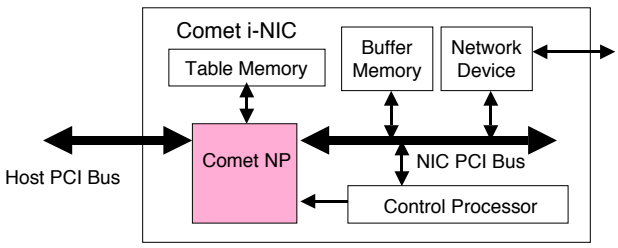

図 9 Comet i-NIC の構成

Comet i-NIC は PCI（Peripheral Component Interconnect）規格準拠のボードで Comet NP を搭載した[20]。PCI Bus は Host PCI Bus、NIC PCI Bus 共に 64/32bit、66/33MHz で、NIC PCI Bus に 1000Base-T を 2 port、256MB の大容量バッファメモリを設けた。

NIC 内部の制御は Control Processor （CP、Intel PXA255、400MHz）が行う。

Comet TCP のパケットフローを図 10 に示す。パケット解析、IP パケットと LFT（IPSec ESP）パケットの変換、LFT ACK パケットの生成を Comet NP などハードウェアで実装した。CP は帯域制御、バッファ管理、ハードウェアの制御を担当する。Comet TCP ではほとんど全てのパケット処理を実時間処理することで処理能力を向上させた。CP は 20μs 周期で処理を行っており、ネットワーク側、ホスト側の両方に対して 1Mbps 単位で帯域制御を行うことができる。

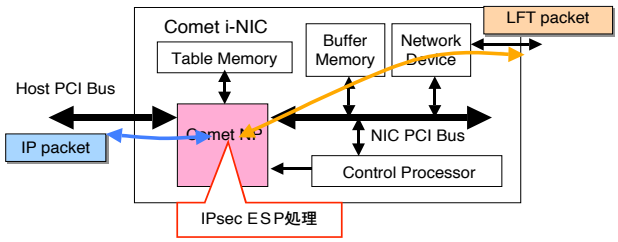

図 10　Comet TCP のパケットの流れ

3.6.　Comet TCP 基本性能

実験環境で測定した Comet TCP の性能を示す。Comet TCP の測定は IA サーバ（Xeon 2.4GHz）及び RedHat Linux 9.0 に Comet i-NIC およびドライバをインストールした装置で行った。ネットワークは 1000Base-T を用い、Comet Delay で回線遅延を実現した。

図 11 に Comet TCP と他方式の性能比較を示す。

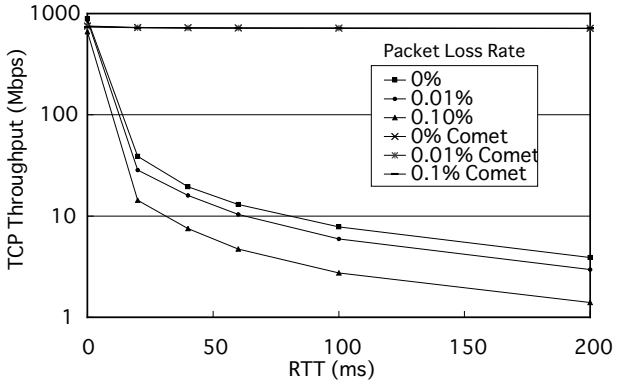

図 11　Comet TCP 性能

Comet TCP はパケットロス率にかかわらず RTT 200ms でも 700Mbps 以上の TCP 性能を実現する。なお遅延が小さい場合は Comet TCP のほうが性能は低い。これは LFT 制御のオーバヘッドのためである。

図 12 に Comet TCP の詳細性能を示す。

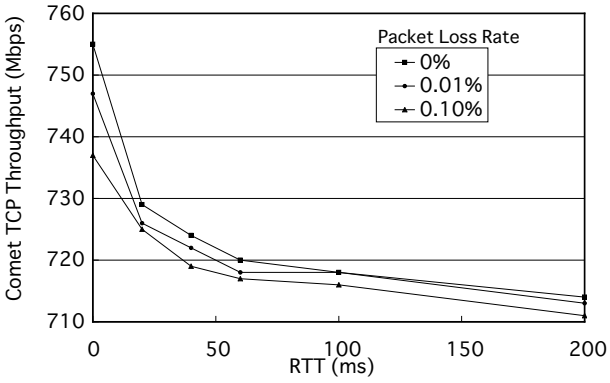

図 12　パケットロス時の Comet TCP 性能

パケットロス率が 0.1%でも RTT 200ms で 710Mbps の性能を実現可能なことがわかる。日本とアメリカ西海岸までの RTT は 140ms 程度、東海岸で 200ms 程度であるから、日米回線を用いて単一セッション 700Mbps の TCP 通信が可能な性能といえる。またこの場合の LFT 最大使用帯域（ネットワーク上のトラフィック）は 800Mbps であった。

すでに述べたように Comet TCP は TCP/IPSec 可能である。図 13 に 3DES CBC の TCP/IPSec 性能を示す。

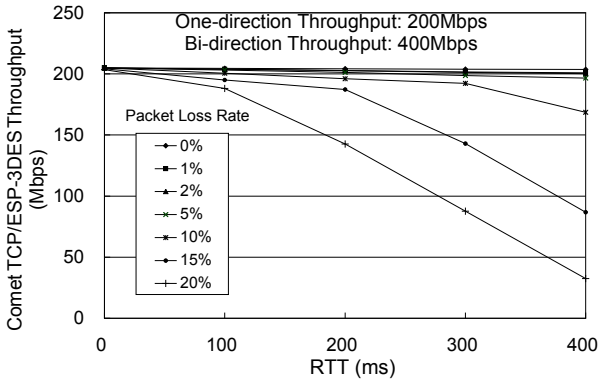

図 13　パケットロス時の Comet TCP 3DES 性能

Comet NP は 66MHz 動作で片方向 200Mbps、両方向 400Mbps の IPSec ESP 処理性能のため、これが制限となるが、遅延 200ms まで 200Mbps／400Mbps の TCP 性能を実現可能なことがわかる。絶対性能が 200Mbps と低いため、許容できる遅延、パケットロス率は大きくなる。回線帯域に制限がない場合、遅延 200ms、パケットロス率 10%でも 170Mbps の TCP/IPSec 性能を達成することがわかる。

4.　Comet TCP Accelerator

4.1.　アプライアンス機器

Data Reservoir では Host model によってサーバに

Comet TCP を組込んだ。しかし一般に長距離 TCP 高速化を必要とする Disaster Recovery System などの利用環境を考えると Comet i-NIC をユーザのサーバに組込めない場合や複数のサーバから一本の WAN 回線を共用しなければならない場合が多い。

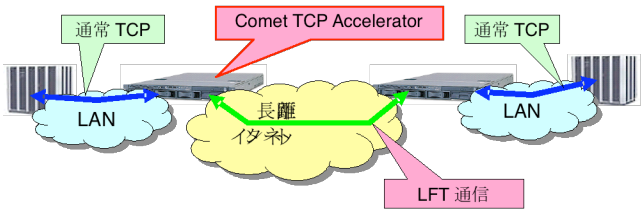

図 14　Comet TCP Accelerator 使用形態

より汎用的に Comet TCP を利用するためには Proxy model、Router model アプライアンス機器の形態での Comet TCP ゲートウェイ装置が便利である。そこで Comet TCP Accelerator を開発した。

4.2.　試作機

Comet TCP は PCI ボードなので Comet i-NIC を装着可能であればどのような PC でも使用可能である。また Comet i-NIC の性能は同じなので Comet TCP Accelerator の性能はプラットフォームマシンの性能に依存する。そこでサーバ CPU 性能の異なる Comet TCP Accelerator を試作した（表 1）。

表- 1　試作機一覧

モデル	CPU/Chipset	RAM	PCI/LAN
Small Box	MediaGX 300MHz Cyrix CS5530A	128MB	33-32 100B-TX
Barebone Box	Pen3 1GHz Intel 815E, ICH2X	512MB	33-32 100B-TX
PICMG Box	Pen4 2.4GHz ServerWorks GC-SL	1GB	66-64 1000B-T
IA Server	Xeon 2.4GHz Intel E7501	2GB	66-64 1000B-T

性能を左右する第一の条件はメインメモリと PCI デバイス PCI 転送性能である。これにはチップセットの性能が影響する。第二の条件はプロセッサ性能である。高速であるに越したことはないが、アプリケーションゲートウェイや IP Routing 程度の処理では PCI 転送性能のほうがむしろ問題になる場合が多い。Dual Processor も試験したが、RedHat Linux 9 SMP カーネルではデバイスドライバが特定の CPU で動作するためロックオーバヘッド分、かえって性能が低下した。表 2 に試作機の性能を示す。Routing 性能は Agilent Technology 社の Router Tester で測定し、パケット長が短い場合（Min）、長い場合（Max）を示した。

表- 2　性能評価結果（Mbps）

モデル	Proxy	Router	
		Routing Min/Max	TCP
Small Box	50	11/56	50
Barebone Box	90	20/97	90
PICMG Box	740	40/720	755
IA Server	740	42/770	755

Router としてみると短パケットのフォワーディング性能が低い点が問題となるが、TCP Accelerator としてみると低能力の Small Box でも 50Mbps の能力をだしており、Comet i-NIC で処理している効果が現れている。衛星回線のように比較的性能が低く遅延の大きなネットワークでは Small Box 程度で十分と考えられる。FTTH などでは Barebone Box が適する。Small Box、Barebone Box では 3DES による暗号化を行っても回線帯域を 100%使用可能である。一例として PICMG Box の仕様を表 3 に示す。

表- 3　PICMG Box 仕様

	仕様
CPU	Pentium 4 2.4GHz
PCI	PCI 64bit/66MHz x 2
メモリ	1GB
ディスク	3.5インチIDE／Flash
ネットワーク	10/100/1000Base-T x 2
最大WAN帯域	800Mbps
WAN帯域制限	1-800Mbps（1Mbps刻み）
TCP中継性能	700Mbps以上（RTT 200ms）
TCP同時中継数	1000
RAS	通信状態監視、異常時の自動停止
制御	Webインタフェース
装置サイズ	482 x 44 x 500 mm
重量	8.0 Kg
電源	200W

ここで TCP 同時中継数はあくまで同時に扱えるセッション数を示すだけで、中継数が増えるとセッションあたりの性能は低下する。仮に 1000 セッションを中継すると一セッションあたりの TCP 性能は 700kbps（700Mbps/1000）になる。

なお、短パケットで性能が低いのはホスト計算機と Comet i-NIC のデータ転送制御のオーバヘッド（パケット毎に固定）によるもので、これは Comet i-NIC の最大の問題点である。

5. おわりに

Data Reservoir の高速化を出発点として長距離 TCP の高速化技術を検討し、Comet TCP を開発した。Comet TCP を PCI ネットワークインタフェースカードである Comet i-NIC を用いて実装することで、Linux マシンをベースとした Host model、Proxy model、Router model の長距離 TCP 高速化機構を容易に構築できる。独自の高速高信頼プロトコルとして LFT（Long Fat Tunnel）を開発し、Window サイズ固定の帯域制御によって輻輳制御を行う方式を実装した。

Comet TCP は帯域 800Mbps、RTT 200ms のネットワークで 700Mbps 以上の 120 秒短時間 TCP 性能を達成した。Comet TCP を Proxy model、Router model でアプライアンス装置とした Comet TCP Accelerator はプラットフォームを適切に選択することで、低速回線から高速回線までの幅広い応用分野を全く同じアーキテクチャを用いて最適なコストと性能でサポート可能である。

今後の課題としては、まず Comet TCP、特に LFT の解析と改善がある。Comet TCP は性能を高速化することを一義に開発したが、比較的初期からハードウェア性能の限界に達したためアルゴリズムを詳細に練ったとはいいがたい。現在の再送方式、Acknowledgement 方式を詳細に解析し、改善を行う余地は大きい。

Comet TCP の性能限界は主として Comet i-NIC にある。これを解決するため現在 Comet NP チップの後継である TNP（Trusted Network Processor）チップを開発中である。

謝辞

Data Reservoir システムへの Comet TCP 組込み、SC2003 の実験に関して東京大学情報理工学研究科玉造潤史氏、東京大学情報基盤センター加藤朗氏に感謝する。

Comet TCP の開発では以下の方々に非常な協力をいただいた。富士通コンピュータテクノロジーズ株式会社（CTEC）の都筑俊秀氏は Comet i-NIC のハードウェア実装を行った。CTEC の長沼征典氏は Comet NP ファームウェア実装を行った。株式会社トライテックの的場宏純氏は Comet i-NIC ファームウェアならびに Linux 用 Comet TCP ドライバの実装を行った。これらの実装なくして Comet TCP の開発はできなかったことを明記し、ここに深く感謝する。

参考文献

[1] J. Postel: Transmission Control Protocol, RFC 793, Sep 1981

[2] http://data-reservoir.adm.s.u-tokyo.ac.jp/

[3] Hiroyuki Kamezawa, Makoto Nakamura, Junji Tamatsukuri, Nao Aoshima, Mary Inaba, and Kei Hiraki: Inter-layer coordination for parallel TCP streams on Long Fat pipe Networks, SC2004, Nov 2004

[4] Tsuyoshi Ito and Mary Inaba: Theoretical Analysis of Performances of TCP/IP Congestion Control Algorithm with Different Distances, Networking 2004, May 2004

[5] S. Floyd, and K. Fall: Promoting the Use of End-to-End Congestion Control in the Internet, IEEE/ACM Transactions on Networking, August 1999

[6] http://dast.nlanr.net/Projects/Iperf/

[7] Hiroyuki Kamezawa, Makoto Nakamura, Mary Inaba, and Kei Hiraki: Coordination between parallel TCP streams on Long Fat Pipe Network, DPSN, 2004

[8] Makoto Nakamura, Junsuke Sembon, Yutaka Sugawara, Tsuyoshi Itoh, Mary Inaba and Kei Hiraki: End-node transmission rate control kind to intermediate routers - towards 10 Gbps era, PFLDnet, 2004

[9] http://scinet.supercomp.org/2003/bwc/results/

[10] C. Jin, D. X. Wei, S. H. Low: FAST TCP for high-speed long-distance networks, Internet Draft http://netlab.caltech.edu/FAST/, Jun 2003

[11] S. Floyd: HighSpeed TCP for Large Congestion Windows, RFC 3649, December 2003

[12] T. Kelly: Scalable TCP: Improving Performance in HighSpeed Wide Area Networks, First International Workshop on Protocols for Fast Long-Distance Networks, 2003

[13] J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby: Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations, RFC 3135, June 2001

[14] http://www.netex.com/

[15] http://www.mentat.com/skyx/skyx-gateway.html

[16] Yutaka Sato: DeleGate: A General Purpose Application Level Gateway, WWCA 97

[17] http://www.delegate.org/delegate/

[18] S. Kent: IP Encapsulating Security Payload (ESP), RFC 2406, Nov 1998

[19] http://www.comet-can.jp/

[20] 下國治、河合純、陣崎明、山澤昌夫、中村修、村井純： "Security Network Processor による低消

費電力 IPSec ESP の実装と評価"、インターネッ
トコンファレンス 2003、2003 年 10 月