通信品質を向上させるネットワークアプライアンス
に関する研究

磯部　隆史

システム情報工学研究科

筑波大学

２０１５年　３月

研究概要

情報システムの複雑化や大規模化に伴う ICT (Information and Communication Technology) 機器の所有・管理コストの増加を背景に、ネットワーク越しに ICT サービスを安価に提供するクラウドコンピューティング(以下、クラウド)が普及し、それに伴いネットワークアプライアンスの重要性が増加している。

クラウドは、ネットワーク装置，サーバ，ストレージなどの ICT 機器を集約して、複数のユーザ、或いは、同一ユーザ内の複数の拠点で共有することで、所有・管理コストを削減する[1]。ICT 機器のネットワーク装置，サーバ，ストレージなどのリソースを、仮想化技術などを用いて論理的に作成・分割・削除することで、ユーザが必要なときに必要なだけ利用する形態を可能とする。ネットワーク装置は VLAN (Virtual Local Area Network)単位で、サーバは仮想マシン(Virtual Machine: VM)単位で、ストレージは論理パーティション単位で、それぞれリソースを分割する。

クラウドは、共有形態に応じて、パブリッククラウドとプライベートクラウドに分類できる[2][3]。パブリッククラウドは、料金を払うことで誰もが利用でき、複数の異なるユーザが共有する。米国企業による市場開拓が進んでおり、AWS (Amazon Web Service)などのサービスの提供が開始している。一方、プライベートクラウドは、単一の企業の所有であり、複数の拠点が共有する。主に、グローバル企業による構築が進んでおり、業務データの管理コスト削減のために利用する[4]。グローバル企業は、世界中に拠点を進出させていく際にプライベートクラウドを利用することで、遠隔設計拠点の CAD (Computer Aided Design) データや、遠隔工場の設備監視データなど、拠点毎に管理していた業務データを 1 つのデータセンタに集約して共有することが可能となる。

ユーザや拠点毎に切り分けられたクラウドのリソースは、遠距離にある拠点や端末から、インターネットや仮想専用線や無線網などの広域通信網(Wide Area Network: WAN)の通信回線を経由して、主に TCP (Transmission Control Protocl)を用いて通信することで利用する。そのため、クラウドへのネットワーク経由での安全なアクセスをサポートするセキュリティ系のネットワークアプライアンスと、クラウドサービスを利用する際の速度を向上させる高速化系のネットワークアプライアンスの重要性が増加している。

セキュリティ系のネットワークアプライアンスは、クラウド利用の通信がインターネットのような公衆網を経由する場合に、クラウドが管理する ICT 機器のリソースやデータを

外部の攻撃から守るために用いる。まず、IDS (Intrusion Detection System)、IPS (Intrusion Prevention System)、DPI (Deep Packet Inspection)、ファイアウォールなどのネットワークアプラアインスを用いることで、通信の監視を行い異常な通信を検出して排除する。更に、NAT (Network Address Translation) / NAPT (Network Address Port Translation)等のネットワークアプラアインスを用いて、クラウド内の端末に割り当てたプライベート IP (Internet Protocol)アドレスとパブリック IP アドレスの間で変換を行い、クラウド内の端末と外部端末との直接的な通信を防ぎ、データセンタ内部の ICT 機器の IP アドレスやポート番号を隠蔽することで、セキュリティを確保する。

高速化系のネットワークアプライアンスは、通信データの転送速度を向上させる通信高速化系のネットワークアプライアンスと、クラウドへのアクセス時に用いるアプリケーションの応答速度を向上させるクラウドサービス高速化系のネットワークアプライアンスに分類できる。

通信高速化系のネットワークアプライアンスは、クラウド利用の通信が、通信遅延とパケット廃棄率の大きな長距離通信回線や、ビットエラーによるパケット廃棄が発生しやすい無線回線を経由する場合に、通信遅延増大やパケット廃棄に伴う TCP 通信のスループット劣化を防ぐために用いる。まず、TCP 高速化機能を持つネットワークアプライアンスを用いることで、ネットワークの回線帯域の利用効率を向上させてクラウドサービス利用時のデータ転送速度を向上させる。また、新興国などの通信インフラの整備が進んでいない地域では、広帯域な回線を契約することが難しく、差分転送や圧縮などの機能を持つネットワークアプライアンスを用いて通信データ量を削減することで、回線帯域を仮想的に向上させて、クラウドサービス利用時のデータ転送速度を向上させる。

クラウドサービス高速化系のネットワークアプライアンスは、遠隔地の多数の端末やセンサデバイス(温度や位置情報を取得する装置、設備監視装置、監視カメラ、車載端末、等)がクラウド利用の通信を行う場合に、クラウドが提供するサービスを肩代わりして、クラウドサービスの応答速度を向上させるために用いる。まず、クエリトランザクションや HTML (HyperText Markup Language) / XML (Extensible Markup Language)テキスト解析・変換を高速化するアプライアンスを用いて、遠隔地の端末が発行する SQL (Structured Query Language)コマンドにより DB (Data Base)を操作するクエリトランザクション処理や、センサデバイスからの XML データの解析速度を向上させて、クラウドサービス利用時の応答速度を向上させる。また、キャッシュ機能を持つアプライアンスを用いて、遠隔地の端末から

のデータ要求に素早く応える。更に、SSL (Secure Sockets Layer) - VPN (Virtual Private Network)、IPSec (Internet Protocol Security)などの暗号通信を高速化するネットワークアプライアンスを用いて、クラウドとセンサデバイスの間でやり取りする個人・業務データの機密を守る暗号通信を高速化する。

このように、クラウドの ICT 機器のリソースを、遠隔地から通信回線を経由して利用する際には、安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させるために、以下のネットワークアプライアンス①～③を用いる。

①セキュリティ系のネットワークアプライアンス
②通信高速化系のネットワークアプライアンス
③クラウドサービス高速化系のネットワークアプライアンス

しかしながら、クラウドを利用する際に用いるネットワークアプライアンスには、様々な課題があった。

セキュリティ系のネットワークアプライアンス①は、多数ユーザを収容するクラウドや、クラウドが接続するネットワークを防御するための高スループットを実現しつつ、日々進化する多様な攻撃手法に対応しなければならないという課題があった。通信高速化系のネットワークアプライアンス②は、多様なアプリケーションの通信に対応する高い汎用性を実現しつつ、通常の固定式の PC (Personal Computer)端末だけではなく、スマートフォンやタブレット PC などのモバイル端末や､設備監視装置や車載端末などのセンサデバイスなど、多様な端末に対する通信高速化をサポートしなければならないという課題があった。特に、モバイル端末やセンサデバイス向けの通信高速化をサポートするためには、ビットエラーによるパケット廃棄により通信速度が低下しやすいラストワンマイルの無線回線区間を高速化せねばならないという課題があった。クラウドサービス高速化系のネットワークアプライアンス③は、遠隔地からクラウドにアクセスする際の往復通信時間が長く、遠隔地の端末からコマンド要求を送信してクラウドからリプライ応答が帰ってくるまでのコマンド往復時間が長い場合に、サービスの応答時間を高速化しなければならないという課題があった。

このように、ネットワークアプライアンス①～③には、以下①～③の課題があった。

①高スループットを実現しつつ、日々進化する多様な攻撃手法に対応

②多様なアプリケーションと端末に対応する汎用性の高い通信高速化

③クラウドサービスを遠隔地から利用時のコマンド往復時間短縮および応答高速化

高スループットを実現しつつ、日々進化する多様な攻撃手法に対応せねばならない課題①を解決するためには、高速演算性能と柔軟な機能変更を可能とする汎用性を兼ね備えた装置を用いて、ユーザ要求やネットワーク状況に応じて対異常通信防御の機能を無瞬断で提供する必要性がある。また、多様なアプリケーションと端末に対応する汎用性の高い通信高速化をせねばならない課題②を解決するためには、様々なアプリケーションや端末が一般的に使用する TCP 通信のパケットフォーマットに変更を加えずに TCP 通信のまま高速化する必要性がある。加えて、モバイル端末やセンサデバイス向けの通信高速化をサポートするためには、無線区間への装置設置を要件としない通信高速化が必要となる。更に、クラウドサービスを遠隔地から利用する際のコマンド往復時間の短縮、および応答時間を高速化せねばならない課題③を解決するためには、多様な場所への設置を可能とする小型省電力で高性能な装置を、モバイル端末やセンサデバイスとの距離が近いネットワークエッジに設置して、ユーザ要求やネットワーク状況に応じてクラウドが提供する様々なサービスを肩代わりして代理で実行する必要性がある。

本研究では、これらの必要性に応えるため、以下(1)〜(3)の特徴を持つネットワークアプライアンスを開発し、クラウドデータセンタやネットワークエッジに設置することを提案した。

(1) 高スループットを実現しつつ、様々なセキュリティ機能を無瞬断で更新

セキュリティ系のネットワークアプライアンスのなかで、IDS (Intrusion Detecion System)や IPS (Intrusion Prevention System)などの対異常通信防御を行うネットワークアプライアンスは、高スループットを実現しつつ、日々進化する多様な攻撃手法に追従していく課題①の解決を最も必要とする。課題解決のためには、大量のパケットを高速に解析するための高い演算性能と、日々進化する新手の攻撃に対応できるように、対異常防御アルゴリズムを瞬時に更新できる柔軟性を両立させる必要性がある。本研究では、高速演算性能と柔軟性を備えた動的再構成 LSI を用いることで、高スループットを実現しつつ、DDoS (Distributed Denial of Service)，P2P (Peer to Peer)，Warm など

の異常通信を探知して防御する対異常通信防御の機能を無瞬断で更新する無瞬断更新方式を提案した。提案方式により、高スループットを実現したまま、パケット廃棄を起こすことなく対異常通信防御アルゴリズムを更新することを、シミュレーションおよび実験により確認した。

(2)多様なアプリケーションと端末に対応する TCP 通信のままでの汎用的な通信高速化

通信高速化系のネットワークアプライアンスのなかで、クラウドアクセスの際に広く利用される TCP 通信を高速化するネットワークアプライアンスは、多様なアプリケーションや端末に対応する汎用的な通信高速化の課題②の解決を最も必要とする。課題解決のためには、多様なアプリケーションや端末が一般的に使用する TCP 通信のパケットフォーマットに変更を加えずに、TCP 通信のままでの汎用的な通信高速化を実現する必要性がある。本研究では、TCP 通信のパケットフォーマットに準拠しつつ、パケット廃棄率の変化率に基づく独自の輻輳制御を行うことで回線帯域の利用効率を向上させる TCP 高速化方式を提案した。更に、モバイル端末やセンサデバイス向けの TCP 通信高速化をサポートするため、無線回線区間への装置設置を要件としない TCP 通信高速化方式を提案した。提案方式により、通信遅延とパケット廃棄率の大きな長距離通信回線を模擬したネットワークを経由する TCP 通信が高速化することを確認した。更に、無線ネットワークを経由するモバイル端末向けの TCP 通信が高速化することを確認した。

(3)ネットワークエッジにおいて様々なクラウドサービスを代行して応答を高速化

クラウドサービス高速化系のネットワークアプライアンスのなかで、遠隔地の端末が発行するコマンド要求に対してリプライ応答を返信する操作を複数回シーケンシャルに実行するクラウドサービスを高速化するネットワークアプライアンス、例えば、SQL 等を用いてデータベースを操作するクエリトランザクションを高速化するネットワークアプライアンスは、クラウドサービスを遠隔地から利用する際のコマンド往復時間短縮および応答高速化の課題③の解決を最も必要とする。課題解決のためには、多様な場所への設置を可能とする小型省電力で高性能な装置を、モバイル端末やセンサデバイスとの距離が近いネットワークエッジに設置して、ユーザ要求やネットワーク状況に応じてクラウドが提供する様々なサービスを肩代わりして代理で実行する必

要性がある。本研究では、ユーザ端末に近いネットワークエッジへの設置を可能とする小型省電力で高性能なネットワークアプライアンスを用いて、ユーザ端末の発行する様々な SQL クエリのコマンド要求に対して代理でリプライ応答を行い、クラウドサービスの一部を代行して応答を高速化する方式を提案した。提案方式を適用した試作機を用いた実験評価において、遠隔地からクラウドサービスを利用する際の応答時間を高速化できることを確認した。

本研究で開発した上記(1)〜(3)の特徴を持つネットワークアプライアンスを適宜選択して適用することで、クラウドサービスへの安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させることが可能となる。具体的には、特徴(1)を持つセキュリティ系のネットワークアプライアンスを適用することで、高スループットを実現しつつ、日々進化する多様な攻撃手法に対応することが可能となる。また、特徴(2)を持つ通信高速化系のネットワークアプライアンスを適用することで、多様なアプリケーションと端末に対応する汎用性の高い通信高速化が可能となる。更に、特徴(3)を持つクラウドサービス高速化系のネットワークアプライアンスを適用することで、クラウドサービスを遠隔地から利用する際のコマンド往復時間短縮および応答高速化が可能となる。

目次

第 1 章　序論

1.1　研究の背景

情報システムの複雑化や大規模化に伴う ICT (Information and Communication Technology) 機器の所有・管理コストの増加を背景に、ネットワーク越しに ICT サービスを安価に提供するクラウドコンピューティング(以下、クラウド)が普及し、それに伴いネットワークアプライアンスの重要性が増加している。

クラウドは、図 1 に示すように、ネットワーク装置，サーバ，ストレージなどの ICT 機器を集約して、複数のユーザ、或いは、同一ユーザ内の複数の拠点で共有することで、所有・管理コストを削減する[1]。ICT 機器のネットワーク装置，サーバ，ストレージなどのリソースを、仮想化技術などを用いて論理的に作成・分割・削除することで、ユーザが必要なときに必要なだけ利用することを可能とする。ネットワーク装置は VLAN 単位で、サーバは仮想マシン(Virtual Machine: VM)単位で、ストレージは論理パーティション単位で、それぞれリソースを分割する。

クラウドは、共有形態に応じて、パブリッククラウドとプライベートクラウドに分類できる[2][3]。パブリッククラウドは、料金を払うことで誰もが利用でき、複数の異なるユーザが共有する。米国企業による市場開拓が進んでおり、AWS (Amazon Web Service)などのサービスの提供が開始している。一方、プライベートクラウドは、単一の企業の所有であり、複数の拠点が共有する。主に、グローバル企業による構築が進んでおり、業務データの管理コスト削減のために利用する[4]。グローバル企業は、世界中に拠点を進出させていく際にプライベートクラウドを利用することで、遠隔設計拠点の CAD (Computer Aided Design) データや、遠隔工場の設備監視データなど、拠点毎に管理していた業務データを 1 つのデータセンタに集約して共有することが可能となる。

ユーザや拠点毎に切り分けられたクラウドのリソースは、遠隔地にある端末や拠点から、インターネットや仮想専用線や無線網などの広域通信網(Wide Area Network: WAN)の通信回線を経由して、主に TCP (Transmission Control Protocl)を用いて通信することで利用する。そのため、クラウドへのネットワーク経由での安全なアクセスをサポートするセキュリティ系のネットワークアプライアンスと、クラウドサービスを利用する際の速度を向上させる高速化系のネットワークアプライアンスの重要性が増加している。

セキュリティ系のネットワークアプライアンスは、クラウド利用の通信がインターネットのような公衆網を経由する場合に、クラウドが管理する ICT 機器のリソースやデータを外部の攻撃から守るために用いる。まず、IDS (Intrusion Detection System)、IPS (Intrusion Prevention System)、DPI (Deep Packet Inspection)、ファイアウォールなどのネットワークアプライアンスを用いることで、通信の監視を行い異常な通信を検出して排除する。更に、NAT (Network Address Translation) / NAPT (Network Address Port Translation)等のネットワークアプ

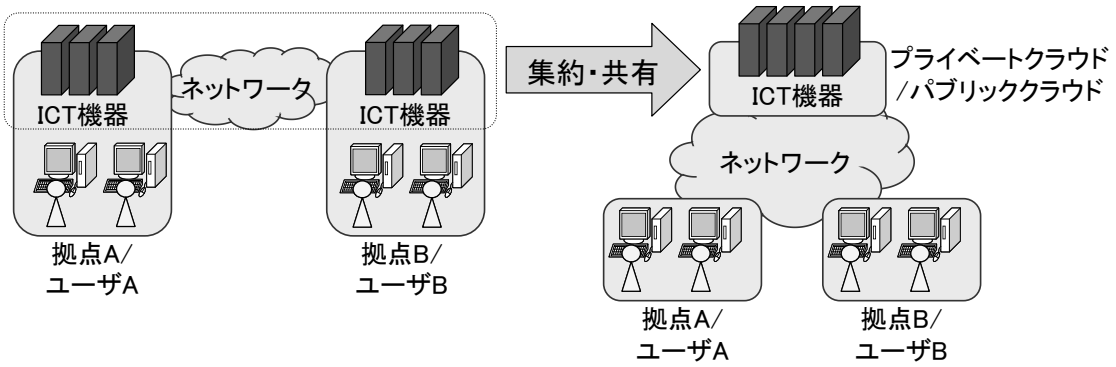


図 1　クラウドコンピューティングを用いた ICT 機器の管理コスト削減

ラアインスを用いて、クラウド内の端末に割り当てたプライベート IP (Internet Protocol)アドレスとパブリック IP アドレスの間で変換を行い、クラウド内の端末と外部端末との直接的な通信を防ぎ、データセンタ内部の ICT 機器の IP アドレスを隠蔽することで、セキュリティを確保する。

高速化系のネットワークアプライアンスは、通信データの転送速度を向上させる通信高速化系のネットワークアプライアンスと、クラウドへのアクセス時に用いるアプリケーションの応答速度を向上させるクラウドサービス高速化系のネットワークアプライアンスに分類できる。

通信高速化系のネットワークアプライアンスは、クラウド利用の通信が、通信遅延とパケット廃棄率の大きな長距離通信回線や、ビットエラーによるパケット廃棄が発生しやすい無線回線を経由する場合に、通信遅延増大やパケット廃棄に伴う TCP 通信のスループット劣化を防ぐために用いる。例えば、WAN アクセラレータ[5]などの通信高速化系のネットワークアプライアンスを用いる。まず、TCP 高速化機能を持つネットワークアプライアンスを用いることで、ネットワークの回線帯域の利用効率を向上させてクラウドサービス利用時のデータ転送速度を向上させる。また、新興国などの通信インフラの整備が進んでいない地域では、広帯域な回線を契約することが難しく、差分転送や圧縮などの機能を持つネットワークアプライアンスを用いて通信データ量を削減することで、回線帯域を仮想的に向上させて、クラウドサービス利用時のデータ転送速度を向上させる。

クラウドサービス高速化系のネットワークアプライアンスは、遠隔地の端末やセンサデバイス(温度や位置情報を取得する装置、設備監視装置、監視カメラ、車載端末、等)がクラウド利用の通信を行う場合に、クラウドが提供するサービスを肩代わりして、クラウドサービスの応答速度を向上させるために用いる[5]。まず、クエリトランザクションや HTML (HyperText Markup Language) / XML (Extensible Markup Language)テキスト解析・変換を高速化するアプライアンスを用いて、遠隔地の端末が発行する SQL (Structured Query Language) コマンドにより DB (Data Base)を操作するクエリトランザクション処理や、センサデバイス

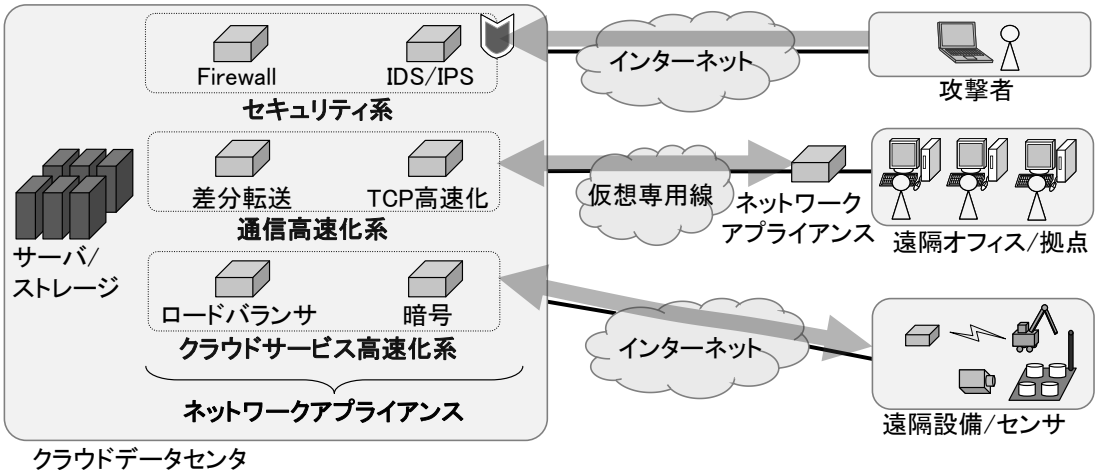


図 2　クラウドを利用する際に用いるネットワークアプライアンス

からの XML データの解析速度を向上させて、クラウドサービス利用時の応答速度を向上させる。また、キャッシュ機能を持つアプライアンスを用いて、遠隔地の端末からのデータ要求に素早く応える。更に、SSL (Secure Sockets Layer) –VPN (Virtual Private Network)、IPSec (Internet Protocol Security)などの暗号通信を高速化するネットワークアプライアンスを用いて、クラウドとセンサデバイスの間でやり取りする個人・業務データの機密を守る暗号通信を高速化する。

このように、クラウドの ICT 機器のリソースを、遠隔地から通信回線を経由して利用する際には、安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させるために、図 2 に示すように、セキュリティ系、通信高速化系、クラウドサービス高速化系のネットワークアプライアンスを利用する。

1.2　研究の目的と意義

クラウドの ICT 機器のリソースを遠隔地から通信回線を経由して利用する際には、クラウドへのネットワーク経由での安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させるために、セキュリティ系と高速化系のネットワークアプライアンスを利用する。高速化系のネットワークアプライアンスには、通信データの転送速度を向上させる通信高速化系と、アプリケーションの応答速度を向上させるクラウドサービス高速化系がある。

クラウド利用の通信が、インターネットのような公衆網を経由する場合は、クラウドが管理する ICT 機器のリソースやデータや、ネットワークの通信装置を守るため、セキュリティ系のネットワークアプライアンスを用いる。クラウド利用の通信が、通信遅延とパケット廃棄率が大きい回線を経由する場合は、ネットワークの回線帯域の利用効率を向上させてクラウドサービスと利用者との間のデータ転送速度を向上させるために、通信高速化系のネットワークアプライアンスを用いる。遠隔地の端末やセンサデバイスが、クラウド

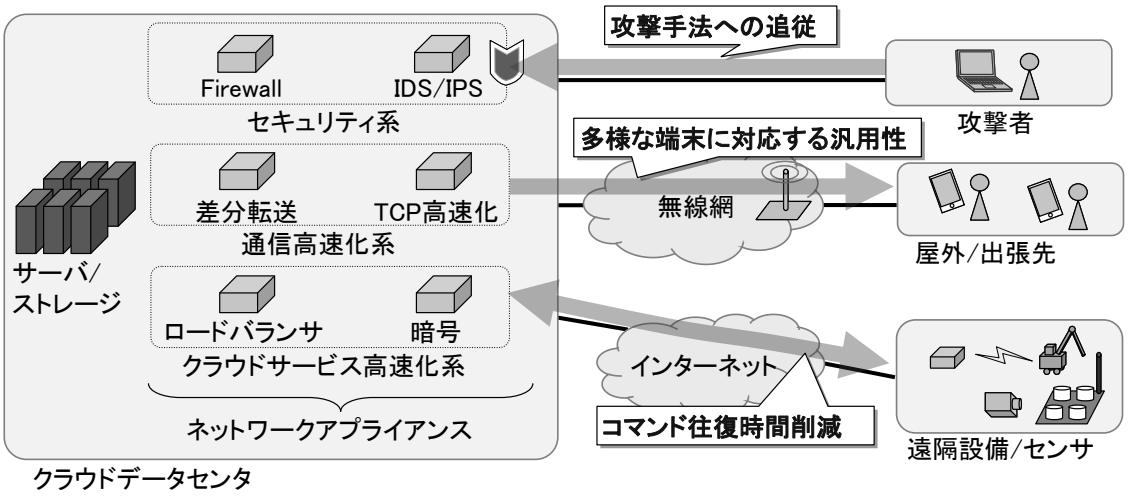


図 3　クラウドアクセス時に用いるネットワークアプライアンスの課題

利用の通信を行う場合は、クラウドが提供するサービスを肩代わりして、クラウドサービスの応答速度を向上させるために、クラウドサービス高速化系のネットワークアプライアンスを用いる。

このように、クラウドの ICT 機器のリソースを、遠隔地から通信回線を経由して利用する際には、安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させるために、以下のネットワークアプライアンス①～③を用いる。

① セキュリティ系のネットワークアプライアンス
② 通信高速化系のネットワークアプライアンス
③ クラウドサービス高速化系のネットワークアプライアンス

しかしながら、クラウドを利用する際に用いる上記のネットワークアプライアンス①～③には、図 3 に示すように、様々な課題があった。

セキュリティ系のネットワークアプライアンス①は、多数ユーザを収容するクラウドや、クラウドが接続するネットワークを防御するための高スループットを実現しつつ、日々進化する多様な攻撃手法に対応しなければならないという課題があった。通信高速化系のネットワークアプライアンス②は、多様なアプリケーションの通信に対応する高い汎用性を実現しつつ、通常の固定式の PC (Personal Computer)端末だけではなく、スマートフォンやタブレット PC などのモバイル端末や､設備監視装置や車載端末などのセンサデバイスなど、多様な端末に対する通信高速化をサポートしなければならないという課題があった。特に、モバイル端末やセンサデバイス向けの通信高速化をサポートするためには、ビットエラーによるパケット廃棄により通信速度が低下しやすいラストワンマイルの無線回線区間を高

速化せねばならないという課題があった。クラウドサービス高速化系のネットワークアプライアンス③は、遠隔地からクラウドにアクセスする際の往復通信時間が長く、遠隔地の端末からコマンド要求を送信してクラウドからリプライ応答が帰ってくるまでのコマンド往復時間が長い場合に、サービスの応答時間を高速化しなければならないという課題があった。

このように、ネットワークアプライアンス①～③には、以下①～③の課題があった。

①高スループットを実現しつつ、日々進化する多様な攻撃手法に対応
②多様なアプリケーションと端末に対応する汎用性の高い通信高速化
③クラウドサービスを遠隔地から利用時のコマンド往復時間削減および応答高速化

高スループットを実現しつつ、日々進化する多様な攻撃手法に対応せねばならない課題①を解決するためには、高速演算性能と柔軟な機能変更を可能とする汎用性を兼ね備えた装置を用いて、ユーザ要求やネットワーク状況に応じて対異常通信防御の機能を無瞬断で提供する必要性がある。また、多様なアプリケーションと端末に対応する汎用性の高い通信高速化をせねばならない課題②を解決するためには、様々なアプリケーションや端末が一般的に使用するTCP通信のパケットフォーマットに変更を加えずにTCP通信のまま高速化する必要性がある。加えて、モバイル端末やセンサデバイス向けの通信高速化をサポートするためには、無線区間への装置設置を要件としない通信高速化が必要となる。更に、クラウドサービスを遠隔地から利用する際のコマンド往復時間の削減、および応答時間を高速化せねばならない課題③を解決するためには、多様な場所への設置を可能とする小型省電力で高性能な装置を、モバイル端末やセンサデバイスとの距離が近いネットワークエッジに設置して、ユーザ要求やネットワーク状況に応じてクラウドが提供する様々なサービスを肩代わりして代理で実行する必要性がある。

本研究では、これらの必要性に応えるため、図 4 に示す以下①～③の特徴を持つネットワークアプライアンスを提案および開発し、クラウドデータセンタやネットワークエッジに設置することを提案した。

①高スループットを実現しつつ、様々なセキュリティ機能を無瞬断で更新
②多様なアプリケーションと端末に対応するTCP通信のままでの汎用的な通信高速化
③ネットワークエッジにおいて様々なクラウドサービスを代行して応答を高速化

本研究で開発した上記①～③の特徴を持つネットワークアプライアンスを適宜選択して適用することで、クラウドサービスへの安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させることが可能となる。具体的には、特徴①を持つセキュリティ系のネットワークアプライアンスを適用することで、高スループットを実現しつ

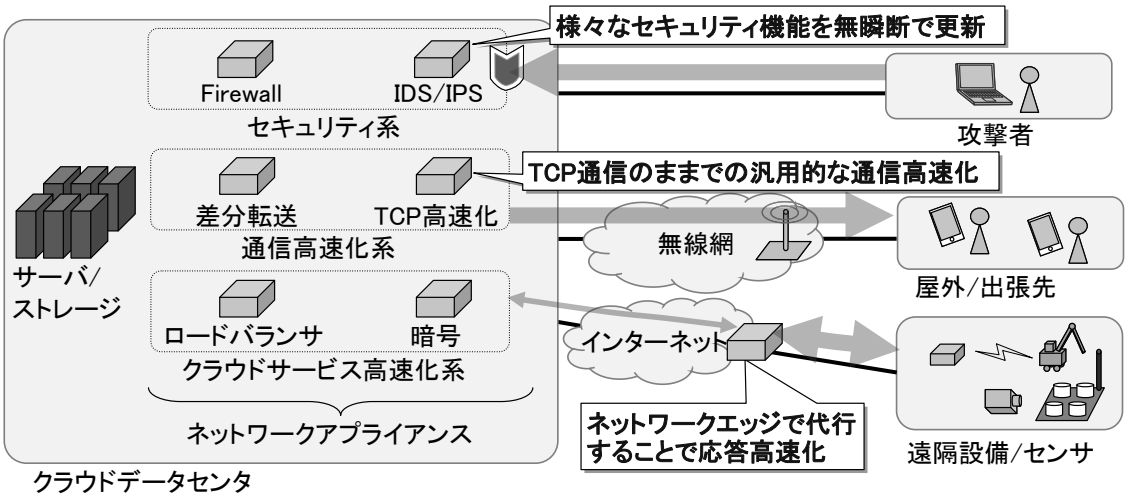


図 4　提案するネットワークアプライアンスの特徴

つ、日々進化する多様な攻撃手法に対応することが可能となる。また、特徴②を持つ通信高速化系のネットワークアプライアンスを適用することで、多様なアプリケーションと端末に対応する汎用性の高い通信高速化が可能となる。更に、特徴③を持つクラウドサービス高速化系のネットワークアプライアンスを適用することで、クラウドサービスを遠隔地から利用する際のコマンド往復時間短縮および応答高速化が可能となる。

なお、本研究で提案するネットワークアプライアンスは、クラウドデータセンタやネットワークエッジに、ゲートウェイと同じ形態で導入して利用する(図 4)。ブリッジのように動作する L2 透過型の装置もあれば、L3 以上のレイヤで IP アドレスやポート番号の変換を行う装置もある。

1.3　本論文の構成

本研究では、2 章にてネットワークアプライアンスの概要を整理する。クラウドの ICT 機器のリソースを、通信回線を経由して利用する際に用いるセキュリティ系、通信高速化系、クラウドサービス高速化系のネットワークアプライアンス(1.2 節記載の①～③)のそれぞれについて概要を説明する。更に、それぞれのネットワークアプライアンスにおいて最も重要と考える課題と、その解決のために開発したネットワークアプライアンスの機能について述べる。

セキュリティ系のネットワークアプライアンスにとって、高スループットを実現しつつ、日々進化する多様な攻撃手法に追従していく課題①の解決は重要である。3 章では、セキュリティ系のネットワークアプライアンスのなかで、高スループットを実現しつつ、日々進化する多様な攻撃手法に追従していく課題①の解決を最も必要とし、提案する特徴①(1.2 節記載)が最も必要とされている IDS や IPS などの対異常通信防御を対象とする。高スループットを実現しつつ、DDoS，P2P，Warm などの異常通信を探知して防御する対異常通信防御

アルゴリズムを無瞬断で動的再構成する方式を明らかにし、提案する特徴①(1.2 節記載)の実現方式について述べる。

通信高速化系のネットワークアプライアンスにとって、多様なアプリケーションや端末に対応する汎用的な通信高速化の課題②の解決は重要である。4 章では、通信高速化系のネットワークアプライアンスのなかで、多様なアプリケーションや端末に対応する汎用的な通信高速化の課題②の解決を最も必要とし、提案する特徴②(1.2 節記載)が最も必要とされている TCP 高速化を対象とする。TCP 通信のパケットフォーマットに準拠しつつ、廃棄率の変化率に基づく独自の輻輳制御を行うことで回線帯域の利用効率を向上させる TCP 高速化方式や、モバイル端末やセンサデバイス向けの TCP 通信高速化をサポートするため、無線回線区間への装置設置を要件としない TCP 通信高速化方式を明らかにし、提案する特徴②(1.2 節記載)の実現方式について述べる。

クラウドサービス高速化系のネットワークアプライアンスにとって、クラウドサービスを遠隔地から利用する際のコマンド往復時間短縮および応答高速化の課題③の解決は重要である。5 章では、クラウドサービス高速化系のネットワークアプライアンスのなかで、クラウドサービスを遠隔地から利用する際のコマンド往復時間短縮および応答高速化の課題③の解決を最も必要とし、提案する特徴③(1.2 節記載)が最も必要とされているクエリトランザクション高速化を対象とする。ネットワークエッジへの設置を可能とする小型省電力で高性能なネットワークアプライアンスを用いて、ユーザ端末の発行する様々な SQL クエリのコマンド要求に対して代理でリプライ応答を行い、クラウドサービスの一部を代行して応答を高速化する方式を明らかにし、提案する特徴③(1.2 節記載)の実現方式について述べる。

最後に 6 章で、本研究で得られた成果をまとめる。

第２章　ネットワークアプライアンスの関連研究

2.1　緒言

クラウドの ICT 機器のリソースを遠隔地から通信回線を経由して利用する際には、クラウドへのネットワーク経由での安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させるために、セキュリティ系と高速化系のネットワークアプライアンスを利用する。高速化系のネットワークアプライアンスには、通信データの転送速度を向上させる通信高速化系と、アプリケーションの応答速度を向上させるクラウドサービス高速化系がある。

本章では、クラウドのリソースを遠隔地から利用する際に用いる上記のセキュリティ系，通信高速化系，クラウドサービス高速化系のネットワークアプライアンスについて概要を説明し、関連研究について述べる。更に、本論文が対象とする機能について述べる。

2.2　ネットワークアプライアンスの概要

クラウドのリソースを遠隔地から利用する際に用いるネットワークアプライアンスは、1.2 節や 2.1 節において述べたように、セキュリティ系、通信高速化系、クラウドサービス高速化系の機能を提供する。ネットワークアプライアンスの系統分類と、系統毎の課題と、本研究における提案方式①～③(1.2 節)との対応関係をまとめると、図 5 に示す通りとなる。

セキュリティ系の機能としては、クラウドが管理するデータやネットワークのノードを守るために用いる IDS (Intrusion Detection System) / IPS (Intrusion Prevention System)などの対異常通信防御，DPI (Deep Packet Inspection)，Firewall (FW)，NAT / NAPT などが主要な機能である。

通信高速化系の機能は、ネットワークの回線帯域の利用効率を向上させて、クラウドの応答速度を向上させるために用いる。TCP 高速化，圧縮，差分転送などが主要な機能である。

クラウドサービス高速化系の機能は、遠隔地の多数のモバイル端末やセンサデバイスとの間で、大容量データの高速配信・解析やリアルタイムな情報取得を行うために用いる。クエリトランザクション、HTTP / XML テキスト解釈・変換、キャッシュ、SSL / IPSec 暗号、VOD、ロードバランサなどが主要な機能である。

以下 2.3～2.5 節において、各系統の個別機能と、本研究が対象とする機能について詳細に説明する。

2.3　セキュリティ系の機能

セキュリティ系の機能は、パケットの中身データと正常なパターンデータの差異から異常を検出する手法と、通信の統計情報と正常なパターンデータの差異も併せて異常を検出

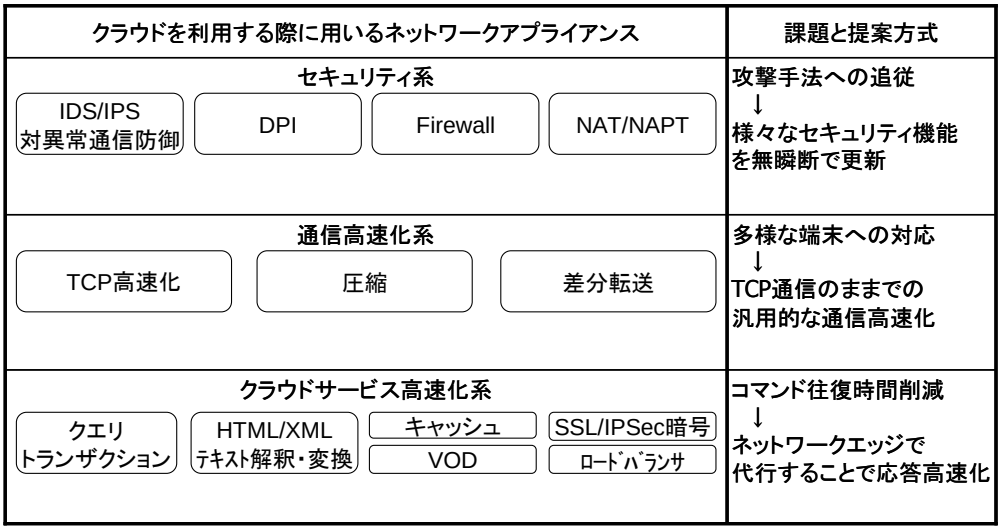


図 5　ネットワークアプライアンスが提供する機能の分類

する手法がある。パケットの中身データと正常なパターンデータの差異から異常を検出する手法として Firewall や NAT / NAPT があり、通信の統計情報と正常なパターンデータの差異も併せて異常を検出する手法として対異常通信防御や DPI がある。以下に詳細を説明する。

2.3.1　IDS や IPS などの対異常通信防御

対異常通信防御は、IDS (Intrusion Detection System) [6][7][8][9][10]と IPS (Intrusion Prevention System) [11]から成る。

IDS 機能を搭載したネットワークアプライアンスは､主に 2 つのタイプに分類できる[7]。1 つ目のタイプは、文献[6]に報告があるように、ロジック型攻撃(Land Attack 等)等の探知に適したパターンマッチングに基づくシグネチャ型である。もう一つのタイプは、SYN flood[12]をはじめとする DDoS 攻撃に代表されるフラッド型攻撃等の探知に適した通信の統計情報に基づくアノマリ型[8][9][10]である。共に、多数のパケットを高速に解析することが必要である。シグネチャ型は、パケットのデータの中身を見て、通信の内容がアプリケーションの正規の動作に則っているかどうか等をチェックする。一方、アノマリ型は、パケットのヘッダ情報を見て、通信パターンからトラフィックの異常を検出する。パケットのデータの中身やヘッダ情報を見るだけでなく、トラフィック量の日常の変化を学習して通常と異なるパターンを見つけた際に、ネットワークへの攻撃を検出する方法[13][14]の報告もある。

IPS 機能を搭載したネットワークアプライアンスは、IDS 同様のトラフィックの異常検出や、異常通信の検出や、攻撃トラフィックの検出を行うだけでなく、検出したトラフィックがサーバやストレージなどのリソース内部に進入しないように防御を行う。防御を有効

にしたトラフィックは通信できなくなるので、正常な通信まで通信できなくなる可能性を排除するためには高い検出精度が必要となる。

2.3.2 DPI

DPI (Deep Packet Inspection)機能を搭載したネットワークアプライアンスは、ネットワークを流れるパケットのデータの中身を解析することで、ネットワークを流れるトラフィックの詳細な統計情報を取得する機能を持つ。ネットワークに計測用のパケットを意図的に流すことで情報を取得するアクティブ型[15][16]と、ネットワークに影響を与えずにキャプチャしたパケットやルータやスイッチの統計情報などから情報を取得するパッシブ型[17]の 2 種類が存在する。ルータやスイッチなどは、sFlow[18]や NetFlow[19]など、コネクション毎の統計情報を取得する機能をサポートしている。DPI の機能を持つネットワークアプライアンスは、sFlow や NetFlow がカバーしない、より詳細なトラフィックの統計情報を取得するために利用する。

DPI で取得したネットワークの解析結果は、ネットワークの品質測定[20][21][22]や、ネットワーク障害の検出[16]や、異常通信やシステム侵入の検出[13][14][23][24][25]や、ネットワークを流れるコンテンツやアプリケーションの分析結果に基づく加入者の要望に応じた通信サービス提供[26][27][28][29]、等を目的として用いる。また、DPI 機能を持つアプライアンスは、用途によって取得すべき情報が異なる上に、ネットワークを流れるパケットを高速に解析するための高い演算性能を必要とするため、用途毎に特化したハードウェアを用いることで高スループットを実現する。GPU を用いた DPI[24]や、FPGA を用いた DPI[25][28]や、ネットワークプロセッサを用いた DPI[30]など、数多くのハードウェアを用いた DPI 高速化研究の報告がある。更に、ブルームフィルタと呼ばれるハードウェアを用いた文字列比較の高速化手法[31]の報告がある。ブルームフィルタは、ストリームデータの複数のハッシュ関数を同時に計算してシグネーチャセットと比較することで文字列比較を高速化する。

2.3.3 Firewall

Firewall は、文献[32]に報告があるように、2 つのネットワークの境界に設置して、予め許可したポート番号・プロトコル以外の通信を遮断するゲートウェイとして、インターネット黎明期から存在する。本概念に加えて、あらかじめ許可したポート番号・プロトコルの通信についても、TCP コネクション確立時の 3-Way-Handshake が正常に行われているか否か、シーケンス番号が正しいか否か、通信データの中身がアプリケーションの正規の動作に則っているか否か、等をチェックして、正規の手続きに則り通信が行われているか否かを判定する。通信が異常と判定した場合は、あらかじめ許可したポート番号・プロトコルの通信であっても通信を遮断する。

防御すべきネットワークが、他のネットワークとの接続点を複数持つ場合は、複数のフ

ァイアウォールを設置せねばならず、設定が複雑化しやすい。設定複雑化の問題を解決するために、マルチファイアーウォール環境における設定を容易化する手法[33]の報告がある。

また、ファイアウォールが予め許可したポート番号・プロトコル以外の通信を遮断する処理に必要な演算リソースは少ないが、予め許可したポート番号やプロトコルの通信が正規の手続きに則り通信が行われているかどうかを判定する処理は、多くの演算リソースを消費するので、高速化が難しい。高速化するために、FPGA 等のハードウェアを用いて高速化する手法[34]や、統計的な情報に基づいてフィルタ検索ツリーを最適化して高性能化する工夫[35]などの報告がある。加えて、通信データの中身がアプリケーションの正規の動作に則っているかどうかをチェックするアルゴリズムは、アプリケーションのアップデートに応じて、定期的に更新する必要がある。本ニーズを実現するために、定期的なアルゴリズム更新と、高速性能を両立するハードウェア[36]の報告がある。

2.3.4 NAT / NAPT

NAT / NAPT 機能を搭載したネットワークアプライアンスは、クラウドデータセンタ内部のプライベート IP アドレスと、クラウドデータセンタ外部に公開するパブリック IP アドレスの間で変換を行う。クラウドのリソースにインターネット経由でアクセスする際には、文献[37]に報告があるように、データセンタ外部においてパブリック IP アドレスを用いた通信を行い、データセンタ内部においてプライベート IP アドレスを用いた通信を行うことで、データセンタ内部の ICT 機器の IP アドレスを隠蔽し、セキュリティを確保する。クラウドデータセンタ内部において IPv6 アドレス、クラウドデータセンタ外部においてパブリックな IPv4 アドレスを用いることで､アドレスを拡張する手法[38]についての報告もある。

IP アドレスの変換に加えて、ポート番号の変換も行うことで、少数のパブリック IP アドレスを、プライベート IP アドレスを持つ多数の端末で共有することを可能とする。外部からのアクセス要求を、ポート番号毎に異なる宛先 IP アドレスに変換することで、サービス毎にアクセス要求を分散させる目的で用いることも可能である。

2.3.5 本研究の対象

セキュリティ系のネットワークアプライアンスの機能には、IDS / IPS などの対異常通信防御、DPI、Firewall や、NAT / NAPT などがある(図 6)。

IDS / IPS などの対異常通信防御は、日々進化する多様な攻撃手法を検出・判別する必要がある。DPI は、用途毎に取得すべき情報が変化する。Firewall は、正規通信の手続きの変更頻度が低く、機能変更の頻度が低い。NAT / NATP は、標準的な枯れた機能であり、機能変更は不要である。

セキュリティ系のネットワークアプライアンスには、1.2 節に述べたように、高スループットを実現しつつ、日々進化する多様な攻撃手法に対応せねばらならない課題①がある。本研究では、課題①の解決が最も重要と考え、上記機能のうち日々進化する多様な攻撃手

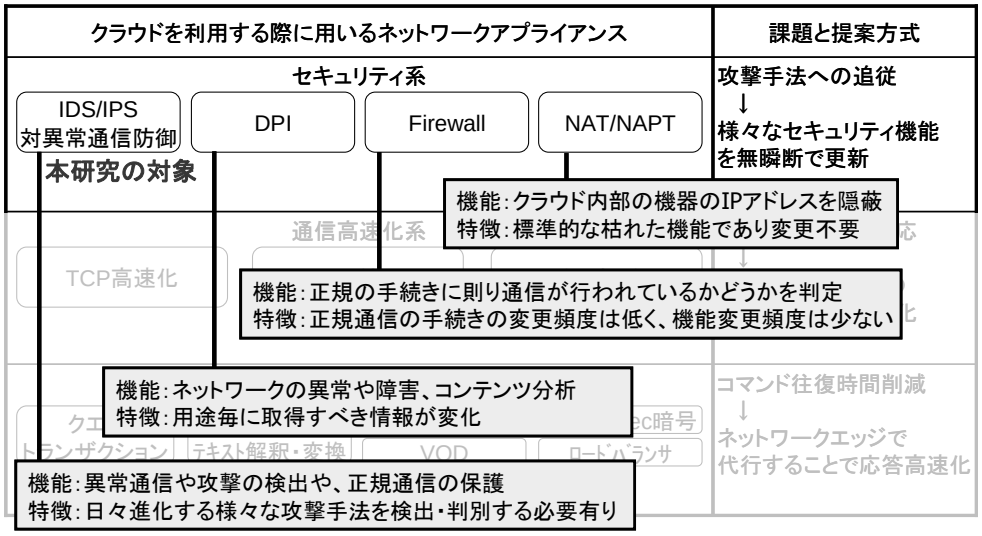


図 6　本研究が対象とするセキュリティ系のネットワークアプライアンス

法を検出・判別する必要がある IDS / IPS などの対異常通信防御を対象とする。具体的には、高速演算性能と柔軟性を備えた動的再構成 LSI を用いて、高スループットを維持したまま、DDoS，P2P，Warm などの異常通信を探知して防御する対異常通信防御の機能を無瞬断で更新する方式の提案および評価を行う。

2.4　通信高速化系の機能

通信高速化系の機能は、帯域利用効率を向上させることで転送時間を削減する手法と、ネットワークに流れる通信量を削減することで転送時間を削減する手法から成る。帯域利用効率を向上させる機能として TCP 高速化があり、ネットワークに流れる通信量を削減する機能として差分転送と圧縮がある。以下に、詳細を説明する。

2.4.1　TCP 高速化

TCP 高速化機能を搭載したネットワークアプライアンスは、文献[39]に報告があるように、WAN 経由の遠距離通信や無線通信など、通信遅延やパケット廃棄率の大きな回線を使用する通信のスループットを向上させる。本ネットワークアプライアンスは、クラウドデータセンタやネットワークエッジに設置され、送信側端末が送るパケットに対して代理 ACK を返信して送信側との間の TCP コネクションを一旦終端して、パケットをバッファリングする。受信側との間の TCP コネクションでは、独自の輻輳アルゴリズムを用いてパケット送信を行うことで高速化を行う。

IETF 標準の TCP Reno [40], New Reno [41]は、パケット廃棄の発生に基づいてネットワークで輻輳が発生しているか否かを判断する。他のトラフィックと一時的に競合して 1 パケットでも廃棄が発生すると、ネットワークに輻輳が発生したと判断してスループットを半

分に減少させるので、長距離通信回線のように通信遅延が大きく廃棄の起きやすいネットワークでは、スループットが低下しやすい。Linux 標準搭載の CUBIC-TCP [42]では、輻輳発生時のスループットの低下率を 5 割未満にしてスループットの減少を抑制している。また、Windows 搭載の Compound TCP [43]は、RTT が大きい環境において、輻輳未発生時の帯域増加率を増加させ、スループットを向上させやすくしている。その一方で、CUBIC-TCP も Compound TCP も、TCP Reno 同様にパケット廃棄の発生に基づいてネットワークにおける輻輳の発生を判断するので、他のトラフィックとの一時的な競合により発生するパケット廃棄の影響を受けやすく、スループットが低下しやすい。

TCP 高速化機能を搭載したネットワークアプライアンスは、パケット廃棄の発生をベースとしたアルゴリズム(High-Speed TCP[44])や、遅延増減をベースとしたアルゴリズム(Fast-TCP [45])など、改良型の輻輳制御アルゴリズムの利用によりスループットの低下を抑制して、TCP 通信の性能を向上させる。

2.4.2 圧縮

圧縮機能を搭載したネットワークアプライアンスは、クラウドデータセンタとネットワークエッジの両方に設置され、送信側の装置で通信データの圧縮を行い、受信側の装置で圧縮したデータの解凍を行うことで、通信データ量を削減する。無線デバイスの消費電力を減らすために圧縮を用いる研究[46]の報告もある。

通信データの圧縮には、データ圧縮ツールの LHA，GZIP，ZIP などが利用する可逆データ圧縮アルゴリズムを用いる。Lempel-Ziv アルゴリズム(LZ77)とハフマン符号化を組み合わせた方式[47]や、LZ77 の改良アルゴリズムである LZSS にハフマン符号化を組み合わせて改良した Deflate (デフレート)などのアルゴリズムを用いる。

テキストデータ等の同一データの出現頻度が高いファイルを転送する際の圧縮効果が高いが、ビデオデータ等の圧縮済みデータや、暗号通信等のランダムビット列データでは、圧縮効果が得られにくい。

2.4.3 差分転送

差分転送の機能は、ストレージやネットワーク向けの重複排除(Deduplication)[48]と呼ばれる分野で数多くの研究報告がある。

ストレージにおける重複排除[49]では、ファイルデータを一定サイズのデータブロック単位で分割して、データブロックの位置を示すポインタの配列と共に保存する。2 つ以上のファイルデータの間で重複するデータブロックが存在する場合は、重複するデータブロックを一つにまとめて、複数の異なるファイルデータからポインタを用いて参照することで、ディスクスペースの節約を可能にしている。

ネットワークにおける重複排除[50]では、差分転送機能を搭載したネットワークアプライアンスを、クラウドデータセンタとネットワークエッジの両方に設置する。送信側と受信

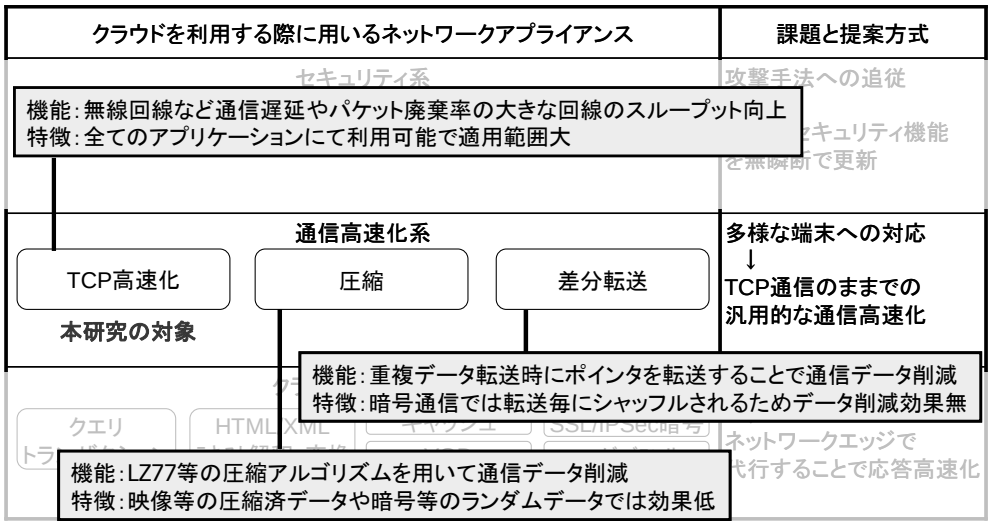


図 7　本研究が対象とする通信高速化系のネットワークアプライアンス

側の両方のネットワークアプライアンスに、転送済みデータを保持しておき、重複するデータを再び転送する際に、送信側でリファレンスやポインタに変換して転送し、受信側で復元することで重複を排除して通信データ量を削減する。学術分野において数多くの研究発表[51][52][53]の報告があるだけでなく、産業界においても数多くの WAN 最適化装置がレガシ機能の一つとして差分転送機能を搭載しており[54]、公開特許[55][56]も存在する。

差分転送機能を搭載したネットワークアプライアンスは、主に、データブロック単位で重複排除を行うタイプ[55][56]と、ファイル単位で重複排除を行うタイプ[57]に分類できる。データブロック単位で重複排除を行うタイプは、数百から数十 K バイト単位で一致判定を行うので、通常のキャッシュと異なり、一部書き換わったファイルを転送する場合でも、通信データ量を削減することが可能である。また、ファイル単位で重複排除を行うタイプのように、アプリケーション毎にアクセス要求を解析して、キャッシュとの一致判定をする必要がないため、全てのアプリケーションに対して汎用的に利用することが可能である。その一方で、小さなデータサイズ単位で重複判定するため、必要な演算量が増加しやすく、高速化が難しい課題があった。本課題の解決のため、重複判定に必要な演算量を削減することで高速な差分転送を実現する手法[58]の報告がある。

なお、暗号通信等のランダムビット列データは、同一のファイルデータを再送する場合でも、異なるビット列データとなるため、データ削減効果が得られない。

2.4.4　本研究の対象

通信高速化系のネットワークアプライアンスの機能には、TCP 高速化，圧縮，差分転送などがある(図 7)。

TCP 高速化は、無線ネットワークを経由するモバイル端末向けなど、ほとんど全てのアプリケーションや端末が利用可能で適用範囲が広い。圧縮は、映像等の圧縮済みデータや

暗号通信等のランダムデータではデータ削減効果が低い。差分転送は、同一データでも転送する毎にランダムデータ列となる暗号通信において、データ削減効果が得られない。

通信高速化系のネットワークアプライアンスには、1.2 節に述べたように、多様なアプリケーションと端末に対応する汎用性の高い通信高速化に対応せねばらならない課題②がある。本研究では、課題②の解決が最も重要と考え、上記機能のうち、無線ネットワークを経由するモバイル端末向けなど、ほとんど全てのアプリケーションや端末が利用可能で適用範囲が広い TCP 高速化を対象とする。具体的には、TCP 通信のパケットフォーマットに準拠しつつ、パケット廃棄率の変化率に基づく独自の輻輳制御を行うことで回線帯域の利用効率を向上させる TCP 高速化方式の提案および評価を行う。

2.5　クラウドサービス高速化系の機能

クラウドサービス高速化系の機能は、コマンド要求に対するリプライ応答を一時的にキャッシュしたデータを用いて実行することで、コマンド要求からリプライ応答までの往復時間を短縮してクラウドサービスの応答を高速化するキャッシュ系と、クラウドサービスの中で負荷の大きな演算処理を代行するハードウェアなどを用いて短時間で実行することでクラウドサービスの応答を高速化するオフロード系と、大量のコマンド要求を受け付けて複数のサーバに分散させる負荷分散系がある。キャッシュ系として、キャッシュ、VOD、クエリトランザクションがあり、オフロード系として、SSL / IPSec 暗号化、HTML / XML テキスト解析・変換があり、負荷分散系として、ロードバランサがある。以下に詳細を説明する。

2.5.1　クエリトランザクション

クエリトランザクション高速化機能を搭載したネットワークアプライアンスは、クラウドサービスの DB クエリトランザクションの一部を代行して実行することで、高速応答な DB クエリトランザクションを実現する。FPGA 等を用いて DB 分析やクエリトランザクションをハードウェア化して高速化する手法[59][60]の報告がある。

クエリトランザクション高速化機能を搭載したネットワークアプライアンスをネットワークエッジに分散配置する場合、ネットワークアプライアンスが、クラウドデータセンタの DB の一部をキャッシュしておき、クライアント端末の発行する SQL 等のコマンド要求を解析して、代理でリプライ応答を返す。全てのネットワークアプライアンスに DB データをレプリケーションしてしまうと、クラウドデータセンタのオリジナルと、複数の全てのネットワークアプライアンスのレプリカとの間で DB データの最新性をチェックする必要があり、応答時間が遅くなる。DB データの最新性をチェックする処理を減らして応答時間を高速化するためには、DB データのレプリケーション先を 1 つのネットワークアプライアンスに限定する必要がある。個人が各自の機密データにアクセスするなど、同一の DB データに複数のユーザから同時アクセスしないケースでは、各ユーザが使うデータをユーザの

一番近くのネットワークアプライアンスにコピーしておけば、データの同一性を保ちつつ高速応答なクエリトランザクション型サービスを実現することが可能となる。

2.5.2 HTML / XML テキスト解析・変換

HTML / XML テキスト解析・変換機能を搭載したネットワークアプライアンスは、HTML 形式や XML 形式のテキストデータ, センサや設備監視装置などが取得するバイナリデータ, RDB (Relational Data Base)が蓄積するデータ，RDB を操作するための SQL、等の様々な形式のデータを解析したり、データの形式を変換したりすることで、クラウドサービスの応答を高速化する。

XML はアプリケーションが自動的にアクセスしやすいデザインとなっており、WEB において構造情報を表現・交換する方式として産業界でスタンダードになりつつある。一方、オンライン文書の多くは、人に表示するためのデザインである HTML で記述する。HTML で記述したオンライン文書を XML アプリケーションが理解できるよう、HTML 形式のテキストデータを XML 形式のテキストデータに変換するネットワークアプライアンス[61]の報告がある。

また、一般的な RDB は、SQL を介してデータを入出力する仕様である。RDB のデータを XML アプリケーションが直接理解することを可能とするために、SQL を介さず、RDB のデータを XML へ直接変換する手法[62]の報告がある。逆に、RDB のデータを XML アプリケーションが直接操作することを可能とするために、XML を SQL へ変換する手法[63]の報告もある。

XML アプリケーションが XML 形式のテキストデータに対して実行するフォーマットチェック・解読・分類・比較などの処理は、多くの演算を必要とするため、応答速度が低下しやすい。応答速度を改善するために、ハードウェアによる XML パーサを用いてフォーマットチェックやデータ分類を高速化する手法[64]や、XML 形式のテキストデータ同士の包含関係の比較を高速化する手法[65]の報告がある。

2.5.3 キャッシュ

キャッシュ機能を搭載したネットワークアプライアンスは、ユーザ側に近いネットワークエッジに設置され、アクセス頻度の高いファイルやデータをキャッシュしておき、クラウドへのファイルやデータへのアクセスが発生した時に、クラウドに代わりファイルやデータの返信を行うことで、高速応答なアクセスを実現する。

ネットワークで一般的に使われる Web アプリのトラフィックには冪乗則が知られており、アクセス要求の 92%は過去にアクセスしたファイルへの要求という報告[66]がある。キャッシュを、ファイルアクセスの人気度が冪乗則に従う Web アプリなどのアプリケーションで用いれば、高い確率でキャッシュヒットし、高速応答なアクセスを実現することが可能となる。キャッシュのヒット確率を高めるに、複数のキャッシュ同士でキャッシュデータを

共有する手法[67]の報告がある。また、キャッシュを、大容量なビデオデータの転送とシークタイムの高速化が必要なストリーム配信系の VOD で利用することで、効果的にネットワークを流れるデータ量を削減して、利用者の体感品質を向上させる手法[68][69][70]の報告がある。

アクセス要求のあったファイルやデータがキャッシュにヒットした場合、ヒットしたキャッシュの最新性を確認する必要がある。クラウドデータセンタのオリジナルファイルやデータの更新時刻を確認するため、クラウドデータセンタとネットワークアプライアンスの間で、最低でも一往復分の通信遅延が発生する。

2.5.4 SSL / IPSec 暗号化

暗号化機能を搭載したネットワークアプライアンスは、クラウドデータセンタやネットワークエッジに設置され、サーバやクライアントの代わりに暗号通信を行いオフロードすることで、インターネットなどの公衆網を用いて遠隔地からクラウドにアクセスする際の暗号通信の応答速度を高速化する。レイヤ 4 の暗号通信プロトコルである SSL の高速化[71]や、レイヤ 3 の暗号通信プロトコルである IPSec の高速化[72]の報告がある。

暗号化機能を搭載したネットワークアプライアンスは、暗号通信開始時に RSA 等の公開鍵暗号アルゴリズムを用いて鍵交換を行う。その後、交換した鍵をベースにして AES や RC4 等の共通鍵暗号アルゴリズムを用いて、送信側で平文データの暗号化を行い、受信側で暗号データの復号を行う。更に、MD5 や SHA1 などのハッシュアルゴリズムを用いて、データブロック毎にチェックサムを計算して、通信経路上でデータがすり替わることなく正しく到達していることを確認する。

暗号通信が用いる公開鍵暗号アルゴリズム、共通鍵アルゴリズム、ハッシュアルゴリズムは、多くの演算を必要とする。そのため、暗号化機能を搭載したネットワークアプライアンスが暗号通信をオフロードすることで、暗号通信の応答速度を高速化することが可能となる。RSA 等の公開鍵暗号アルゴリズムの高速化[73][74]、AES や RC4 等の共通鍵暗号アルゴリズムの高速化[75][76]、MD5 や SHA1 などのハッシュアルゴリズムの高速化[77][78]の報告がある。

2.5.5 VOD

VOD サービスは、テキストや画像などから成る Web サービスと比較して使用帯域が大きいため、ディスク I/O やサーバのボトルネックを解消して、スループットを向上させるネットワークアプライアンス[79]の報告がある。加えて、ビデオの再生位置を変更する時のシークタイムがユーザ満足度にインパクトを与えるとの調査結果[80]が知られており、ネットワークのボトルネックを解消して、通信品質を向上させることで、VOD の応答時間を高速化して、ユーザ満足度を高めるネットワークアプライアンス[81][82][83]の報告がある。

VOD サービスにおけるディスク I/O やサーバのボトルネックを解消するため、ロードバ

ランサのようなネットワークアプライアンスを用いて、ストレージやサーバをクラスタ化してスループットを向上させる。複数台のストレージやサーバの間でアクセス要求を分散することで、高性能なサーバやストレージを仮想的に 1 台で実現する。また、VOD サービスでは、再生端末が一定量のビデオデータを蓄積してからビデオ再生を開始するため、ビデオの再生位置を変更する時に高いスループットが必要となる。ビデオの再生位置を変更する時の先頭データだけを広帯域送信するサーバと、後続データを通常の帯域で送信するサーバの間でロードバランシングすることで、サーバのピークスループットを抑制し、サーバ数を削減する手法[84]の報告がある。

加えて、VOD サービスにおけるネットワークのボトルネックを解消するため、クライアントの近くのネットワークエッジにミラーサーバやキャッシュサーバの機能を持つネットワークアプライアンスを設置して、往復通信時間を短縮して応答性を向上させる手法[85]の報告がある。また、ネットワークアプライアンスがキャッシュすべきデータ容量を削減するため、プレフィックスキャッシュ[81][82][83][86][87][88][89][90][91][92][93]と呼ばれる手法の報告がある。高速転送が必要な初期データのみをキャッシュしておき、後続データはクラウドのデータセンタから送信することで、キャッシュすべきデータ容量を削減することが可能となる。更に、ネットワークアプライアンスのキャッシュヒット率を向上させるため、パーシャルキャッシュ[68][69][70]と呼ばれる手法の報告がある。パーシャルキャッシュは、複数に分割することでサイズを小さくしたビデオデータを、アクセス頻度に応じてキャッシュしておく手法である。同じビデオデータであっても再生位置に応じてアクセス頻度が異なる。アクセス頻度の高い再生位置のビデオデータのみをキャッシュしておくことで、キャッシュヒット率を高めつつ、ディスク容量を削減することが可能となる。

上記の VOD 高速化機能を搭載したネットワークアプライアンスを、クラウドデータセンタやネットワークエッジへ設置することで、VOD サービスを提供する際のスループットや応答性を向上させる。

2.5.6 ロードバランサ

ロードバランサ機能を搭載したネットワークアプライアンスは、サーバやストレージの性能が 1 台では不足するときに、文献[94][95][96]のようにクラスタ化した複数台のサーバにアクセス要求を分散させたり、文献[97][98][99][100][101][102][103]のようにクラスタ化した複数台のストレージにリード／ライト要求を分散させたりして、仮想的に高性能なサーバやストレージを 1 台で実現する。クラウド事業者が、ユーザ向けサービスの一つとして、ロードバランサ機能を提供する手法[104]の報告もある。

ロードバランサが受信したパケットをサーバへ振り分けるアルゴリズムは、主にスタティック方式[105]とダイナミック方式[106]の 2 つがある。

スタティック方式は、ポリシードリブン方式とも呼ばれ、予め定義したルールやポリシーに基づいてパケットを振り分ける。クライアントからのリクエストをサーバに均等に順

番に振り分けるラウンドロビン(均等負荷分散)方式の利用が一般的である。サーバ毎に定義した重みの割合に応じてパケットを振り分ける重み付けラウンドロビン(Weighted Round-Robin)方式もある。

ダイナミック方式は、現在のトラフィックに基づいて負荷分散することで、効率的なサーバリソースの活用を可能にする。現在のアクティブなコネクション接続数が最も小さいサーバに転送する最小コネクション(Least Connection)方式の利用が一般的である。他に、現在のアクティブなコネクション接続数とサーバ毎に定義した重みの割合に基づいてパケットを振り分ける重み付け最小コネクション(Weighted Least Connection)方式、レイヤ 7 のアプリケーション層のリクエストの未処理数が最も少ないサーバに転送する最速(Fastest)方式、ある一定時間の平均コネクション数が最も少ないサーバに転送する監視(Observed)方式、ある一定時間の傾向を解析して、平均コネクション数が減少している場合はより多く転送を行い、平均コネクション数が増加している場合は転送を少なくする予測(Predicteive)方式、CPU やメモリ使用率が低いサーバに優先的に振り分ける動的比率(Dynamic Ratio)方式などがある。動的比率方式を利用するためには、SNMP エージェント等のモニタ機能を用いて各サーバを監視する必要がある。サーバの負荷が頻繁に変化する場合は、可用性や負荷情報や応答時間から振り分け先を総合的に判断するダイナミックフィードバック方式を用いる。

クラスタ化したサーバ向けのロードバランサ以外に、ランダム方式[107]、VM(Virtual Machine)向けの集中方式[108]、コンテンツアウェアの非集中方式[109]、等のロードバランサの形態の報告がある。

ランダム方式は、システムの不確実性に対応するために用いられ、ランダムサンプリングに基づいてパケットを振り分けるサーバを選択する。サーバは仮想ノードとして表され、複数の仮想ノードと接続したネットワークにおいて、各仮想ノードの可用性を反映させたランダムサンプリングに基づいてパケットの振り分け先を選択する。

VM 向けの集中方式は、異なる OS を搭載した複数のサーバにおいて複数のインスタンスを同時に実行するシステムなど、一つのトランザクションが同時に必要とする演算リソースが分散する環境において、複数のサーバにわたり負荷を均一に分散するために用いる。

コンテンツアウェアの非集中方式は、P2P に似たソリューションである。分散コンピューティング環境を用いて、同一のコンテンツを複数のサーバにコピーして蓄積する。コンテンツへのアクセス要求が来たら、コンテンツの存在するサーバを探索して、負荷が最小となるサーバへとアクセス要求を振り分ける。

2.5.7 本研究の対象

クラウドサービス高速化系のアプライアンスの機能には、クエリトランザクション、HTML / XML テキスト解析・変換、キャッシュ、SSL / IPSec 暗号化、VOD、ロードバランサなどがある(図 8)。

クエリトランザクションは、SQL 等のコマンド発行とレスポンス返信の往復を大量にシ

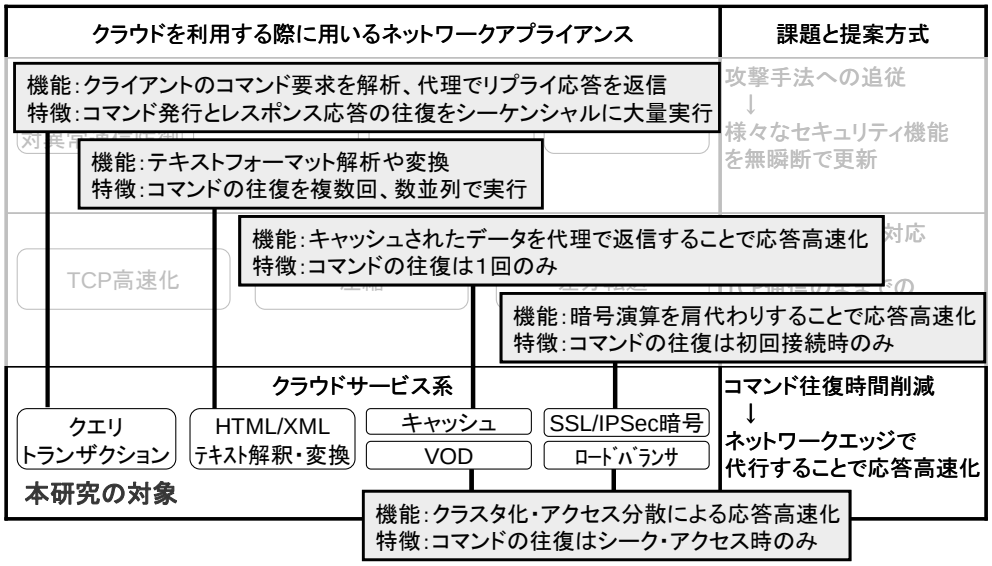


図 8　本研究が対象とするクラウドサービス高速化系のネットワークアプライアンス

ーケンシャルに実行する。HTML / XML テキスト解析・変換は、HTML や XML 形式のコマンド発行とレスポンス返信の往復を複数回、並列に実行する。キャッシュは、大きなファイルに対する読み出し要求コマンド発行とデータ返信が 1 回だけ発生する。SSL / IPSec 暗号化は、初回接続時の鍵交換の際に、コマンド発行とレスポンス返信の往復が数回だけ発生する。VOD やロードバランサは、ビデオの再生開始・シーク時や、TCP セッション確立時にコマンド発行とレスポンス返信の往復が発生する。

クラウドサービス高速化系のネットワークアプライアンスには、1.2 節に述べたように、クラウドサービスを遠隔地から利用する際のコマンド往復時間の削減および応答時間を高速化せねばならない課題③がある。本研究では、課題③の解決が最も重要と考え、上記機能のうち、コマンド発行とレスポンス応答の往復を大量にシーケンシャルに実行する必要があり、コマンド往復時間削減という課題の解決が極めて重要であるクエリトランザクションを対象とする。具体的には、ユーザ端末に近いネットワークエッジへの設置を可能とする小型省電力で高性能なネットワークアプライアンスを用いて、ユーザ端末の発行する様々な SQL クエリのコマンド要求に対して代理でリプライ応答を行い、クラウドサービスの一部を代行して応答を高速化する方式の提案および評価を行う。

2.6　本研究の対象領域

対異常通信防御機能を持つネットワークアプライアンスは、「高スループットを実現しつつ、様々なセキュリティ機能を無瞬断で更新」に対する要求が高いが、研究事例は少ない。そこで、第３章では、特徴①を備えた対異常通信防御機能を持つネットワークアプライアンスを実現するために、高速演算性能と柔軟性を備えた動的再構成 LSI を用いて、高スループットを維持したまま、DDoS、P2P、Warm などの異常通信を探知して防御する対異常通

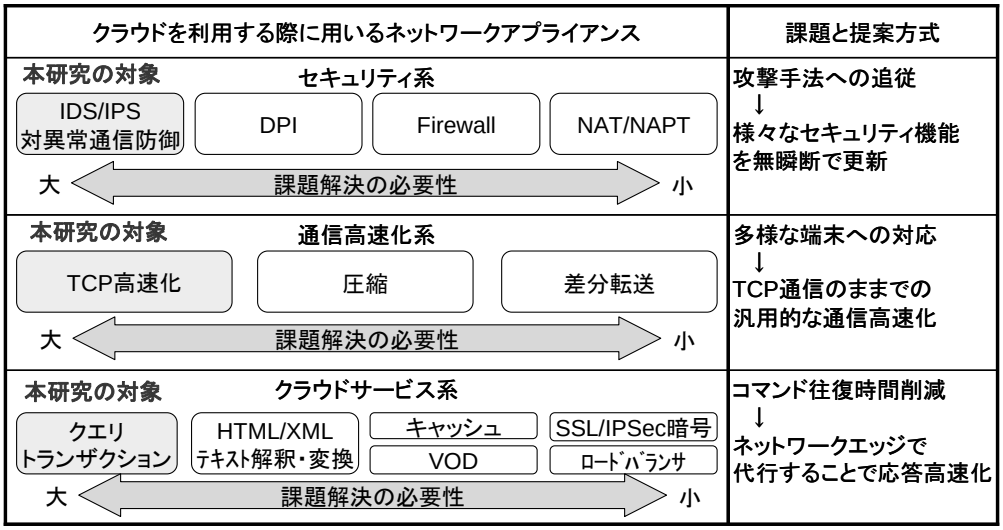


図 9　本研究が対象とするネットワークアプライアンスの機能

信防御の機能を無瞬断で更新する方式を提案する。提案方式により、高スループットを実現したまま、パケット廃棄を起こすことなく対異常通信防御アルゴリズムを更新することが可能となる。

また、TCP 通信高速化機能を持つネットワークアプライアンスは、「多様なアプリケーションと端末に対応する TCP 通信のままでの汎用的な通信高速化」に対する要求が高いが、研究事例は少ない。そこで、第 4 章では、特徴②を備えた TCP 高速化機能を持つネットワークアプライアンスを実現するために、TCP 通信のパケットフォーマットに準拠しつつ、廃棄率の変化率に基づく独自の輻輳制御を行うことで回線帯域の利用効率を向上させる TCP 高速化方式を提案する。更に、モバイル端末やセンサデバイス向けの TCP 通信高速化をサポートするため、無線回線区間への装置設置を要件としない TCP 通信高速化方式を提案する。提案方式により、通信遅延とパケット廃棄率の大きなネットワークを経由する TCP 通信や、無線ネットワークを経由するモバイル端末向けの TCP 通信を高速化することが可能となる。

更に、クエリトランザクション高速化機能を持つネットワークアプライアンスでは、「ネットワークエッジにおいて様々なクラウドサービスを代行して応答を高速化」が重要であるが、研究事例は少ない。そこで、第 5 章で、特徴③を備えたクエリトランザクション高速化機能を持つネットワークアプライアンスを実現するために、ユーザ端末に近いネットワークエッジへの設置を可能とする小型省電力で高性能なネットワークアプライアンスを用いて、ユーザ端末の発行する様々な SQL クエリのコマンド要求に対して代理でリプライ応答を行い、クラウドサービスの一部を代行して応答を高速化する方式を提案する。提案方式により、遠隔地からクラウドサービスを利用する際の応答時間を高速化することが可能となる。

図 9 に、クラウドのリソースを遠隔地から利用する際に用いるネットワークアプライア

ンスのセキュリティ系、通信高速化系、クラウドサービス高速化系の各機能と、系統毎の課題と、本研究における提案方式①～③(1.2 節)との対応関係と、各系統内における機能毎の課題解決の必要性を示す。本研究は、必要度の最も高い図 9 の左側に記載された機能を対象とする。

第３章　攻撃手法の変化に対応可能な対異常通信防御

3.1　緒言

近年、インターネット等の IP (Internet Protocol)ネットワークの社会インフラ化に伴い、ネットワークセキュリティの重要度が高まってきている。そのため、多くの企業や個人、あるいは、クラウドデータセンタ事業者は、自分のネットワーク内に設置した端末からの情報漏えいや、端末への攻撃を防止するため、他のネットワークとの外部アクセス点(図 10 ①)に、IDS (Intrusion Detect System[6][7][8][9][10])や IPS (Intrusion Prevention System[11])などの対異常通信防御の機能を持つネットワークアプライアンスを設置して、内部の情報資源の防御を行うのが一般的になっている。

加えて、DDoS(Denial of Service)攻撃のような異常通信が、クライアントとサーバの間に設置したルータ(図 10 ②)に障害を引き起こすのを防ぐため、ネットワークエッジ(図 10 ③)における対異常通信防御が、近年ますます重要になりつつある(図 10)[36][110]。クラウドデータセンタの他のネットワークとの外部アクセス点(図 10 ①)だけでなく、ネットワークエッジに設置したエッジルータ(図 10 ③)にも、異常通信を探知・制御する高速回線対応の対異常通信防御機能を追加して、ネットワークを流れる全通信を解析し、通信障害の原因となる P2P (Peer to Peer) などの過大通信や、DoS (Denial of Service)、DDoS (Distributed Denial of Serive) などのルータやサーバの攻撃を目的とした不正通信など各種の異常通信を確実に探知、排除したいとの要求が高まりつつある。

対異常通信防御の機能を持つネットワークアプライアンスは、ネットワークを流れるパケットを高速に解析して、ネットワーク帯域を占有する P2P 通信や ルータへ障害を引き起こす DDoS 攻撃のような多様な異常通信を防御する。それと同時に、ユーザ拠点とクラウドデータセンタとの間の正常通信を高確率で保護する必要がある。正規通信の保護では、往路と復路のパケットのシーケンス番号の整合性を確認することにより、正常か異常かを判定する。大きな RTT (Round Trip Time)を持つ TCP 通信の場合、正常か異常か判別可能となるために必要な時間が増大する。

ネットワークにおいて対異常通信防御の機能を持つネットワークアプライアンスが解析すべきパケットデータの量は、ネットワークの通信量の増加に伴って、非常に膨大なものとなっている。また、ルータやサーバへの攻撃手法は絶えず進化し続けている[111]。そのため、対異常通信防御の機能を持つネットワークアプライアンスは、異常通信を短時間で検出できるよう、大量のパケットを高速に解析できる高い演算性能と、日々進化する新手の攻撃に対応できるよう、対異常防御アルゴリズムを瞬時に更新できる柔軟性とを兼ね備えたプロセッサを必要とする。

汎用プロセッサや ASIC (Aplication Specific Integrated Circuit)等の従来のプロセッサは、高速演算性能と柔軟性のいずれか一方が不足しているという点で問題である。一方で、動的

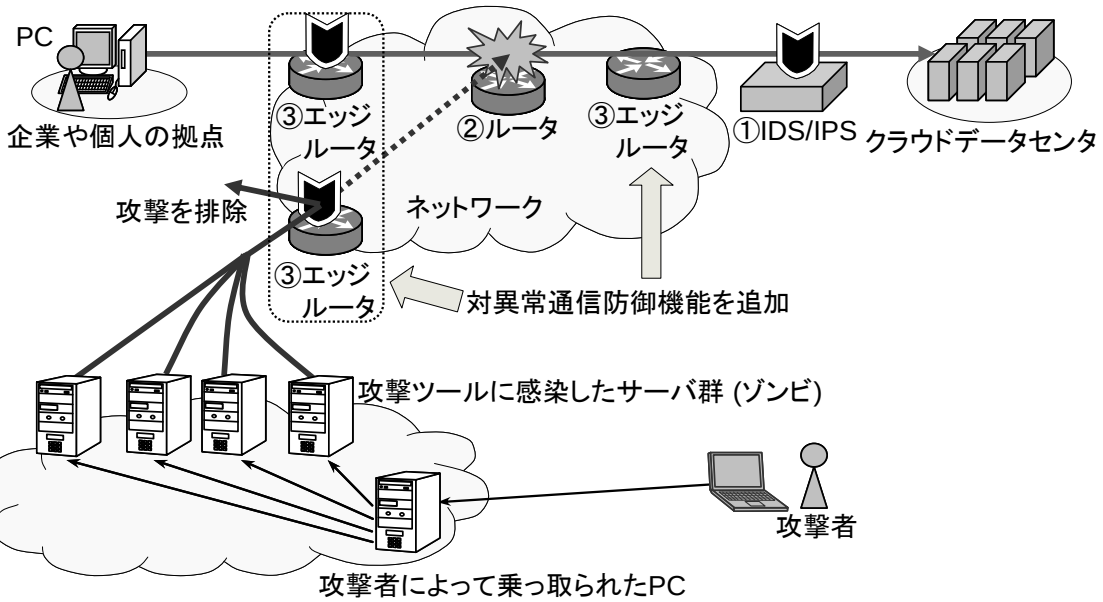


図 10　ネットワークエッジにおける対異常通信防御機能

再構成 LSI[112][113][114][115]は、実行エレメント(Execution Element: EXE)、 遅延エレメント(Delay Element: DE)、メモリエレメント(Memory Element)等の多くの演算エレメント(Processin Element: PE)から成る演算領域を備える。これらのエレメントは、セレクタを経由して相互に接続しており、パイプライン演算によって ASIC 並みの高速演算性能を実現する。加えて、動的再構成 LSI は、実行エレメントの役割や演算エレメント間の接続を 1 クロックサイクルで、あらかじめ設定した回路に変更することが可能であり、高い柔軟性を実現する。そのため、本研究では、短時間で大量のパケットを解析するための高速応答性と共に、攻撃手法の変化に応じて対異常通信防御アルゴリズムを更新できる高い柔軟性を必要とする対異常通信防御機能向けのプロセッサとして、動的再構成 LSI が最適と考えた。

以上の考察を背景に本研究では、動的再構成 LSI を用いて、対異常通信防御の機能を持つネットワークアプライアンスを試作した。試作したネットワークアプライアンスにおいて、パケット全数解析による異常探知方式を用いることにより、広帯域回線において対異常通信防御時間を短縮することを可能にした。また、多種類の異常通信を同時に防御可能なアルゴリズムによる一括防御方式を用いることで、演算リソースを削減することを可能にした。加えて、スループットを維持したまま、対異常通信防御アルゴリズムを無瞬断で更新する無瞬断更新方式を用いることで、パケット廃棄を起こすことなく対異常通信防御アルゴリズムを更新することを可能にした。

シミュレーション評価と実験評価では、対異常防御アルゴリズムを構成するのに必要な実行エレメントの数を見積もり、スループット性能と対異常防御時間を測定した。更に、スループットを劣化させずにアルゴリズムを無瞬断更新可能な時間帯の割合を評価した。

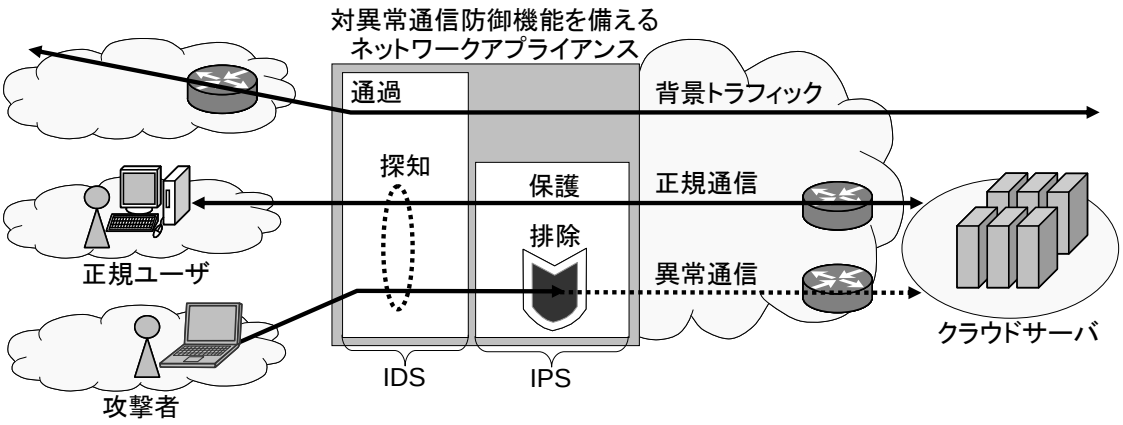


図 11　対異常通信防御機能

以降、3.2 節では、提案する対異常通信防御の概要とアーキテクチャを述べる。3.3 節では、従来の対異常通信防御で用いる手法と問題点を述べる。3.4 節では、提案する対異常通信防御で用いる手法の利点を述べる。3.5 節では、シミュレーションと実験の評価結果を述べる。3.6 節では、提案する対異常通信防御方式についてまとめる。

3.2　対異常通信防御の概要

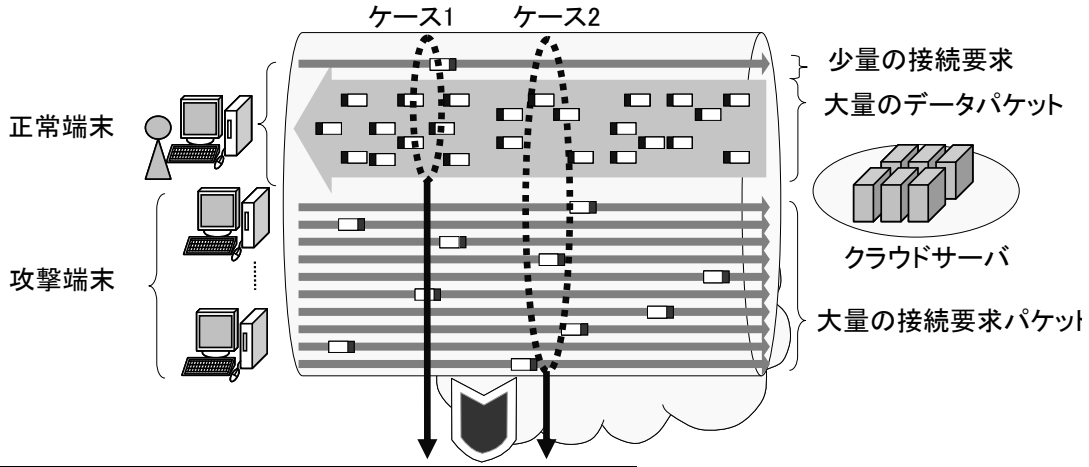
対異常通院防御の機能を持つネットワークアプライアンスは、高速なネットワークで用いるため、毎秒 15M パケット(pps) (10 Gbps 相当)を転送する広帯域回線から高速応答で異常通信を探知する IDS の機能を必要とする。加えて、正規通信を保護しつつ、探知した異常通信のパケットを廃棄したり帯域を制限したりする IPS の機能を必要とする(図 11)。

3.2.1　IDS 部の概要

IDS は、主に 2 つのタイプに分類できる[7]。1 つ目のタイプは、ロジック型攻撃(Land Attack 等)等の探知に適したパターンマッチングに基づくシグネチャ型[6]である。もう一つのタイプは、フラッド型攻撃(SYN flood[12]などの DDoS 等)等の探知に適した多数のパケットを解析することによって得る通信の統計情報に基づくアノマリ型[8][9][10]である。広域ネットワークで、パケット転送装置のシステム障害などを引き起こすのはロジック型攻撃であり、パケット転送機能低下などの問題を引き起こすのはフラッド型攻撃である。

本研究では、DDoS 攻撃等のフラッド型攻撃により、広域ネットワークにおいて、正常通信向け帯域が確保できなくなることが問題と考える。そのため、フラッド型攻撃を探知することに焦点を当てる。そして、フラッド型攻撃を探知するのに適した後者のアノマリ型の異常通信探知機能を採用する。

アノマリ型の異常通信探知機能で使用するアルゴリズムは、筆者らがすでに提案している引例[116]に記載のアルゴリズムを使用する。本アルゴリズムは、パケット数を記録するテーブルを持ち、端末毎に以下に示す集約したデータを記録する。



	ケース1	ケース2
データパケットの返信先数	1	1
接続要求パケットの送信元数	1	11
判定結果	正常	異常

図 12　DDoS の探知

a) 端末がパケットを送信している対向端末の数

b) 端末に向けてパケットを送信している対向端末の数

c) 端末が利用しているポート番号の数

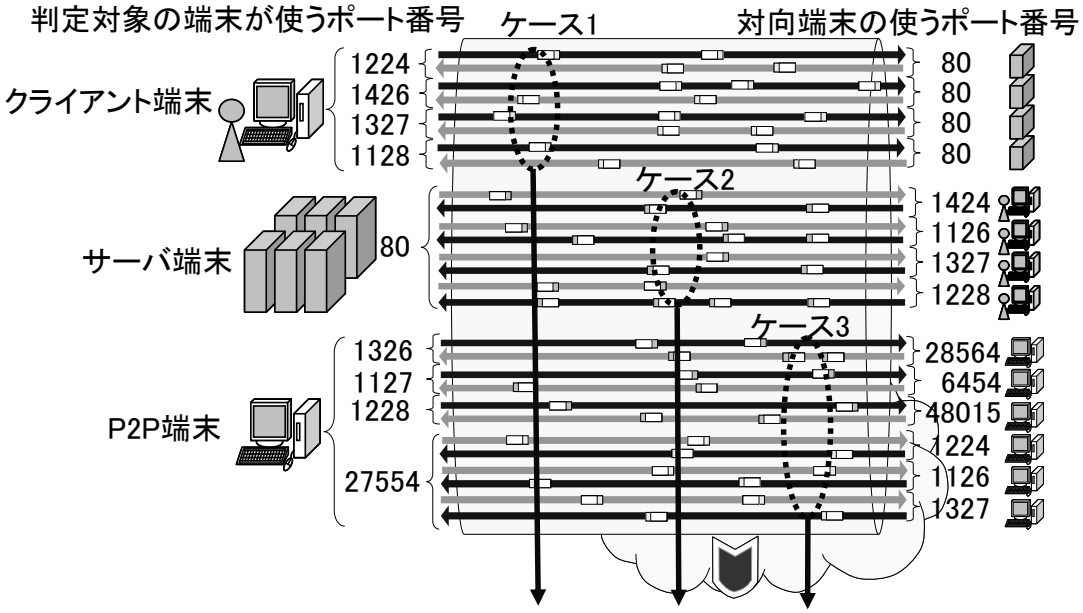
d) その端末と通信している対向端末が利用しているポート番号の数

本アルゴリズムは、a)と b)を用いることで、DDoS やワームを探知する。更に、パケット数の情報と共に c)と d)を用いることで P2P 通信を探知する。以下に、代表的な異常通信の探知の例として、DDoS 攻撃の探知、及び P2P ユーザが送受信する過大通信の探知及び分類方法について示す。

はじめに、DDoS 攻撃の探知方法について述べる。DDoS 攻撃時には、サーバが多数の端末から大量の接続要求を受信する一方で、データパケットを送信しない、という特徴がある。逆に、正規通信は、少量の接続要求に対して、大量のデータパケットを送信する、という特徴がある。このような特徴差に基づき、データパケットの返信先の数と、接続要求パケットの送信元の数の多寡を基準に正常/異常の探知を実施する。図 12 に例を示す。

(1) 正常な状態

サーバが正常な通信を行っている状態では、少量の接続要求パケットを受信して、接続を要求してきた端末に対して、大量のデータパケットを送信する。この状態で、通信情報を抽出すると、データパケットの返信先の数が、接続要求パケットの送信元の数と同等となる(図 12は、正常端末が1台の時、データパケットの返信先の数が1、接続要求パケットの送信元の数が1となる場合を示す)。



	ケース1	ケース2	ケース3
判定対象の端末が使うポート番号の数	4	1	4
対向端末の使うポート番号の数	1	4	6
判定結果	クライアント	サーバ	P2P

図 13 P2P/サーバ/クライアントの分類

(2) DDoS攻撃を受けている状態

サーバがDDoS攻撃を受けている状態では、サーバはランダムな送信元から大量の接続要求パケットを受信する。この状態で、通信情報を抽出すると、データパケットの返信先の数が、接続要求パケットの送信元の数より小さくなる(図 12では、正常端末が1台、攻撃端末が10台の時、データパケットの返信先の数が1、接続要求パケットの送信元の数が11となる場合を示す)。

次に、P2P通信の探知方法について述べる。P2P通信を行っている端末は、多数のポート番号を用いて、多数の対向端末の多様なポート番号との間で、多数のコネクションを確立して通信を行う特徴がある。一方、Webサーバ等のサーバ端末は、少数の常に同一のポート番号を用いて、多数のクライアント端末の多様なポート番号との間で通信を行う特徴がある。反対に、Webクライアント等のクライアント端末は、複数のポート番号を用いて、複数のサーバ端末の常に同一のポート番号との間で通信を行う特徴がある。上記の特徴差に基づき、判定対象の端末の使うポートの数と、対向端末の使うポートの数の多寡を基準に正常か、または異常かの判定を行う。図 13に例を示す。

(a) クライアント端末の通信

クライアント端末は、自分の複数のポート番号と、複数のサーバ端末の固定ポート番号

を使って通信を行う。この状態で、通信情報を抽出すると、対向端末の使うポートの数が、クライアント端末の使うポートの数よりも少なくなる(図 13に、Webクライアント端末の通信相手が4台の時に、Webサーバ端末の使うポート番号の数が1、Webクライアント端末の使うポート番号の数が4となる場合を示す)。

(b) サーバ端末の通信

サーバ端末は、自分の固定ポート番号と、複数のクライアント端末の様々なポートを使って通信を行う。この状態で、通信情報を抽出すると、対向端末の使うポートの数が、サーバ端末の使うポートの数よりも多くなる(図 13は、Webサーバ端末の通信相手が4台の時に、Webクライアント端末の使うポート番号の数が4、Webサーバ端末の使うポート番号の数が1となる場合を示す)。

(c) P2P端末の通信

P2P端末は、自分の固定ポート番号と、対向端末の様々なポート番号を使って通信を行う。同時に、自分の様々なポート番号と、多数の対向端末の様々なポート番号を使った通信も行う。この状態で、通信情報を抽出すると、対向端末の使うポート番号の数と、P2P端末の使うポート番号の数の間に大きな差異が無くなる(図 13では、P2P端末が自分の固定ポート番号を用いて通信する対向端末が3台、対向端末の固定したポート番号を用いて通信する通信相手が3台の時に、対向端末の使うポート番号の数が6、判定対象のP2P端末が使うポート番号の数が4となる場合を示す)。

上記のアノマリ型異常通信探知アルゴリズムを用いることで、異常/正常の判定や、通信の種類の分類などが可能となる。

3.2.2 IPS 部の概要

IPS 部は、IDS 部の探知結果に基づいて、通信に対して様々な制御を行う。ワーム通信のパケット廃棄、P2P 通信の帯域制限、DDoS 攻撃のパケット排除、正規通信の保護を行う。

一例として、IPS 部が DDoS 攻撃を受けた時に、正規通信を保護する方法を図 14 に示す。クライアントから TCP の接続要求を行う SYN パケットを受け取ると、まず、ランダムな値 B を送信シーケンス番号(SEQ 番号)に付与した SYN-ACK パケットを返信する(図 14 ①)。SYN-ACK パケット返信後に、受信シーケンス番号(ACK 番号)が B+1 である ACK パケットを受信すると図 14 ②)、正規通信と判定して TCP コネクションを確立する[117]。クライアントとの間で TCP コネクションが確立したら、次にサーバとの間で TCP コネクションの確立を開始する。サーバから受け取った SYN-ACK パケットが記載する SEQ 番号の値 C(図 14 ③)を用いて、データパケットの SEQ 番号や ACK 番号の変換を行う(図 14 ④⑤)。

上記のように、対異常通信防御装置の IPS 部は、往路と復路のパケットのシーケンス番号の整合性に基づいて、正常か異常かを判定して、TCP コネクションを継続するか否かを決定する。高機能な IPS では、TCP のような通信レイヤだけでなく、アプリケーションレイ

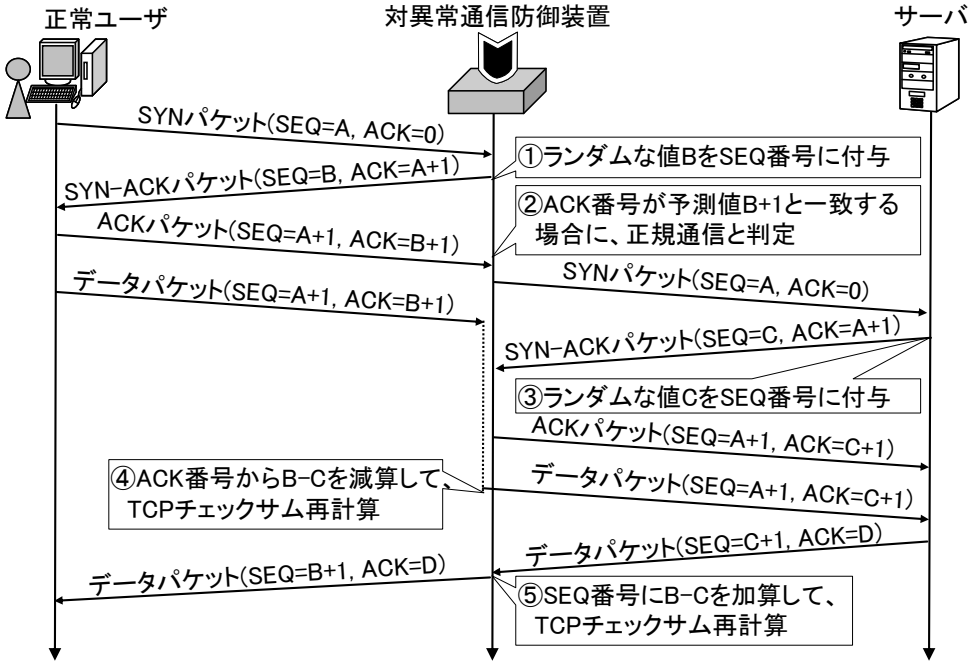


図 14　正規通信の保護

ヤでも、整合性に基づく正常か異常かの判定を行う。

3.2.3　提案する対異常通信防御装置のアーキテクチャと目標性能

本研究において提案する対異常通信防御機能は、図 15 および表 1 に示すように、パケットデータ抽出/転送部，対異常通信防御部，アルゴリズム更新部の 3 つの部分から成る。

パケットデータ抽出/転送部が、入力パケットから解析に必要なデータ(送受信 IP アドレス、送受信ポート番号、パケット長、TCP フラグ、シーケンス番号、等)を抽出して、対異常通信防御部に入力する。対異常通信防御部へ入力するデータを、解析に必要なデータだけに絞りこむことで、高スループット性能を実現する。対異常通信防御部は、動的再構成 LSI とメモリから構成され、パケットから抽出したデータを用いて、通信やセッションの統計情報を作成してメモリに記録する。メモリアクセスの頻度はパケットの到着レートに比例する。更に、統計情報に基づいて、異常/正常を判定して出力する。パケットデータ抽出/転送部は、判定結果に基づいて、パケットの廃棄や、SEQ/ACK 番号の変換を行い、パケットを出力する。アルゴリズム更新部は、対異常通信防御部のアルゴリズムの更新を行う。

更に、本研究では、下記①～③の性能を持つ対異常通信防御の実現を目標とした。

① 10Gbps 回線での 1 秒以内での対異常通信防御

対異常通信防御装置は、異常通信による長期間のネットワーク帯域資源の浪費を防止することを目的として用いる。そのため、大容量回線(10Gbps)を流れる大量のパケットを解析して、高速応答(1秒以内)にて異常通信を探知して防御する高速演算性能を必要とする。

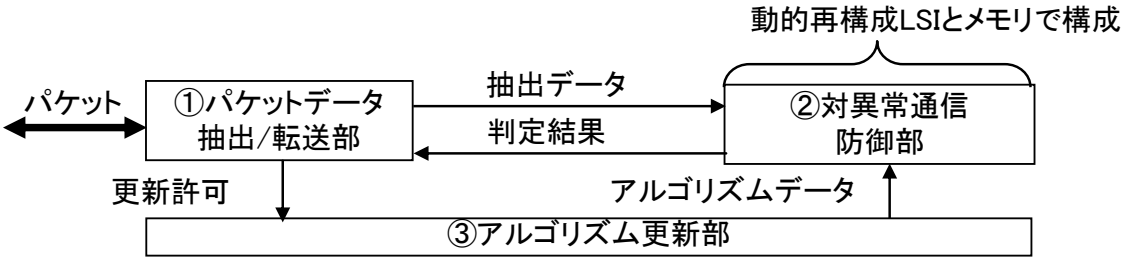


図 15 対異常通信防御機能の構成要素

表 1 対異常通信防御機能の構成部

#	構成部の名称	構成部の役割
1	パケットデータ抽出/転送部	ネットワーク回線を流れるパケットデータから、対異常通信防御に必要なデータを抽出する。更に、通過/保護と判定した通信のパケットに必要な処理(廃棄、SEQ/ACK番号の変換等)を施して転送する。
2	対異常通信防御部	入力した抽出データに基づき、通信やセッションの統計情報を作成してメモリに記録する。更に、メモリに記録した通信やセッションの統計情報に基づいて異常通信を探知して、通過/保護/廃棄の判定を行い、判定結果を返信する。
3	アルゴリズム更新部	パケットデータを廃棄したりスループットを低下させたりせずに、対異常通信防御機能部のアルゴリズムを更新する。

② 高速ネットワークにおける 3 種類の異常通信の探知と防御 (DDoS, ワーム, or P2P)

高速ネットワークでは、様々な異常通信によるパケットが流れる。それゆえ、対異常通信防御装置は、たとえ防御すべき異常通信の数が増加したとしても、それらの全てを防御せねばならない。

③ パケット廃棄やスループット劣化の無いアルゴリズムアップデート

対異常通信防御装置は、新手の異常通信からシステムを防御するために、古いアルゴリズムを新しいアルゴリズムに更新する際に、ネットワークサービスの停止を回避するため、パケット廃棄やスループット劣化を起こすことなく対異常通信防御アルゴリズムをアップデートできねばならない。

これらの性能目標を実現するために、パケットデータ抽出/転送部と、対異常通信防御部と、アルゴリズム更新部に対して、3 つの方式を提案した。従来手法と提案手法の詳細は、3.3 節および 3.4 節にて述べる。

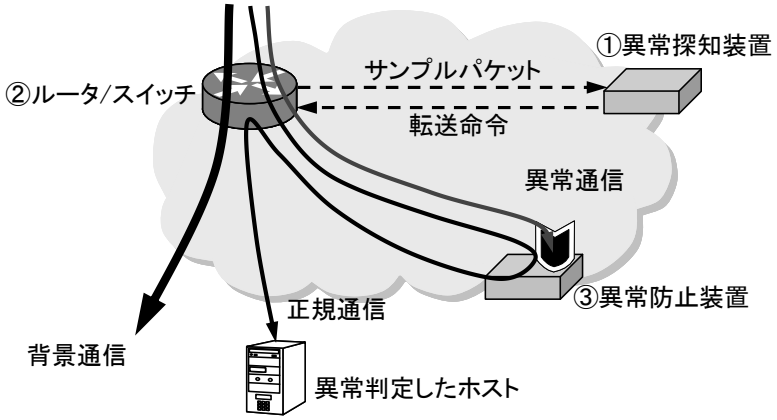


図 16　従来の大容量回線対応の対異常通信防御方式

3.3　従来の対異常通信防御

従来の対異常通信防御では、パケットデータ抽出、対異常通信防御、アルゴリズム更新のそれぞれに対して、以下の方式を用いており、3.2 節に記載した性能目標を達成する際に問題があった。

- 標本解析による異常探知
- 異常通信毎に特化した防御
- 入力バッファを用いたアップデート

3.3.1　標本解析による異常探知

対異常通信防御機能は、パケット抽出/転送部(図 15)において回線を流れるパケットから解析に必要なデータを抽出して、対異常通信防御部(図 15)においてメモリに記録した通信やセッションの統計情報に基づいて異常通信を探知して防御する。対異常通信防御部が、パケットから抽出したデータを用いて、通信セッションの統計情報を作成する際には、大量のメモリアクセスを必要とし、時間がかかる。そのため、高速ネットワークで使用する10～100Gbps以上の高速回線において、短時間で異常通信を探知するためには、解析するパケット数を標本化するなどして減らし、通信やセッションの統計情報の作成にかかる時間を減らすことが効果的である。

既に、sFlow[18]やネットワークプロセッサを用いた標本パケット解析に基づく対異常通信防御方式[116][118]の報告がある。本方式は、パケットデータ抽出/転送部(図 15①)において標本化したパケットデータを抽出して、異常通信を探知する。本方式を用いた対異常通信防御の一例を図 16 に示す。本システムは、次の 3 つのノードから成る。

IP		ポート番号		SYN 受信 状態	SYN-ACK 送信 状態	Window Scale オプション	MSS オプション	防御対象端末 のシーケンス番号	対向端末の シーケンス番号	開始 時刻	最終 時刻
防御対象 端末	対向 端末	防御対象 端末	対向 端末								
10.0.0.4	10.0.1.5	80	1045	1	1	8	1460	0x44444444	0x55555555	10:09	10:10
10.0.0.5	10.0.1.6	80	1245	1	1	4	1380	0x76543210	0x12345678	08:10	08:15
10.0.0.6	10.0.1.7	80	1423	1	1	2	1446	0x23456789	0x87654321	12:45	19:50

TCPコネクションの特定　3-Way-Handshake　TCPオプション　シーケンス番号　時間

図 17　専用テーブルの例（SYN フラッド DDoS 攻撃用）

① 異常探知装置(Intrusion Detection System: IDS)

サンプルパケットに基づいて異常通信を行っているホストを判定する。更に、異常通信を行っているホストの通信が常にIPSを経由するように、ルータの設定変更を行う。

② 標本化機能を持つルータ/スイッチ

ネットワーク回線を流れるパケットをsFlow[18]等をを用いてサンプリングして、抽出したパケットデータを異常探知装置へ送信する。また、異常探知装置が異常と判定したホストが送受信するパケットを異常防止装置に転送する。

③ 異常防止装置(Intrusion Prevention System: IPS)

異常と判定したホストが行う通信の中には、異常通信と、正規通信が含まれる。本システムは、異常と判定したホストが送受信する全パケットについて詳細解析を行い、異常通信を遮断しつつ、正規通信のみを通過させる。

本従来方式は、サンプリングによって最小化したパケットデータを解析して異常探知を行うため、汎用 CPU のような演算性能が低く、低価格なプロセッサでも大容量(10Gbps 以上)回線に対応した異常通信探知を容易に実現できる、という利点があった。一方で、サンプリングによる統計誤差の影響で、異常探知までの時間が長く変動量も大きく(30 秒程度(3.5 節))、3.2 節記載の①の目標時間を満たさないという問題があった。また、異常と判断したホストの異常通信の排除または帯域制御と、正規通信の保護を行う機能を、IPS に実行してもらう必要があり、一つの装置で実現出来ず、装置数が増大するという問題があった。

3.3.2　異常通信毎に特化した防御

従来の対異常通信防御は、異常通信の種類(DdoS / Worm / P2P)毎に専用の演算部とテーブルを用意して搭載する方式を用いていた。例えば、DDoS 専用の対異常防御装置[119]では、DDoS 攻撃を短時間で探知するために、探知に必要な情報を整理した形でテーブルに記録する。図 17 に、ファイアウォール等で一般的に用いられているステートフルインスペクション[120]を実施する場合に必要となる、Web サーバ宛 SYN フラッド型 DDoS 攻撃探知用ステートテーブルの例を示す。本例では、TCP コネクションを特定する防御対象端末と対向端

図 18　従来の異常通信毎に特化した防御

表 2　対異常通信防御アルゴリズムの構成要素

#	構成要素名	構成要素の役割
1	探知演算部	各通信に対して異常/正常を判定
2	通信統計テーブル	異常/正常判定に用いる通信統計情報を記録
3	防止演算部	異常と判定した端末がやり取りするパケットデータを詳細に解析して、遮断/通過を判定
4	セッション統計テーブル	遮断/通過判定に用いるセッショ統計情報を記録

末の IP アドレス及びポート番号と、シーケンス番号と、TCP の 3-way-handshake [121]の手続きに関する SYN や SYNACK パケットの有無と、通信開始時刻・最終時刻と、クライアントが指定してきた TCP オプションに関する Window Scale、MSS (Maximum Segment Size) のステートを記録する。なお、SYN cookie[122]と呼ばれる防御手法では、図 17 のようなテーブルを持たずに防御することが可能であるが、クライアントが指定してきた TCP オプションに関する情報が保持できない、という問題点がある。

探知する異常通信の種類毎に専用テーブルを作成する方式は、防御対象及び探知すべき異常通信を予め限定している状況で、精度よく短時間で異常通信を抽出する場合に有効である。

図 18 には、本研究で試作した対異常通信防御アルゴリズム(詳細は 3.4.2)を、対 DdoS / Worm / P2P 用に特化した 3 つのアルゴリズムに分割して搭載する方法を示す。各アルゴリズムは、図 18 と表 2 に示した 4 つの部分から成る。

異常通信毎に特化した従来方式では、異常通信毎に専用の探知演算と防止演算を実施する。そのため、対異常通信防御の設計と、各テーブルに記載する統計情報の作成方法が容易となる。その一方で、異常通信の種類毎に異常探知処理と異常防止処理を行うので、3.2 節記載の目標 2)を満たすために防御すべき異常通信の種類を増加させると、アルゴリズム演算部を構成する実行エレメントの数が増大する(3 種類の異常通信に対応する場合、演算

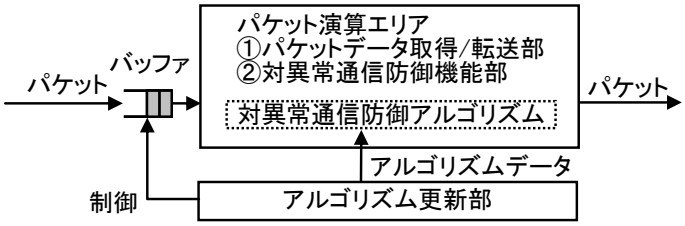


図 19　入力バッファを用いたアップデート

部の数 6、テーブル数 6)問題があった。

3.3.3　入力バッファを用いたアップデート

従来の対異常通信防御装置は、図 19に示すように、パケットデータ抽出/転送部の前段に、バッファを備える。対異常通信防御機能部(図 19 ②)に搭載する古いアルゴリズムを新しいアルゴリズムに変更する場合は、パケット演算エリアが演算中のパケットデータを全て外部に出力するまで、新たに受信したパケットデータをバッファに蓄積する。その後、アルゴリズム全体を、一括して更新していた[123][124]。これにより、入力データを失うことなくアルゴリズムを更新することができる。

一方、上記の従来方式では、アルゴリズムの変更部分が全体の一部であり、そのアルゴリズムが必要とするデータがパケットの一部であっても、新たに受信した全てのパケットデータをバッファに蓄積して演算を停止せねばならない。そのため、アルゴリズム更新時にスループットが劣化して[123]、3.2 節記載の目標 3)を満たさないという問題があった。

3.4　提案する対異常通信防御の方式

本研究では、従来方式の問題を解決して、3.2 節記載の技術的目標を満たすため、パケットデータ抽出/転送部、対異常通信防御部、アルゴリズム更新部の各々に対して、以下のような 3 つの方式を用いることを提案する(表 3)。

① パケット全数解析
② 様々な異常通信の一括防御方式
③ アルゴリズムの無瞬断更新方式

これら①②③の提案方式の詳細と、提案方式を用いた場合の効果について、以下に詳細を説明する。

3.4.1　パケット全数解析

対異常通信防御装置は、データ量の多い大容量(目標10Gbps)回線を流れる大量のパケットを解析して異常通信を短時間(目標1秒以内)に探知して遮断するための高速演算性能だけで

表 3　従来方式と提案方式の比較

	パケットデータ抽出/転送部	対異常通信防御部	アルゴリズム更新部
目標性能	大容量(10Gbps)回線対応	多(3)種類の異常防御	スループット一定 無瞬断更新
従来方式	標本パケット解析	異常通信毎に 特化した防御	入力バッファを用いた アップデート
課題	探知＆防御時間の増加	実行エレメント数の増大	スループット劣化
提案方式	パケット全数解析	様々な異常通信の 一括防御	アルゴリズム 無瞬断更新
効果	大容量回線にて 短時間での防御	省実行エレメント数にて 多種類の対異常防御	スループット無劣化・ パケット無廃棄での アルゴリズムアップデート

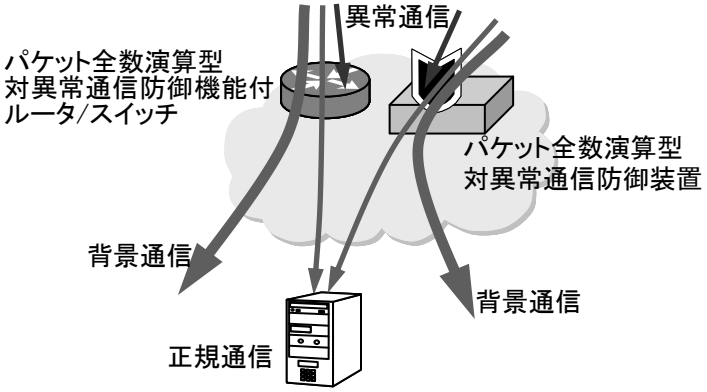


図 20　パケット全数演算型の対異常通信防御方式

なく、新手の攻撃に対応するために異常通信防御アルゴリズムをアップデート可能な柔軟性も必要とする。

このため本研究では、高速演算性能と柔軟性を併せ持つ動的再構成LSIを用い、パケット全数演算に基づく対異常通信防御方式(図 20)を提案する(目標の10Gbps回線にて毎秒15Mパケット演算)。

本方式は、パケット全数解析を用いているため、探知時間が異常探知向けのパケット数の閾値に依存し、異常探知における統計誤差が無いため、短く一定の時間での対異常防御が可能という利点がある。

その一方で、パケット全数解析は、高速なパケット解析を必要とする。また、パケット全数解析は、プロセッサ内部のキャッシュメモリやプロセッサ外部の SRAM (Static Random Access Memory) / CAM (Content Addressable Memory)等の高速なランダムアクセスが可能なメモリを使用するため、標本パケット解析よりも使用可能なメモリ量が少なく、見逃し率(異常通信を正常と判定する確率)が高くなる可能性がある。そのため、本研究では、動的再構成 LSI を用いることによるパケット解析性能だけでなく、見逃し率についても評価をおこ

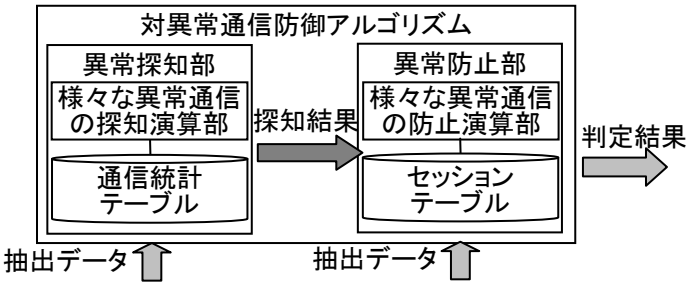


図 21 様々な異常通信の一括防御

なった（評価結果は 3.5 節で記述）。

3.4.2 様々な異常通信の一括防御

対異常通信防御アルゴリズムを実行する動的再構成 LSI の実行エレメントの数には、チップ面積に依存して上限がある。そのため、対異常通信防御アルゴリズムを実行する動的再構成 LSI の実行エレメントの数は、防御すべき異常通信の種類が増加(目標 DDoS / Worm / P2P の 3 種類)したとしても、一定数(目標エレメント数 672)以下に抑える必要がある。

本研究では、図 18 のように異常通信の種類(DDoS / Worm / P2P)毎に特化していた異常探知演算部，通信統計テーブル，異常防止演算部，セッションテーブルを、図 21 に示すように、幾つかの汎用的な演算部とテーブルに集約して、様々な異常通信を一括して探知および防止する構成とした。

提案方式では、演算部の数とテーブル数が減少し、防御すべき異常通信の種類数に依存せずに、一定数となる(演算部の数 2、テーブル数 2)。そのため、防御すべき異常通信の種類が増加(目標 DDoS / Worm / P2P の 3 種類)しても、実行エレメントの数を小さい値(目標数 672 以下)に保つことが容易となる。その一方で、探知と防御の設計や通信統計テーブルの作成方法が複雑化する。

3.4.3 アルゴリズムの無瞬断更新

対異常通信防御装置は、アルゴリズム更新時にパケット廃棄、およびスループット劣化の無いアルゴリズム更新を実現する必要がある。

このため、本研究では、再構成回路を通過する抽出データが無い時に、演算部のアルゴリズムを更新することで、パケット廃棄、およびスループット劣化の無いアルゴリズム更新を可能とするアルゴリズムの無瞬断更新方式を提案する。

図 22 を用いて提案方式の例を示す。本例では、3.2.3 節記載のパケットデータ抽出/転送部、対異常通信防御部およびアルゴリズム更新部は、図 22 と表 4 に示す部分から成る。

提案方式では、演算に使用するデータがパケットヘッダのように入力パケットデータの一部である場合、演算に不必要なペイロードデータが通過している時は、再構成回路へパケットデータを分配しない(図 23)。

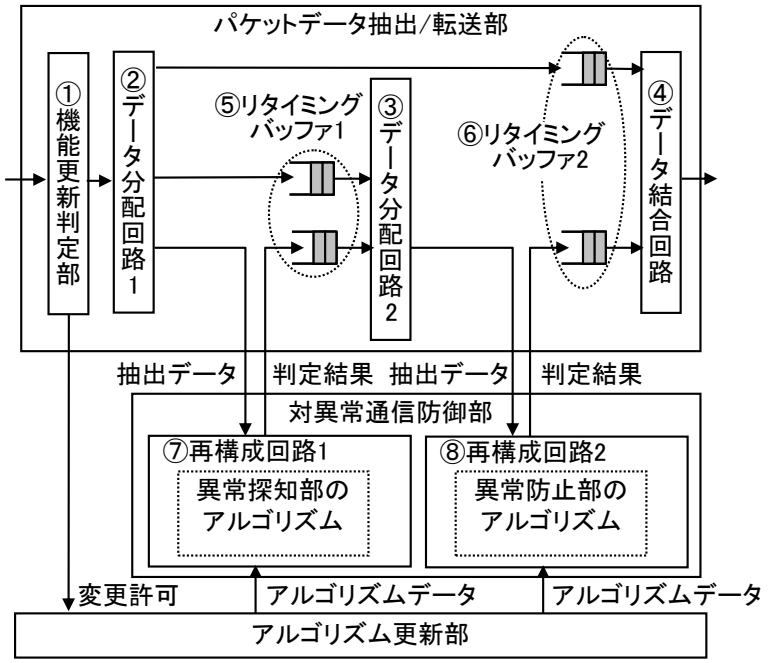


図 22　アルゴリズムの無瞬断更新を実現する構成

表 4 無瞬断更新を実現する構成

#	構成部の名前	構成部の役割
1	機能更新判定部	演算に不必要なデータ入力が一定時間継続するか否かの判定
2	データ分配回路 1	受信パケットをペイロード、ヘッダの一部などに分割して出力
3	データ分配回路 2	再構成回路 1 やデータ分配回路 1 からの出力データを、前段バッファを使用してタイミングを揃えて結合して出力
4	データ結合回路	再構成回路 2 やデータ分配回路 2 からの出力データを、前段バッファを使用してタイミングを揃えて結合して出力
5	リタイミングバッファ 1	データ分配回路 1 と再構成回路 1 からのデータを出力する時間を調整
6	リタイミングバッファ 2	データ分配回路 1 と再構成回路 2 からのデータを出力する時間を調整
7	再構成回路 1	異常探知部のアルゴリズム搭載
8	再構成回路 2	異常防止部のアルゴリズム搭載

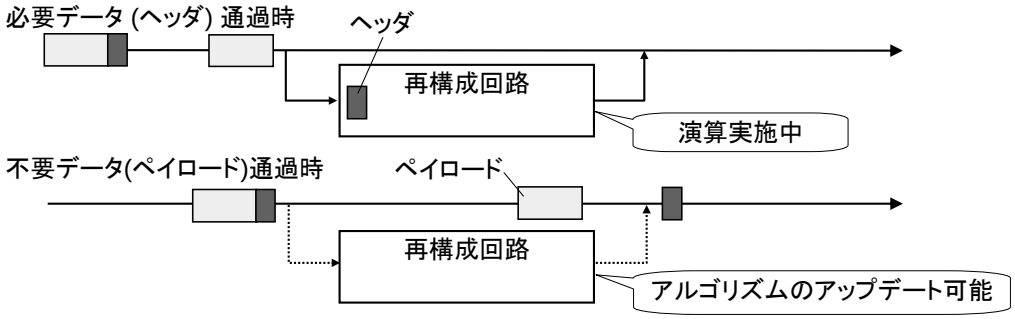


図 23 必要/不要データ通過時のデータの通り道

演算に不必要なペイロードデータが通過している間は、データが再構成回路を通過しない無演算時間が発生する。この時間を利用してアルゴリズムの更新を行うことで、パケットデータを失うこともなく、パケットデータを一時的に堰き止めてスループットを劣化させることもなく、アルゴリズムを更新することができる。

一方、回線帯域の利用率が高くなる程、演算に必要なパケットヘッダデータの通過する間隔が短くなる。その結果、再構成回路に搭載するアルゴリズムのアップデートが可能な時間帯の割合が小さくなる。アップデート可能な時間帯の割合がどの程度まで小さくなるのかを評価するため、本研究では、スループットと、アルゴリズムのアップデートが可能な時間帯の割合の関係を評価した(評価結果は、3.5 節に記載)。

3.5 評価実験

3.5.1 シミュレーション評価

対異常通信防御を行うために必要な回路量と、対異常通信防御時間と、アルゴリズムの無瞬断更新の実現性を評価するため、シミュレーションを実施した。3 種類の異常通信(DDoS、ワーム、P2P)を探知して制御する対異常通信防御機能を、168 個の 32 ビットの実行エレメントを 166MHz で動作させる動的再構成 LSI [112]を用いて設計した。シミュレーション評価では、図 24 に示すシミュレーション環境を用いた。パケットデータ抽出部は、C 言語プログラミング環境におけるシミュレーションにて、入力パケットに基づいて抽出データを作成する。対異常通信防御部は、設計ツール[125]上におけるシミュレーションにて、パケットデータ抽出部が作成した抽出データに基づいて判定結果を出力する。ランダムな送信元 IP アドレスと宛先 IP アドレスを持つ背景通信パケットと、DDoS、ワーム、P2P 等の異常通信パケットをパケットデータ抽出部へ最大 83Mpps(最小フレーム長である 64 バイト長パケット換算で 56Gbps 相当)で入力した(図 24)。対異常通信防御部は、抽出データに基づいて判定結果を出力した。

最初に、異常通信毎に特化した防御を行う従来方式と、様々な異常通信を一括して防御を行う提案方式のそれぞれを設計した。更に、シミュレーション環境上において、パケット解析速度と、対異常通信防御を構成するために必要な実行エレメント数を評価した。パケット解析速度と実行エレメント数の評価結果を表 5 に示す。

シミュレーション評価の結果から、一括防御する提案方式を用いた場合、20Mpps のスループットを実現する対異常通信防御機能が、332 個の実行エレメント(2 個の動的再構成 LSI に相当)の使用により実現できた。一方、特化防御する従来方式を用いた場合、同一性能を実現する対異常通信防御機能が、435 個の実行エレメント(3 個の動的再構成 LSI に相当)の使用により実現できた。

一括防御する提案方式は、テーブル作成を行う演算回路を複数の異常通信の間で兼用することで、特化防御する従来方式と比較して、実行エレメント数を 24%削減できることを確認した。

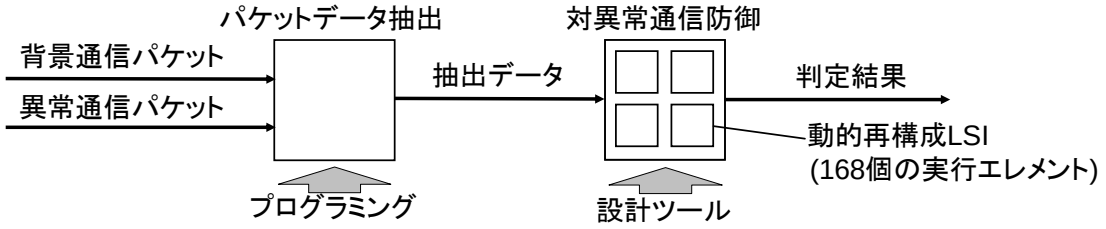


図 24　シミュレーション環境

表 5　従来方式と提案方式が必要とする実行エレメント数とパケット解析速度の比較

防御方式	実行エレメントの数（32bit, 166MHz）	動的再構成 LSI の数	メモリ	セッションテーブルのスケーラビリティ	パケット解析速度
一括防御	332	2	オンチップ	No	20Mpps
特化防御	435	3	オンチップ	No	20Mpps
一括防御	432	3	オンチップ	No	83Mpps
一括防御	600	4	チップ外付	Yes	83Mpps

また、一括防御する提案方式を用いて、実行エレメント(3 個の動的再構成 LSI に相当)の使用数を 432 個に増やすと、図 25 に示すように、抽出データの入力ビット幅を 32 ビットから 96 ビットに増加させて演算の並列度を高めることが可能となり、83Mpps のスループットを実現することが可能となった。

対異常通信防御部の構成を図 26 に示す。異常通信の探知演算部、通信統計テーブル、異常通信の防止演算部、セッションテーブルから成る。

通信統計テーブルは､32 ビット×4K エントリの 16KB のメモリエレメント 13 個を用いて、2 つの IP アドレスおよび 2 つのポート番号の間で行われている通信毎のパケット数と、1 つの IP アドレスで行われている通信毎のパケット数と対向端末数と使用するポート番号の数を記録する。1 パケットから抽出した 1 つの抽出データにつき、32 ビット×13 の単位で 1 回ずつ読み書きを行う。

異常通信の探知演算部は、通信統計テーブルに記録した統計情報に基づいて、DDoS、ワーム、P2P などの異常通信を探知して、探知結果や帯域利用状況を防止演算部へ出力する。

探知防止部は、DDoS 攻撃と判定した抽出データを用いて、セッションテーブルの更新を行い、DDoS と判定したパケットの通過／遮断を判定する。ワームと判定したパケットについては遮断を行い､P2P と判定したパケットについては帯域利用状況に基づいて通過／遮断を判定する。

セッションテーブルは、32 ビット×4K エントリの 16KB のメモリエレメント 6 個を用いて、2 つの IP アドレスおよび 2 つのポート番号の間で行われている通信毎に、シーケンス番号とセッション確立状態を記録する。1 パケットから抽出した 1 つの抽出データにつき

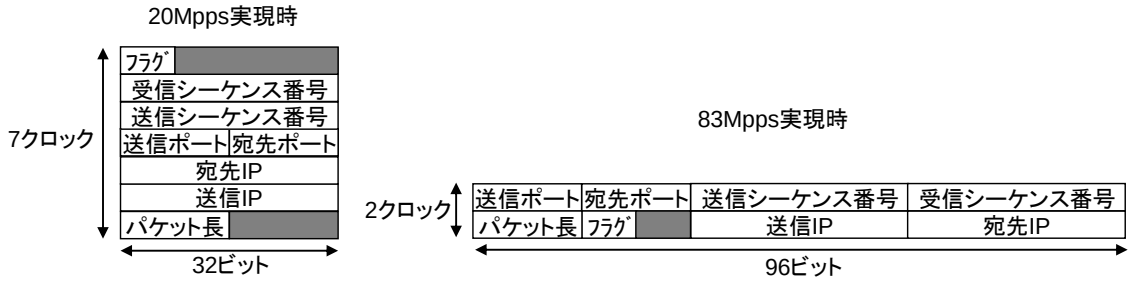


図 25　抽出データの対異常防御部への入力方法

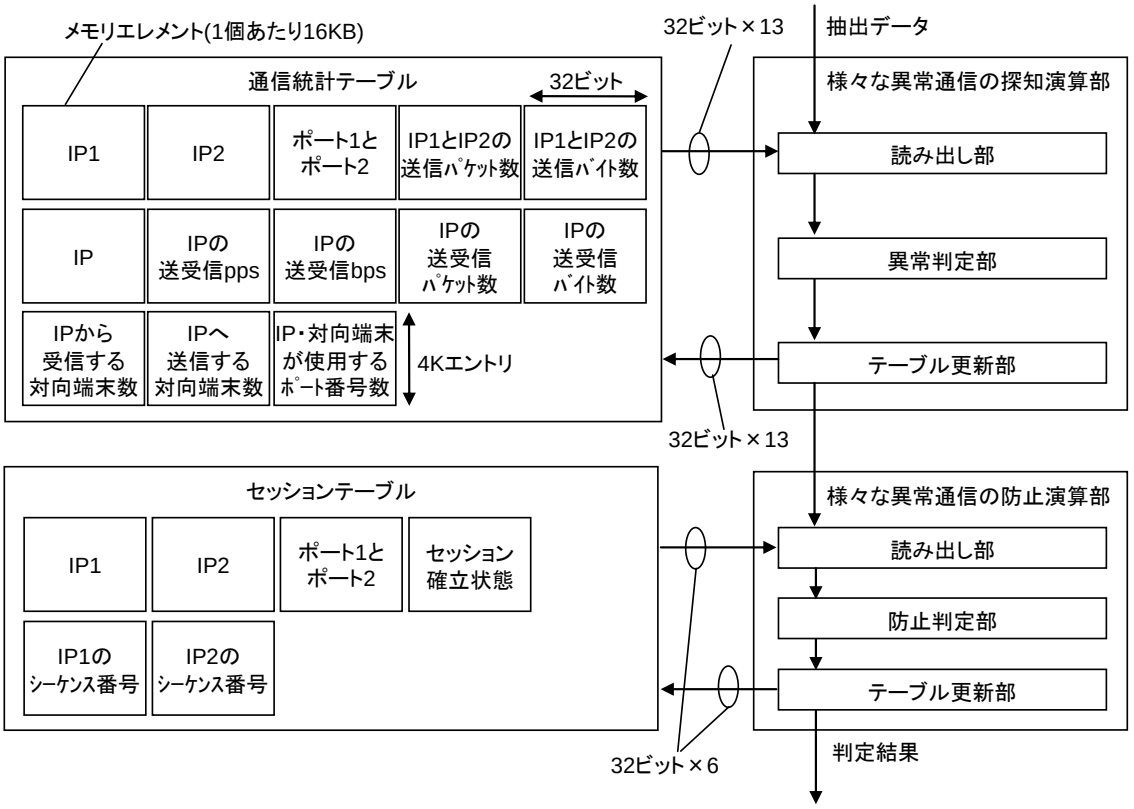


図 26　対異常通信防御部の構成図

32 ビット×6 の単位で 1 回ずつ読み書きを行う。

動的再構成 LSI を 4 個使用した場合は、外付メモリへのアクセス帯域を用意することが可能となる。異常通信の防止に使用するセッションテーブル向けに外付メモリを使用することで、エントリ数を 4K から大きく増加させることが可能となる。

以上の結果により、高速回線(10 Gbps 回線)上におけるパケット全数解析を実現できる見通しを得た。

次に、実行エレメント 432 個(表 5)から成る対異常通信防御機能を用いて、異常通信の帯域 B を変化させたときの対異常通信防御時間 T(異常通信の開始から防御完了までの時間)をシミュレーション評価した。異常通信探知の閾値は、異常通信の帯域 B に応じて、100ms

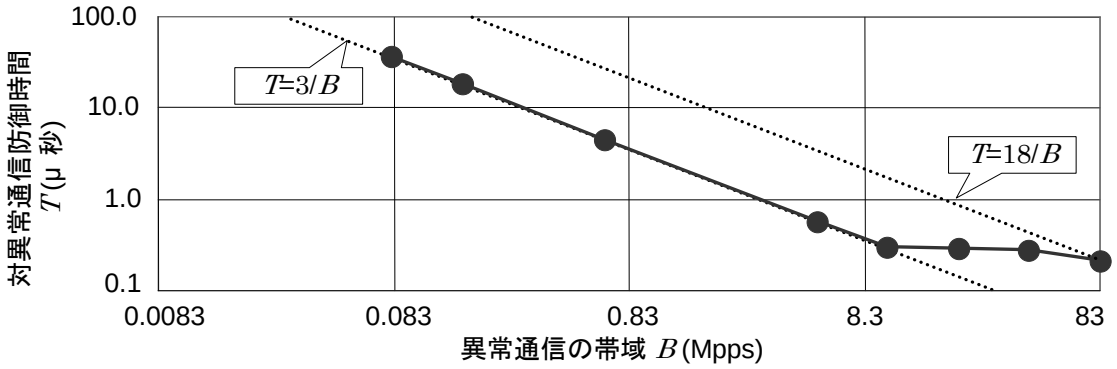


図 27　対異常通信防御時間の評価

あたり 3 から 18 パケットにセットした。異常通信(DDoS)の帯域 *B* と対異常通信防御時間 *T* の関係を図 27 に示す。

異常通信の最初の 3～18 個のパケットの解析が完了すると、異常通信の排除が始まるため、対異常通信防御時間 *T* は、3～18/*B* 秒となった。異常通信(DDoS)の帯域 *B* が 83kpps 以上であれば、36μ 秒以下の短時間での対異常通信防御が実現できた。これにより、動的再構成 LSI を用いたパケット全数解析は、パケット数の閾値に依存しつつも、短時間(1 秒以内)での対異常通信防御を可能にすることを確認した。

最後に、無瞬断更新する提案方式を用いて、回線の帯域利用率と、アルゴリズムを無瞬断で更新可能な時間帯の割合の関係を評価した。図 28 に示す環境を用いて、各回線において引例[126]のパケット長の分布を用いてパケットを発生させることで、シミュレーション評価を行った。各回線におけるパケットは、負指数分布に従う間隔で到着した。MAC アドレスを含むパケット長が 1514 バイトの場合、プリアンブルとインターフレームギャップと FCS (Frame Check Sequence)を含めた到着間隔は 1230 ns となる。設計した対異常通信防御部をデータが通過する時間は 852 ns (142 クロックサイクル相当)であり、アルゴリズムを変更するために必要な時間は、102 ns (17 クロックサイクル相当)であった。

10Gbps × 4 回線における帯域利用率とアルゴリズム無瞬断更新が可能な時間帯の割合の関係を図 29 に示す。

アルゴリズムを無瞬断で更新可能な時間帯は、帯域利用率が増加するに従い減少するが、帯域利用率が 100%のときでも、全時間帯の 1500 分の 1 (1 時間に 24M 回のアップデートが可能)を占めることを確認した。これは、データ通過時間とアルゴリズム更新時間を合わせた時間が 954ns となるのに対して、フルサイズ(MAC アドレスを含むパケット長が 1514 バイト)のパケットの到着間隔は 1230 ns であり、4 つの回線でフルサイズのパケットが同じ時間帯に到着すれば、アルゴリズムを無瞬断で更新可能な時間が発生するためである。

上記評価から、無瞬断更新する提案方式は、実運用上問題ないことを確認した。

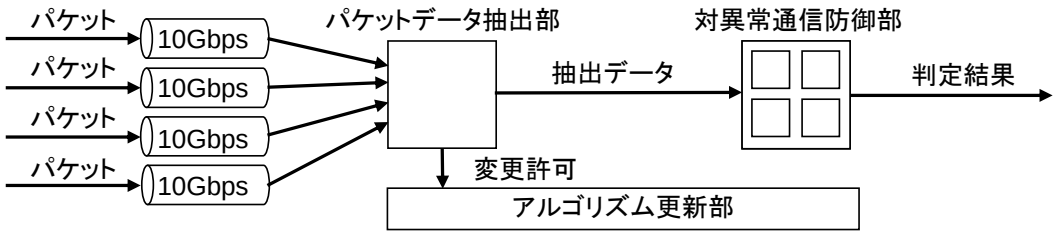


図 28　無瞬断更新の評価環境

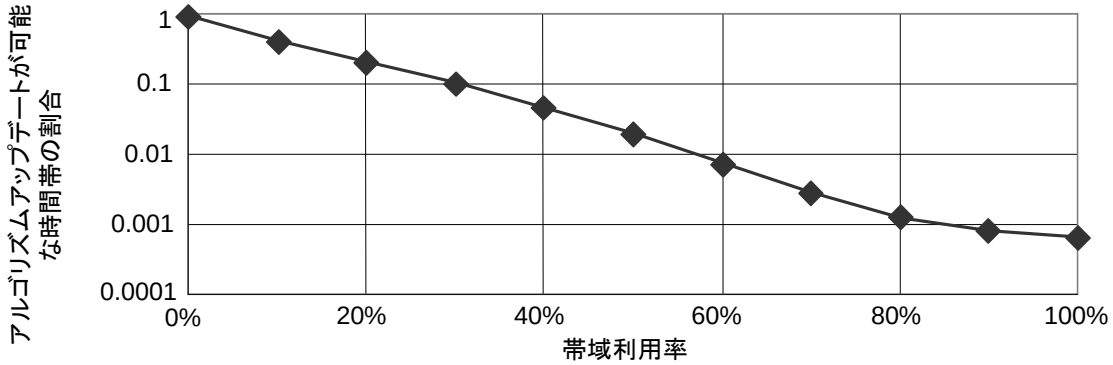


図 29　アルゴリズムアップデートが可能な時間帯の割合と帯域利用率の関係

3.5.2　実験評価

提案する対異常通信防御が、実環境で想定通り動作することを確認するために、実験評価を行った。実験評価では、2 個の動的再構成 LSI を搭載した端末を用いて、実行エレメント 332 個(表 5)から成る対異常通信防御機能を動作させた。実験評価で使用した構成を図 30 に示す。

トラフィック生成装置は、背景通信としてランダムな送信 IP アドレスと宛先 IP アドレスを持つパケットを最大 6Mpps(最小フレーム長 64 バイト換算で 4Gbps 相当)にて送信した。PC1-6 は、Web サーバ、Web ブラウザ、P2P、DDoS 攻撃、ワームなどを模擬するアプリケーションを動作させた。対異常通信防御装置は、異常通信を探知すると、異常通信のパケットを廃棄したり、帯域を制御したりしつつ、正規通信を保護する。遅延発生装置#1,2 は、ネットワークエッジでの対異常通信防御を模擬するために、50, 200ms の RTT を生成した。(図 30)。50ms の RTT を生成することで無線網などのアクセス回線を模擬し、200ms の RTT を生成することで WAN などの広帯域回線を模擬した。

動的再構成 LSI と汎用プロセッサのパケット解析演算性能の比較結果を表 6 に示す。動的再構成 LSI の演算性能は、汎用プロセッサよりも 300 倍高速であることが分かった。

次に、動的再構成 LSI を使用したパケット全数解析と、汎用プロセッサを使用したパケット標本解析(サンプリングレート : 1/2048)を用いて、対異常通信防御時間(異常通信の開始から実際に防御するまでの時間)を評価した。PC3 が PC2 から DDoS 攻撃やワーム攻撃を受

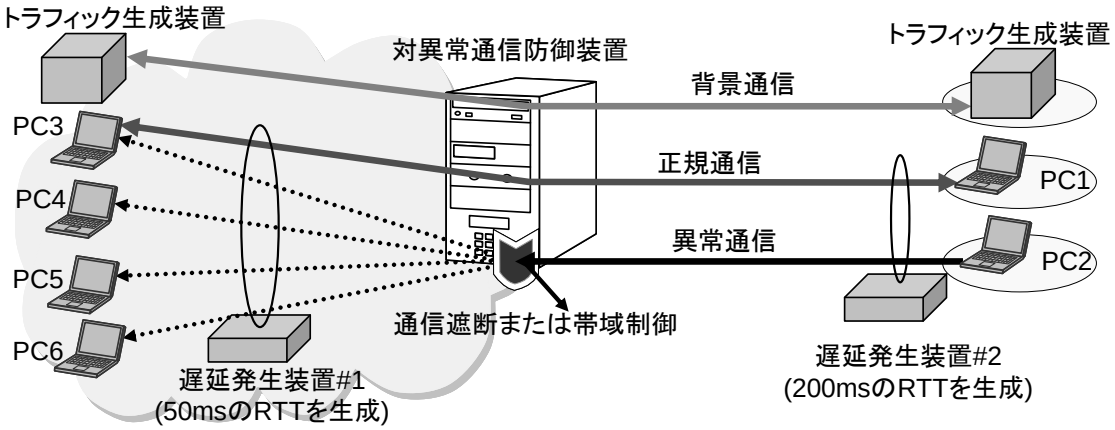


図 30　実験で使用した評価構成

表 6　パケット解析演算性能の比較

プロセッサ	プロセッサ数	パケット解析スピード
シングルコア汎用プロセッサ 1.5 GHz [127]	1	0.01Mpps
動的再構成 LSI 166 MHz [112][114][115]	2	3Mpps

けている時に、提案する対異常通信防御機能を有効にした場合の PC3 のスループットを図 31 ①に示した。対異常通信防御機能を有効にすることで、DDoS 攻撃のパケットが直ちに排除され、PC1 は PC3 への接続が可能となる。更に、PC2 が P2P アプリケーションを動作させて PC3-6 と P2P パケットをやり取りしている時に、提案する対異常通信防御機能を有効にした場合の PC2 のスループットを図 31 ②に示した。対異常通信防御機能は、P2P 通信のパケットが使用する帯域を、有効後直ちに 1Mbps に制限することを確認した。

続けて、異常通信(DDoS 攻撃)のスループットを 512pps として、背景通信のスループットを 0 から 6Mpps にした際の、対異常通信防御時間を評価した。異常通信の帯域占有率と対異常通信防御時間(100 回の平均)の関係を図 32 に示す。従来のパケット標本解析を用いた対異常通信防御評価では、引例[116]に記載の対異常通信防御用のアルゴリズムを利用した。提案するパケット全数解析を用いた対異常通信防御の評価では、異常通信探知用の閾値を 100ms あたり 5 パケットに設定した。一方、従来のパケット標本解析を用いた対異常通信防御の評価では、異常通信探知用の閾値を 20 秒あたり 1000 パケットとすることで、異常通信の探知に最低 10 個のサンプルパケットの受信が必要となるように設定した。

動的再構成 LSI を用いたパケット全数解析による対異常通信防御時間は、0.01 秒で一定であり、3.2 節で述べた目標時間(1 秒以内)を達成した。異常通信の最初の 5 つのパケットの解析が完了すると、異常通信の排除が始まるため、一定時間での対異常通信防御が実現できた。一方で、パケット標本解析における対異常通信防御時間は、統計誤差の影響により変動が大きく、異常通信の帯域占有率が減少するに従い 7 秒から 58 秒へと増加した。

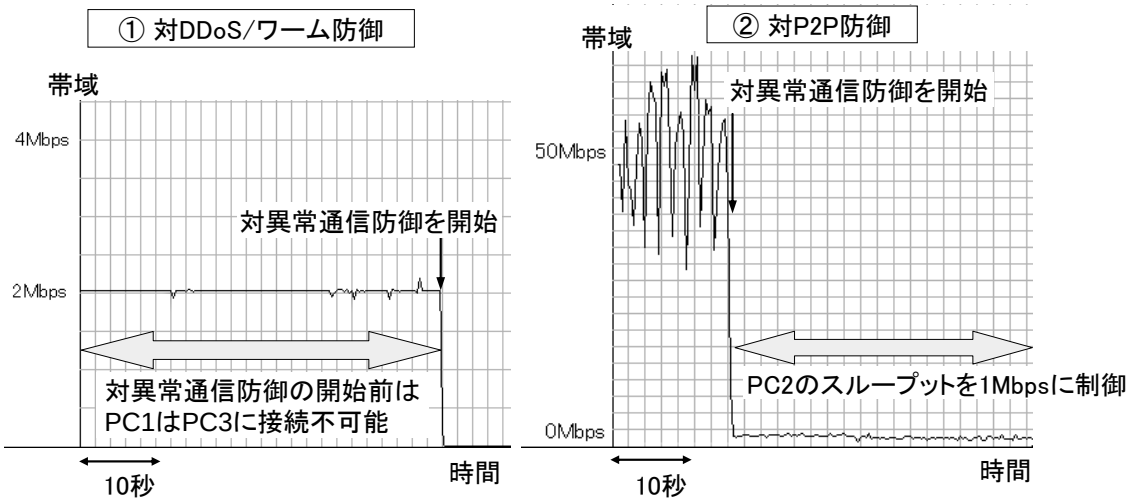


図 31　提案する対異常通信防御における異常通信のスループット

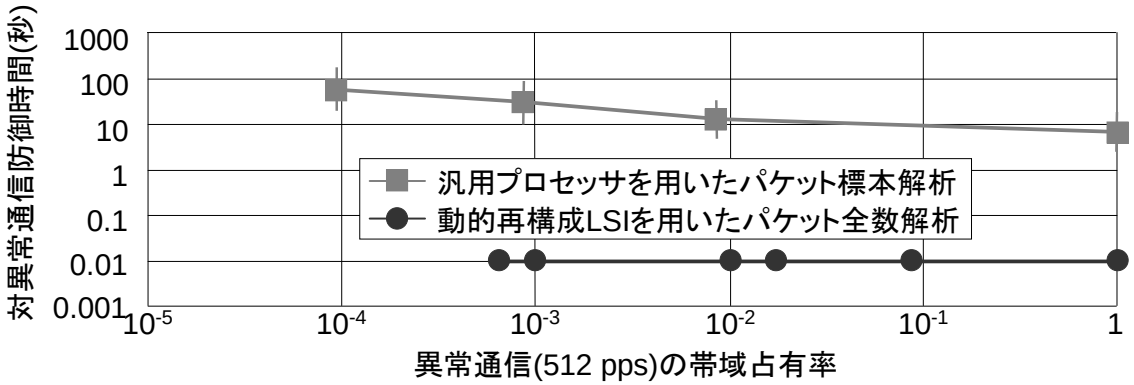


図 32　異常通信の帯域占有率と対異常通信防御時間の関係

以上により、動的再構成 LSI を用いたパケット全数解析は、汎用プロセッサを用いたパケット標本解析(サンプリングレート：1/2048)と比較して、異常通信の帯域占有率が 10^{-3} のケースで、対異常通信防御時間を平均 30 秒から 0.01 秒へと 3000 倍高速化することを確認した。

上記の通り、動的再構成 LSI を用いたパケット全数解析は、対異常通信防御時間を高速化する。その一方で、利用可能なメモリ量(304 KByte)が小さく、汎用プロセッサを用いたパケット標本解析にて利用可能なメモリ量(102 MByte)よりも小さいため、見逃し率(異常通信を正規通信と判定する確率)を増加させる可能性がある。実環境での利用上問題ないことを確認するため、見逃し率の評価を行った。異常通信端末の帯域占有率と見逃し率の関係を図 33 に示す。一般ユーザのインターネット回線における実効速度が 1～数十 Mbps であることを考慮して、本評価では、10Gbps 回線において 10Mbps を占める異常通信を探知することを想定して、帯域占有率が 10^{-3} の異常通信の見逃し率を 1%以内に抑えることを目標

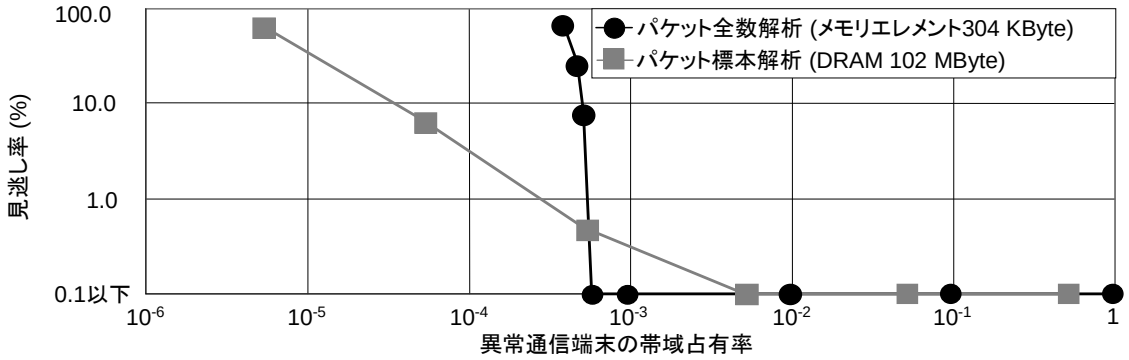


図 33　異常通信端末の帯域占有率と見逃し率の関係

とした。

図 33 に示した評価結果から、異常通信の帯域占有率が 10^{-3} の時の見逃し率は、パケット全数解析を用いた時に 1%未満となり、パケット標本解析を用いた時と同様となった。そのため、パケット全数解析を用いた時の見逃し率は、実環境での利用上問題ないことを確認した。

なお、本研究では、見逃し率(異常通信を正規通信と判定する確率)を評価したが、別の研究[128]において、保護確率(異常と判定した端末の正規通信を正しく保護できる確率)の評価を実施している。保護確率は、セッションテーブルのサイズと RTT に依存する。

3.6　結言

本研究では、以下の 3 つの手法①～③を用いることで、様々な異常通信の高速探知と、高スループットを維持しつつ無瞬断でのアルゴリズム更新を可能とする対異常通信防御機能を提案した。

① 高速演算性能(83Mpps)を持つ動的再構成 LSI を用いたパケット全数解析方式
② 3 種類の異常通信(DdoS，ワーム，P2P)を少ない演算リソースで防止する一括防御方式
③ パケット廃棄やスループット劣化の無いアルゴリズム無瞬断更新方式

提案するパケット全数解析方式と一括防御方式とアルゴリズム無瞬断更新方式を用いて、3 種類の異常通信(DDoS、ワーム、P2P)を探知して制御する対異常通信防御装置を設計してシミュレーションを行った。その結果、以下を実現する見通しを得た。

① 83Mpps のスループットにおける 1 秒以内の対異常通信防御
② 従来の特化防御方式と比較して回路量を 24%削減
③ 帯域利用率 100%でも 1 時間に 24M 回のアップデート可能

具体的には、83Mpps のスループットにおける 1 秒以内の対異常通信防御を、従来方式よ

り少ない 432 個の実行エレメント(動的再構成 LSI の 3 個分に相当)を使用することで実現する見通しを得た。更に、10Gbps × 4 回線からパケットが対異常通信防御装置に到着する環境において、アルゴリズムを無瞬断で更新可能な時間帯は、帯域利用率が増加するに従い減少するが、帯域利用率が 100%のときでも、全時間帯の 1500 分の 1(1 時間に 24M 回のアップデートが可能)を占めることを確認した。設計した対異常通信防御部をデータが通過する時間は 852 ns (142 クロックサイクル相当)、アルゴリズムの変更を指示してから実際に更新が完了するまでに必要な時間は 102 ns (17 クロックサイクル相当)であり、データ通過時間とアルゴリズム更新時間を合わせた時間が 954ns となる。一方、フルサイズ(MAC アドレスを含むパケット長が 1514 バイト)のパケットの到着間隔は 1230 ns なので、4 つの回線でフルサイズのパケットが同じ時間帯に到着すれば、アルゴリズムを無瞬断で更新可能である。

シミュレーション評価の次に、提案する対異常通信防御機能を、2 つの動的再構成 LSI に搭載した試作機を用いて実験評価を行った。その結果、以下を実現する見通しを得た。

① 対異常通信防御機能を有効にした後、直ちに異常通信を検出して制御
② 標本解析による従来方式よりも 3000 倍高速(0.01 秒)に異常通信を防止
③ 異常通信の帯域占有率が 10^{-3} の時の見逃し率は、従来の標本解析と同等

具体的には、DDoS 攻撃や P2P トラフィックが流れる環境において、DDoS 攻撃の排除と正規通信の保護、および、P2P トラフィックの帯域制限を実現することを確認した。更に、異常通信(DDoS 攻撃)のスループットを 512pps として、背景通信のスループットを 0 から 6Mpps に増加させていった際の、対異常通信防御時間を評価したところ、動的再構成 LSI を用いたパケット全数解析による提案方式が、汎用プロセッサを用いたパケット標本解析による従来方式よりも 3000 倍高速(0.01 秒)に異常通信を防止することを確認した。更に、提案方式の利用可能なメモリ量(304 KByte)が小さいことにより、見逃し率(異常通信を正規通信と判定する確率)が増加する可能性を評価した。その結果、異常通信の帯域占有率が 10^{-3} の時の見逃し率は、パケット全数解析を用いた時に 1%未満となり、パケット標本解析を用いた時と同様となった。そのため、パケット全数解析を用いた時の見逃し率は、実環境での利用上問題ないことを確認した。

以上により、ネットワークアプライアンスの様々な機能をユーザ要求に応じて無瞬断再構成する提案方式を用いることで、高スループットをサポートしつつ、日々進化する多様な攻撃手法に対応できることを示した。

第４章　多様な端末に対応する汎用性の高い通信高速化

4.1　緒言

情報システムの複雑化や大規模化に伴う ICT 機器の所有・管理コストの増加を背景に、企業や個人向けの情報システムの一部として、ネットワーク越しに ICT サービスを安価に提供するクラウドの利用が増えつつある。

クラウドの利用ユーザは、多様な端末(PC 等の固定端末，スマートフォン等のモバイル端末)の多様なアプリケーション(HTTP，HTTPS，CIFS，FTP，等)を用いて、多様なネットワーク(WAN(Wide Area Network)，インターネット，無線網，等)経由でクラウドのデータにアクセスする。特に近年は、無線網を用いた遠隔拠点からのアクセスの形態が増えている。データアクセス時の通信品質を、ユーザ・拠点・端末毎にデータを管理していた時と同様に保つため、ユーザが利用する端末・アプリケーション・ネットワークに依存せず、ローカル環境と同等の速度でのデータ転送を実現することが重要となる。

ところが、クラウドのデータにアクセスする時に広く利用される TCP 通信は、ネットワークの通信遅延やパケット廃棄の影響により、スループットが低下しやすく、クラウドサービス利用時の通信品質が劣化するという問題があった。ユーザが遠隔地から WAN やインターネットなどを経由して通信する場合や、無線網を経由して通信する場合は、データを送信してから確認応答を受け取るまでの往復通信遅延時間 RTT (Round Trip Time)が大きくなりやすい。加えて、WAN やインターネットでは、回線帯域が狭くなるボトルネックや他のトラフィックとの競合により、ルータやスイッチのキューが溢れてパケット廃棄が発生しやすい。無線網では、ビットエラーによるパケット廃棄が発生しやすい(図 34)。

そこで、本章では、RTT とパケット廃棄率(PLR: Packet Loss Ratio)の大きいネットワークにおいて TCP 通信のスループットが低下する問題を解決するため、クラウド利用ユーザの既存の端末やアプリケーションに変更を加えることなく、クラウドのデータにアクセスする際の通信品質を改善する TCP 高速化技術を検討する。

まず、プライベートクラウドを利用する際の、データセンタと拠点との間の TCP 通信を高速化するために、ラウンドトリップタイム(RTT)とパケット廃棄(Discard/Drop)に非依存(Independent)な輻輳制御(Congestion Control)を行う TCP 高速化技術 RADIC-TCP [129](RTT-And-Discard-Independent Congestion Control TCP)を提案する。RADIC-TCP は、パケット廃棄の影響を緩和するために、PLR の変化率に基づいてネットワークの輻輳を検出する輻輳制御アルゴリズムを用いる。PLR の変化率が大きい時のみ輻輳発生と判断することで、一時的なパケット廃棄に過敏に反応して帯域が減少するのを防ぐ。輻輳発生時は、輻輳の原因となった RTT 前の送信帯域と、輻輳発生時の廃棄帯域に基づいて送信帯域を減少させることで、残存帯域(他のトラフィックの使い残した帯域)への追従を可能とする。また、トークンバケツアルゴリズムを用いて、RTT に関わらず一定時間毎にトークンサイズに応じた数のパ

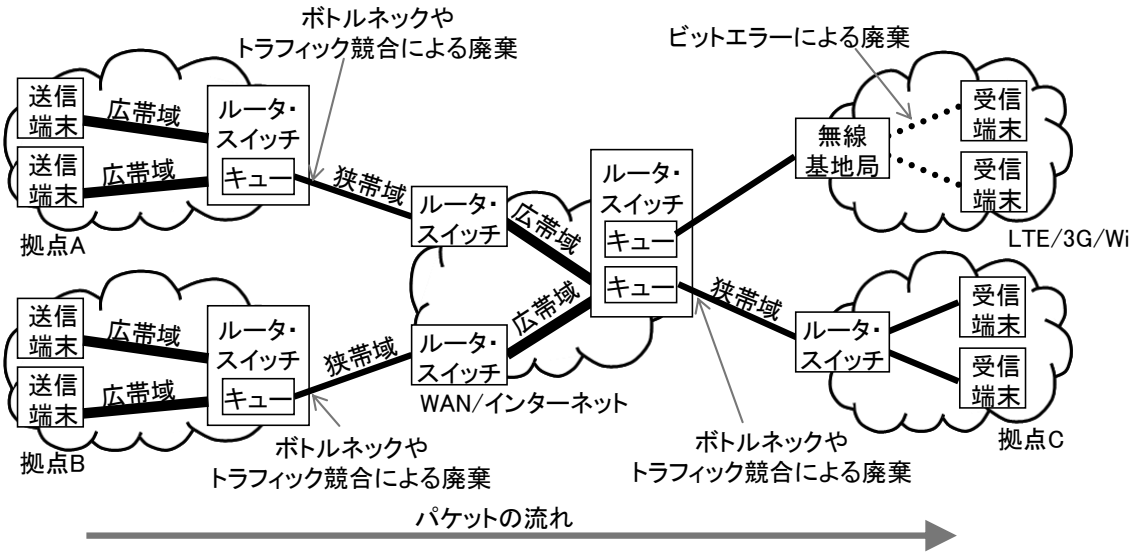


図 34　ネットワークにおけるパケット廃棄発生ポイント

ケットを一定間隔で送信することで、パケット廃棄の発生を抑制しつつ、RTT の影響を緩和する。更に、PLR が高い時のパケット再送の演算コストを削減するために、廃棄箇所の特定が容易な NACK (Negative Acknowledge)による再送制御方式を用いる。

次に、パブリッククラウドを利用する際の、クラウド利用ユーザの端末向けの TCP 通信を高速化するために、RADIC-TCP と SACK (Selective ACKnowledgement)[130]を併用することで、データセンタ側に設置するだけで、モバイル端末をはじめとする多様な端末向けの通信を高速化可能な汎用的な TCP 高速化技術である非対向設置による SACK 併用 RADIC-TCP を提案する。本 TCP 高速化技術は、RADIC-TCP と同じ輻輳制御アルゴリズムを用いつつ、TCP Reno [40]同様の標準的な SACK を用いて再送制御を行う。加えて、遠隔拠点への装置設置や、対向端末へのソフトウェアのインストールを必要としない。そのため、クラウド利用ユーザの多様な端末向けの TCP 通信を高速化することが可能である。特に、モバイル端末の無線回線区間の TCP 通信高速化など、クラウド利用ユーザの端末側に装置を設置することが出来ない場合に有用である。クラウド事業者は、クラウド利用ユーザ向けに本 TCP 通信高速化機能を付加価値サービスとして提供することが可能となる。

本章では、上記の提案する RADIC-TCP を用いて TCP 通信高速化を行うネットワークアプライアンスの試作機を用いて、様々なネットワーク環境やアプリケーションの利用を想定した実験評価を行い、高速化効果を測定した。更に、上記の提案する非対向設置による SACK 併用 RADIC-TCP を用いて、多様な端末向けの汎用的な高速化が実現可能であることを実証するため、PC 端末だけでなく、モバイル端末も用いて評価を行った。モバイル端末向けの通信では、無線通信区間の高速化が重要となる[131]。加えて、無線通信区間では、ハンドオ

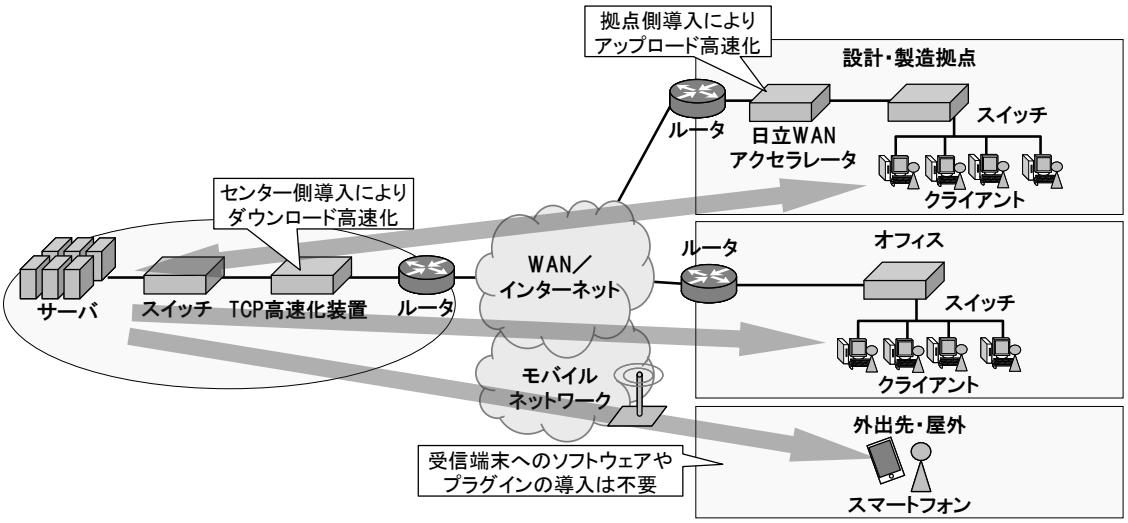


図 35　TCP 高速化装置の利用シーン

ーバの影響によりパケット廃棄の発生間隔に依存して意図しない動作をする可能性がある[132]。そこで、本章では、実際の無線ネットワーク環境を用いて性能評価を行った。

以降、4.2 節では TCP 通信高速化技術の概要を説明する。利用シーンや利用方法の他、アーキテクチャについて説明する。4.3 節では、従来 TCP の詳細と問題点について述べる。4.4 節では、提案 TCP の詳細と効果について述べる。4.5 節では、データセンタ側に設置するだけでよい汎用的な提案 TCP の詳細について述べる。4.6 節では、実験評価と実網評価について述べる。4.7 節では、提案する TCP 高速化技術についてまとめる。

4.2　TCP 通信高速化の概要

4.2.1　利用シーンや利用方法

本研究が対象とする TCP 通信高速化を行うネットワークアプライアンスの利用方法は、転送データの更新頻度や、データを転送する方向により異なる。

データセンタの CAD データを、海外等の遠隔地の設計・製造拠点から参照して編集するユースケースでは、データセンタから CAD データをダウンロードすると共に、拠点から更新済みの CAD データをアップロードする。データ転送量の多いダウンロードとアップロードの双方向を高速化するため、図 35 に示すように、TCP 通信高速化を行うネットワークアプライアンスはデータセンタ側と拠点側の両方に設置して利用する。

一方、データセンタの端末のディスプレイ画面を、海外等の遠隔地のオフィスからシンクライアントなどを用いて参照するユースケースや、データセンタの業務データを、外出先・屋外からスマートフォンなどを用いてシェア・閲覧するケースでは、データを取得するダウンロード方向のデータ転送量が多く、キーボードやマウス操作などを行うアップロード方向のデータ転送量は少ない。データ転送量の多いダウンロード方向を高速化するた

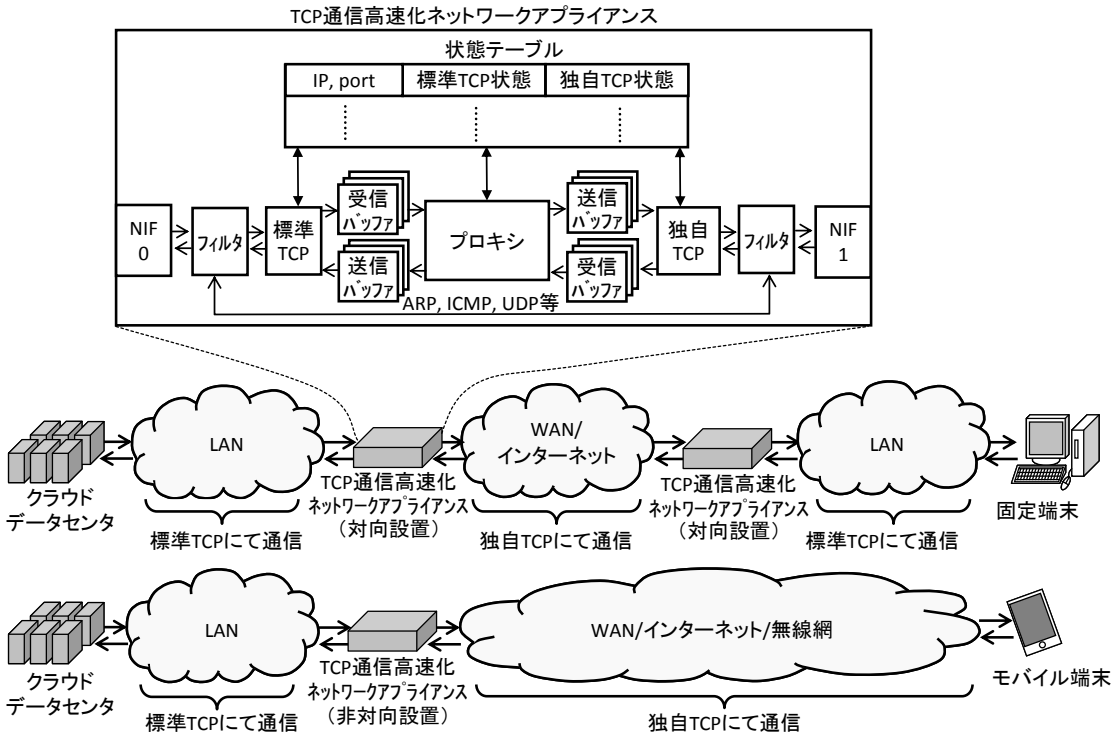


図 36　TCP 通信高速化を行うネットワークアプライアンスの構成と利用形態

め、図 35 に示すように、TCP 通信高速化を行うネットワークアプライアンスはデータセンタ側のみに設置して利用する。この際、ユーザ端末側へのソフトウェアやプラグインの導入は不要である。

4.2.2　アーキテクチャ

TCP 通信高速化を行うネットワークアプライアンスは、ユーザの利用する端末が特別な変更を加えることなく TCP 通信高速化の機能を使えるように、ネットワークアプライアンスと端末の間の標準 TCP を用いた通信を終端し、独自 TCP へと変換した上で通信を行う。

図 36には、TCP通信高速化を行うネットワークアプライアンスの構成と、クラウドデータセンタ側と端末側の両方に設置して利用する対向設置形態、および、クラウドデータセンタ側のみに設置して利用する非対向設置形態を示す。

TCP通信高速化を行うネットワークアプライアンスは、標準TCP、独自TCP、送受信バッファ、1対のIPアドレスとポート番号の組合せ毎にTCP通信の状態を記録する状態テーブル、ネットワークインターフェース(NIF)、非TCP(ARP，ICMP，UDP，等)パケットをパススルーさせるフィルタ、受信バッファの整列済みデータを送信バッファへ載せ換えるプロキシから成る。

TCP通信高速化ネットワークアプライアンスは、NIF0側のLANネットワークと接続する端末との間で標準TCPを用いて通信する一方、NIF1側のWANネットワークでは独自TCPを

用いて通信する。対向設置形態の場合は、対向のネットワークアプライアンスで独自TCPを標準TCPに変換してから対向端末と通信する一方、非対向設置形態の場合は、独自TCPを用いて対向端末と直接通信を行う。

4.3 従来の TCP

TCP は、輻輳制御，送信量制御，再送制御の 3 つの制御から成る。輻輳制御は、ネットワークの輻輳を監視して、ネットワークの輻輳を検出したときに送信帯域を制限するための閾値である輻輳ウィンドウサイズを減少させる一方、ネットワークの輻輳を検出していないときに輻輳ウィンドウサイズを増加させる機能である。送信量制御は、予め定められた時間間隔で TCP が送信するデータ量を制限する機能である。再送制御は、ネットワーク等で廃棄したパケットを再送する機能である。これらの 3 つの制御について、以下 4.3.1～4.3.3 節で詳細に述べる。

4.3.1 パケット廃棄発生やパケット廃棄率や通信遅延増減に基づく輻輳制御

従来の TCP 通信は、ネットワークで輻輳が発生しているか否かを判断するために、パケット廃棄の発生，パケット廃棄率や往復通信遅延増減に基づいて輻輳を検出する。パケット廃棄の発生に基づく輻輳検出を用いる TCP として、TCP Reno [40]，TCP New Reno [41]，High-Speed TCP [44]，Scalable TCP [133]，BIC TCP [134]，CUBIC TCP [42]がある。パケット廃棄率に基づく輻輳検出を用いる通信方式として、SABUL [135]がある。往復通信遅延増減に基づく輻輳検出を用いる TCP として、TCP Vegas [136]，Fast TCP [45]がある。パケット廃棄の発生と往復通信遅延増減の両方を用いた輻輳検出方式として、Compound TCP[43]がある。

ネットワークの輻輳を検出すると、送信端末は輻輳ウィンドウサイズを定率減少[40][41][44][133][134][42][43]、線形減少[136]、RTT に基づく減少[45]のいずれかの方式で減少させることで送信帯域を減少させる。例えば、TCP Reno は輻輳ウィンドウサイズを半分に減少させるし、帯域の急激な減少を防ぐために、半分以下の割合で輻輳ウィンドウサイズを減少させるアルゴリズムもある。一方、ネットワークの輻輳を検出していないときは、送信端末は輻輳ウィンドウサイズを線形増加[40][41][44][136]，指数的増加[133]，RTT に基づく増加[45][43]，キュービック関数に基づく増加[42]，バイナリ探索に基づく増加[134]のいずれかの方式で増加させることで送信帯域を増加させる。例えば、CUBIC TCP や BIC TCP は、輻輳を検出したときの輻輳ウィンドウサイズを記録しておき、輻輳解消後に、記録しておいた輻輳ウィンドウサイズまで急速に増加させる。

(1)パケット廃棄発生に基づく輻輳制御

IETF 標準の TCP Reno[40]を始め、High-Speed TCP[44], Scalable TCP[133], BIC TCP [134], CUBIC TCP[42] , TCP Hybla[137], TCP Westwood[138], TCP Star[139]などは、パケット廃棄発

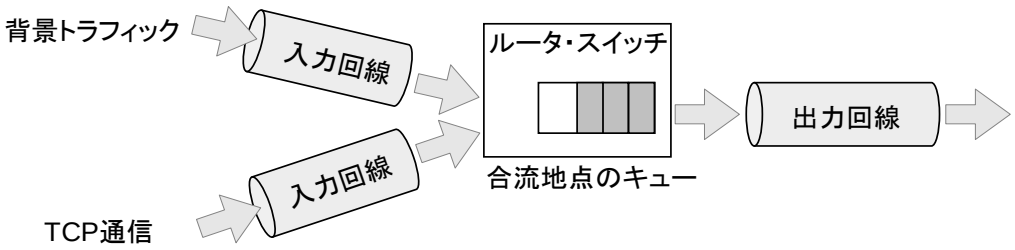


図 37　TCP 通信と背景トラフィックの合流地点における競合

生に基づき輻輳を検出する。Compound TCP[43]は、往復通信遅延増加も併用して輻輳を検出する。上記 TCP は、パケット廃棄が発生するとネットワークに輻輳が発生したと判断して、送信帯域を定率減少[40][44][133][134][42][43][137]、または ACK 受信間隔に基づく推定帯域に減少させる[138][139]。

パケット廃棄発生に基づく輻輳制御は、パケット廃棄が発生すると一時的か継続的かを区別することなく一律に輻輳が発生したと判断して、輻輳ウィンドウサイズを減少させることで送信帯域を減少させる。パケット廃棄の発生が一時的で短期間(1ms 以内)に輻輳が解消する場合でも送信帯域が減少する上に、パケットの再送が完了して送信帯域が増加に転じるまで最短でも RTT の時間がかかるので、一時的なパケット廃棄が発生する RTT の大きな通信回線で帯域利用効率が減少する課題があった。以下に詳細を述べる。

一時的なパケット廃棄は、TCP 通信と背景トラフィックの合流地点(図 37)における短期的(1ms 以内)な競合や、無線網のビットエラーにより発生する。一時的なパケット廃棄の発生例の一つとして、図 38 に、1ms 分のキュー長を持つ合流地点において競合する TCP Reno 通信(RTT 50ms)と背景トラフィックのスループットの時間的推移を示す。背景トラフィックは、パケット発生頻度がポアソン分布に従う例[140]と、平均帯域が 20Mbps となる例[141]を模擬して、1ms あたり平均 2 個のパケットをランダムに発生させた。

背景トラフィックの帯域利用率は 1ms 間隔で見ると 0～100%で変動しており、ネットワークの合流地点で TCP Reno 通信と競合すると、帯域利用率が 100%を超えて、一時的にパケット廃棄が発生する。TCP Reno 通信は、一時的なパケット廃棄が発生すると 50%の割合でウィンドウサイズを減少させることで送信帯域を定率減少させるが、輻輳が解消した後は送信帯域の下げ過ぎとなり、回線帯域を使い切れなくなる。図 38 の例では、平均スループットが 13Mbps に低下し、背景トラフィックの使い残した残存帯域 80Mbps を使い切れない。

図 38 の例のように、パケット廃棄発生に基づく輻輳検出は、ネットワークの回線帯域の長期的な利用率が 100%に到達していなくても、背景トラフィックとの競合により発生する一時的なパケット廃棄に敏感に反応して、輻輳発生と判断して送信帯域を下げ過ぎてしまい、送信帯域が回線帯域よりも小さい領域で安定し、スループットが劣化する課題があった(図 39)。

一時的なパケット廃棄が発生した時の送信帯域の過剰な減少を防ぐために、帯域減少率

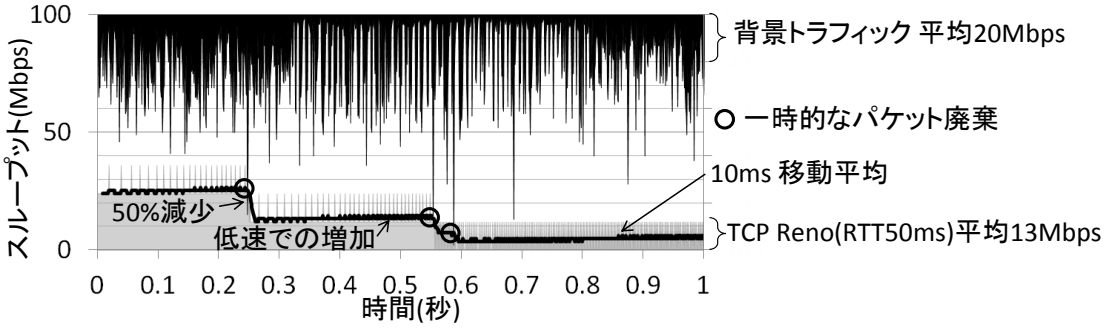


図 38　背景トラフィックと標準 TCP が競合した時のスループット推移

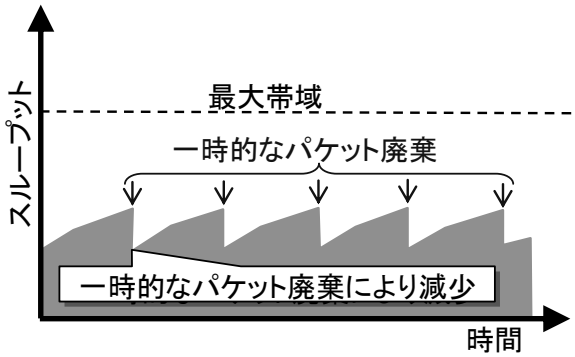


図 39　パケット廃棄発生時に一定割合でスループット減少させる輻輳制御アルゴリズム

を 50%未満とする方式[44][133][134][42]や、パケットペア[142]や ACK クロック[40]を応用して、帯域減少幅を ACK 受信間隔に基づき必要最小限に最適化する方式[138][139]もある。

しかし、パケット廃棄が発生した時の送信帯域の過剰な減少を防ぐだけでは、本質的な課題の解決とはならない。これは、パケット廃棄発生に基づき輻輳を検出する TCP は、パケット廃棄発生後に送信帯域を減少させ、廃棄パケットの再送が完了して送信帯域が増加に転じるまで最短でも RTT の時間がかかるので、RTT に 1 回のパケット廃棄が起こる帯域に達すると、それ以上は送信帯域が増加していかないからである。

パケット廃棄発生に基づき輻輳を検出する TCP の送信帯域を B，最大セグメントサイズを MSS (Maximum Segment Size)，パケット廃棄率を p とすると、RTT あたりのパケット廃棄数 B*RTT/(MSS*8)*p が 1 以下となる条件から、送信帯域 B の上限を表す式 4-1 が導かれる。送信帯域 B の上限はパケット廃棄率 p の増加に反比例して減少する。

$$B*RTT/(8*MSS)*p \leqq 1 \Rightarrow B \leqq MSS*8/(RTT*p) \quad (4\text{-}1)$$

例えば、TCP Star[139]は、パケット廃棄発生時の送信帯域の過剰減少を防ぐために、帯域減少幅を ACK 受信間隔に基づき必要最小限に最適化しているが、式 4-1 の上限を超えるスループットを得られない。文献[139]記載の RTT 250ms, MSS 1448byte, BER (Bit Error Rate) 10^{-6}, 10^{-5} のシミュレーション環境における TCP Star のスループット 4.0, 0.4Mbps は、BER と p の換算式 $p=1-(1-BER)^{MSS*8}$ を用いて計算した式 4-1 の上限 4.0, 0.4Mbps と一致する。

以上のように、パケット廃棄発生に基づき輻輳を検出する TCP は、回線帯域が式 4-1 の上限よりも大きい場合に、回線帯域を使い切ることが出来ない。

(2)パケット廃棄率に基づく輻輳制御

パケット廃棄率に基づいて輻輳を検出する通信方式として SABUL[135]がある。SABUL は、予め定めた計測時間(10ms)毎にパケット廃棄率を測定する。パケット廃棄率が閾値を上回るときにネットワークに輻輳が発生したと判断して、送信帯域を定率(12.5%)で帯域減少させる。一方、パケット廃棄率が閾値(0.1%)以下となるときは、ネットワークに輻輳が発生していないと判断して、送信帯域を指数増加させる。

パケット廃棄率に基づく輻輳制御は、パケット廃棄率の計測間隔や輻輳検出用閾値の大小により、一時的なパケット廃棄を輻輳発生と判断して送信帯域を過剰に減少させるケースや、継続的なパケット廃棄を輻輳発生と判断せずに大量のパケット廃棄を発生させるケースがあり、様々なパケット廃棄発生パターンが存在するネットワークでの利用が困難となる課題があった。以下に詳細を述べる。

パケット廃棄率の計測間隔が小さいと、標本数の減少により 1 パケットでも廃棄するとパケット廃棄率が高まるので輻輳発生と判断されやすく、一時的なパケット廃棄による送信帯域の過剰減少が発生しやすい。一方、パケット廃棄率の計測間隔が大きいと、輻輳発生から輻輳検出までの時間が長くなり、その間に大量のパケット廃棄を発生させる。

また、パケット廃棄率の輻輳検出用閾値が小さいと、背景トラフィックとの競合で発生する一時的なパケット廃棄が、輻輳発生と判断されやすくなり、帯域過剰減少が発生する。一方、閾値が大きいと、他のトラフィックとの競合による継続的なパケット廃棄が起きても輻輳発生と判断されにくくなり、輻輳が継続することで大量のパケット廃棄を発生させる。

(3)通信遅延に基づく輻輳制御

通信遅延増加に基づいて輻輳を検出する TCP として、TCP Vegas [136], Fast TCP [45]がある。これらの TCP は、往復通信遅延 RTT が増加したときに輻輳発生と判断して、送信帯域を減少させる。TCP Vegas は、RTT が予め定めた閾値を超えた時に、RTT 毎に MSS*8/RTT の割合で送信帯域を線形減少させる。Fast TCP は、送信帯域 b (式 4-2)を、2RTT 毎に RTT に反比例するように制御する。α は定数、RTT_{min} は最小 RTT である。

$$b=b*RTT_{min}/RTT+\alpha \qquad (4\text{-}2)$$

通信遅延は、通信回線を信号が進む伝播遅延と、ルータやスイッチにおけるキュー待ち時間から成る。ルータやスイッチの出力回線帯域の利用率が 100%を超過して、キューが溢れてパケット廃棄が始まると、パケット廃棄の多寡に関わらずキュー長は一定となり(図 40)、通信遅延も一定となる(図 41)。通信遅延とパケット廃棄の大きさに相関関係が存在し

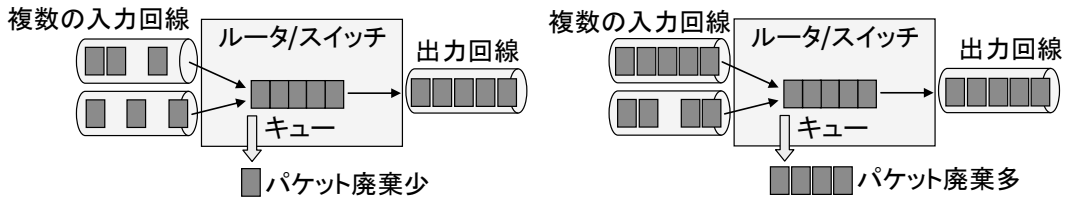


図 40　輻輳規模に関わらず通信遅延（＝キュー待ち時間＋伝播遅延）は一定

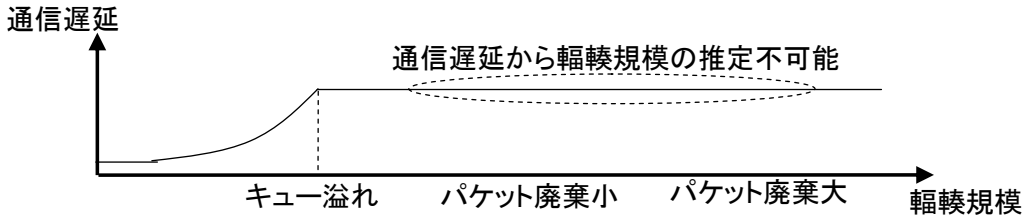


図 41　輻輳規模と通信遅延の関係

ないため、通信遅延に基づく輻輳制御は、通信遅延からの輻輳規模の推定が困難となる課題があった。以下に詳細を述べる。

往復通信遅延 RTT は、ルータやスイッチの出力回線帯域の利用率が一時的に 100%を超過して、キューにパケットが蓄積していくに従い増加する。RTT が増加してくると、通信遅延に基づく輻輳制御は、パケット廃棄が発生していなくても輻輳発生と判断して送信帯域を減少させるため、廃棄発生に基づく輻輳制御よりも送信帯域を減少させやすい。特に、キュー長が長い場合は、パケット廃棄が発生する前に、RTT が増加してしまうので、通信遅延に基づく輻輳制御は、他のトラフィックとの競合による一時的な遅延増加でも輻輳発生と判断して、送信帯域の過剰減少を発生させる。

一方、往復通信遅延 RTT は、ルータやスイッチの出力回線帯域の利用率が継続的に 100%を超過して、キューが溢れてパケット廃棄が始まると増加を止める。RTT が小さい状態で増加を止めると、通信遅延に基づく輻輳制御は、大量のパケット廃棄が発生していても輻輳発生と判断しないため、過剰送信による大量のパケット廃棄を発生させやすい。特に、キュー長が短い場合は、RTT が十分大きくなる前に、パケット廃棄の発生が始まるので、通信遅延に基づく輻輳制御は、他のトラフィックとの競合による継続的なパケット廃棄でも輻輳発生と判断できず、過剰送信による大量のパケット廃棄を発生させる。

以上のように、通信遅延に基づく輻輳制御は、キュー長が長いときに他のトラフィックと一時的に競合する場合や、キュー長が短いときに他のトラフィックと継続的に競合する場合など、様々な状況が発生し得る WAN やインターネットでの利用が困難となる課題があった。

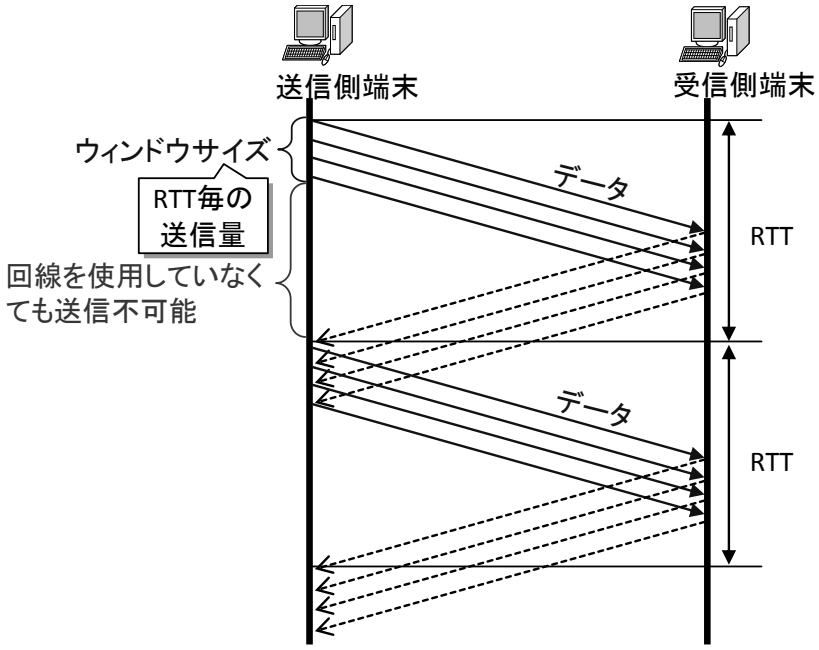


図 42　ウィンドウサイズを用いた送信量制御

4.3.2　ウィンドウサイズを用いた送信量制御

従来の TCP は、確認応答(Acknowledgement: ACK)を受信せずに送信可能なデータ量をウィンドウサイズに基づいて制限している。送信端末は送信バッファサイズを最大とする輻輳ウィンドウ(CWND)、受信端末は受信バッファサイズを最大とする受信ウィンドウ(RWND)と呼ばれる閾値を管理しており[121][134]、ACK を受けることなく送信可能なデータ量を制限している。送信端末は、CWND と RWIN のいずれか小さい方のサイズのデータを、回線帯域を最大値としてバースト送信する。ACK が戻り新たなデータが送信可能になると、CWND と RWIN のいずれか小さい方のサイズのデータを、再び回線帯域でバースト送信する。送信端末は、1RTT あたりにウィンドウサイズ(CWND, RWIN)を超えてデータを送信することができず、送信帯域は、ウィンドウサイズを RTT で除算することで得た値が上限となる(図 42)。

ウィンドウサイズが小さいと、送信端末は、RTT が増加したときに、通信回線が未使用でも、長時間パケットを送信することができない。更に、パケット廃棄が発生すると、廃棄したパケットを再送して ACK パケットを受け取るまで、1RTT を超える期間の間、新たなデータを送信することができなくなる。そのため、RTT が大きくなると、スループットが減少しやすい。

ウィンドウサイズの最大値は、送信側と受信側の端末のバッファサイズのいずれか小さいほうに応じて定まる。そのため、送信側と受信側の両方の端末のバッファサイズを大きい値に設定すれば、送信側端末は大きなウィンドウサイズを使用することが可能となる。

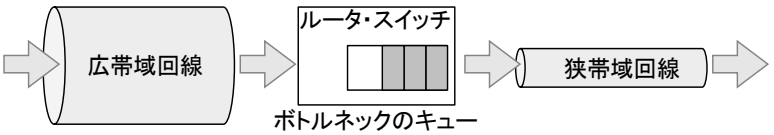


図 43　ボトルネック

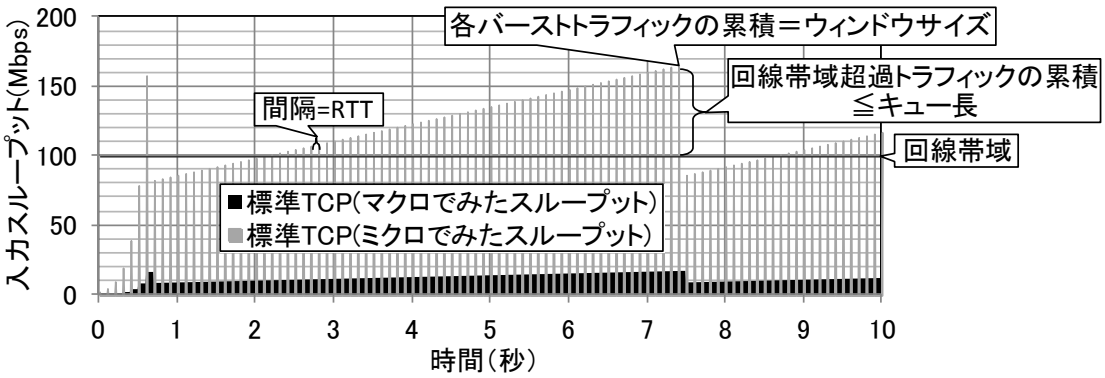


図 44　マクロとミクロで見たスループット

しかしながら、過度に大きなウィンドウサイズは、送信側端末からのバースト送信を原因とするパケット廃棄を発生させやすく、スループットを低下させやすい。バースト送信を抑制するツール(HTB [143], PSPacer [144])を用いた送信帯域の平滑化は、残存帯域が既知のときのみ使用可能であり、残存帯域が未知である WAN やインターネットでの利用は困難である。

従来 TCP のスループットをミクロで見ると、ウィンドウサイズを用いた送信量制御の影響によりバーストトラフィックの繰り返しとなる。バーストトラフィックは、トラフィックの合流地点(図 34)で、4.3.1 節の図 38 に示したバックグラウンドトラフィックとの競合による一時的な廃棄が発生しやすく、スループットが低下しやすい。図 43 に示すような回線帯域が狭くなるボトルネックでも、キューが溢れてパケット廃棄が発生しやすく、スループットが低下しやすい。

図 44 には、回線帯域が 1Gbps から 100Mbps に小さくなるボトルネックを持つ RTT 100ms のネットワークにおける TCP Reno [40]のスループットを、マクロ(100ms 平均)とミクロ(10ms 平均)で示す。

マクロにみると平滑なトラフィックでも、ミクロにみると RTT 周期のバーストトラフィックとなる。各バーストの累積がウィンドウサイズに相当しており、ボトルネックの出力回線帯域 100Mbps を超過したトラフィックの累積がボトルネックのキュー長を上回ると、パケット廃棄が発生し、スループットが減少する。

以上のように、従来の TCP は、自ら発生させたバーストトラフィックにより、合流地点やボトルネックで廃棄を発生させやすく、送信帯域を減少させる場合がある。

4.3.3 SACK を用いた再送制御

従来の TCP は、パケット廃棄が発生すると、SACK (Selective Acknoledgement) [130][145]を用いて廃棄データの再送を行う。パケット廃棄が発生していない時に、受信端末が送信端末からパケットを受信すると、受信済データの末尾位置を表すシーケンス番号を記載した ACK パケットを返信する。パケット廃棄が発生した時に、受信端末が送信端末から廃棄パケットの後続パケットを受信すると、前回返信した ACK パケットと同じシーケンス番号を持つ ACK パケット(重複 ACK)を返信する。SACK が有効な時は、後続パケットのデータの先頭と末尾を表すシーケンス番号を、SACK オプションフィールドに記載した重複 ACK を継続的に送信端末に送信する。送信端末は、重複 ACK が 3 回到着すると、パケット廃棄が発生したと判断して、重複 ACK の受信シーケンス番号が通知する位置からのデータを含むパケットを再送する。但し、SACK オプションフィールドに記載の、受信端末が断続的に受信した後続パケットのデータは再送しない。受信端末からの重複 ACK 送信は、廃棄したパケットのデータを受信するまで継続する。複数のパケット廃棄が断続的に発生すると、一番新しく受信した後続パケットのデータを先頭にして、最大 4 つのデータの先頭と末尾のシーケンス番号を SACK オプションフィールドに記載する。タイムスタンプオプション[146]が有効な場合は、最大 3 つのデータのシーケンス番号を記載する。

受信端末は、不連続な受信済データを予め定められた最大数まで記録する。更に、先頭と末尾の位置を SACK オプションフィールドに記載することで送信端末に通知する。送信端末も、不連続な確認応答済データの位置を、予め定められた最大数まで記録しておく。図 45 には、受信端末が、断続的に受信したデータの位置を最大 3 箇所まで記録しておくことが出来る場合の例を示す。

受信端末は、ネットワークで断続的に発生した 5 つのパケット廃棄(図 45 の B, D, F, H, J)により、5 つの不連続なデータ(図 45 の C, E, G, I, K-L)を受け取ると、最初の 3 つの不連続なデータ(図 45 の C, E, G)を受信バッファに蓄積する。残りの 2 つの不連続なデータ(図 45 の I, K-L)は、記録箇所が不足しているので廃棄する。受信バッファに蓄積した不連続なデータの先頭と末尾のシーケンス番号は、SACK を用いて送信端末に通知する。送信端末は、SACK を受信すると、SACK が記載する不連続な受信済データの先頭と末尾のシーケンス番号に基づいて廃棄したパケットのデータ(図 45 の B, D, F, H)を推定して再送を行う。廃棄したパケットのデータ(図 45 の B, D, F, H)が再送により受信端末に到着すると、受信端末が記録していた不連続な受信データ(図 45 の C, E, G)の位置は消滅する。更に、受信端末は、末尾データの位置(図 45 の H)を持つ ACK パケットを送信端末に送信し続ける。

パケット廃棄が発生してから送信端末が再送した廃棄パケットが受信端末に到着するまで最低でも 1RTT の時間を必要とする。送信端末が、パケット廃棄率の高いネットワークに向けて広帯域でパケットを送信すると、大量のパケット廃棄が断続的に発生して、受信端末が受信バッファに蓄積する不連続な受信データの数が、予め定められた最大数を超える。受信端末は、最大数を超えた不連続な受信データを、受信バッファに蓄積せず廃棄する。

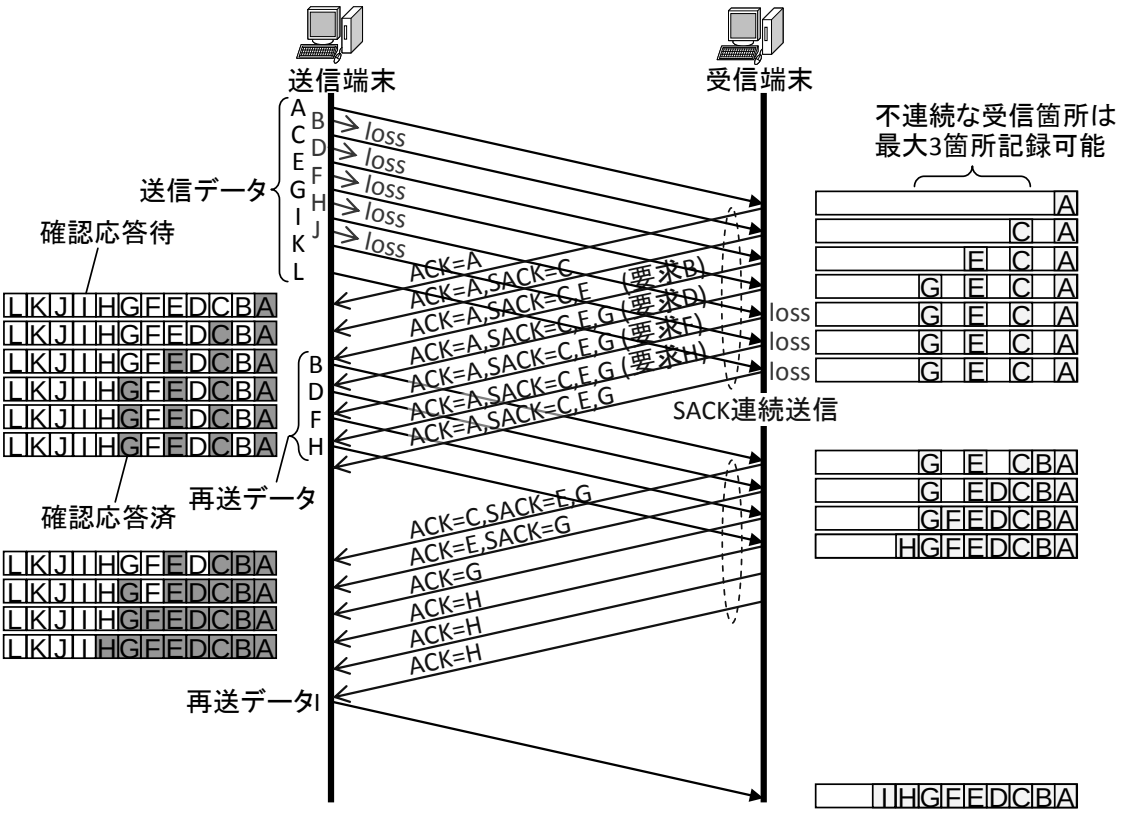


図 45　SACK 連続送信による再送

ネットワークにおけるパケット廃棄だけでなく、受信端末の受信バッファでもパケットが廃棄されるので、パケット転送が完了するまで、より多くの時間が必要となり、スループットが減少する。

以上に加えて、送信端末は、SACK 記載の不連続な受信済データの先頭と末尾のシーケンス番号から、廃棄したパケットのデータを推定して再送するために、不連続な確認応答済データの先頭と末尾のシーケンス番号を管理する。送信端末が SACK を受信する毎に、SACK 記載の不連続な受信済データの先頭と末尾のシーケンス番号と、送信端末が管理する不連続な確認応答済データの先頭と末尾のシーケンス番号の比較を行い、廃棄したパケットのデータを推定して再送を行う。

パケット廃棄率の高い通信回線では、受信端末の受信バッファに蓄積する不連続な受信データが常に存在する状態となり、送信端末は受信端末から高頻度で SACK を受信する。加えて、送信端末が管理すべき不連続な受信済データの数も増加するので、廃棄したパケットのデータを推定するためのシーケンス番号比較に必要な演算量が増加する。ある商用 OS では、受信端末の不連続な受信済データの最大蓄積数を小さく制限することで、シーケンス番号比較のための演算量が増加するのを抑止している。受信端末の不連続な受信済データの最大蓄積数を制限しない OS では、送信端末のシーケンス番号比較に必要な演算量が増加して、最大スループットが低下しやすい。

4.4 RADIC-TCP

本研究において提案する TCP 高速化技術 RADIC-TCP を利用して TCP 通信高速化を行うネットワークアプライアンスは、既存のアプリやネットワークがそのまま使えるようにしつつ、全てのアプリケーションがアクセスタイミングに関わらず WAN の回線帯域を効率的に利用できるようにすることを目標としており、下述①～⑧の 8 つの特徴を備える。

① 端末へのインストール不要で全ての端末の TCP 通信を高速化
② TCP 通信を用いるアプリケーションを一律に高速化
③ 更新頻度やアクセスタイミングに非依存な TCP 通信の高速化
④ ネットワークアプライアンス導入時に既存のネットワーク装置の設定変更不要
⑤ 既存のネットワーク装置の性能制約を受けない
⑥ 付加的なトラフィックでネットワーク負荷を与えない
⑦ 一時的なパケット廃棄の影響を受けず回線帯域を有効活用
⑧ パケット再送の演算量を削減しスループット向上

提案する TCP 高速化技術 RADIC-TCP はカーネルに入れて動作させることが理想だが、本研究ではネットワークアプライアンスに搭載してプロトコルコンバータとして動作させることを想定する。提案する RADIC-TCP を利用して TCP 通信高速化を行うネットワークアプライアンスは、2 つのルータやスイッチの間に挿入して、直列でつなぐインパス構成で設置することにより導入可能であり、端末へ新規ソフトウェアをインストールすることなく全ての端末の TCP 通信を高速化する(特徴①)。また、アプリケーションを限定することなく、TCP を用いるアプリケーションの通信であれば、一律に高速化が可能である(特徴②)。加えて、CAD データ、金融情報など更新頻度の高いデータを初回アクセスから高速化する(特徴③)。また、TCP パケットのフォーマットに準拠しないパケットを用いる SABUL[135]や XCP[147]などと異なり、TCP パケットに準拠したまま、IP アドレスやポート番号を変更することなく、TCP 通信を TCP 通信のまま高速化するので、ファイアウォール等におけるセッション毎のプロトコル番号や IP アドレスやポート番号の設定変更が不要であり、既存のネットワーク装置の設定変更が不要である(特徴④)。コネクション数を増加させて通信を高速化する GridFTP[148][149]と異なり、コネクション数を増加させずに高速化するため、ファイアウォールや NAPT のコネクション上限数など、既存のネットワーク装置の性能制約を受けない(特徴⑤)。更に、独自フォーマットの調査パケットや制御パケットを用いず、TCP パケットのフォーマットに準拠したまま、輻輳制御に使用する遅延や廃棄などを計算するため、付加的なトラフィックを発生させず、ネットワークに負荷を与えない(特徴⑥)。パケット廃棄率の変化率を用いて輻輳制御することで、一時的なパケット廃棄に過敏に反応して送信帯域が過剰に減少するのを防ぎつつ、輻輳発生時に検出した廃棄帯域に応じて送信帯域を減少させることが可能となり、回線帯域の利用効率を向上させることができる

(特徴⑦)。SACK ではなく NACK による再送を用いることで、パケット廃棄発生時の送信端末における廃棄したパケットを推定して再送するためのシーケンス番号比較を不要とし、パケット再送の演算量を削減しスループットを向上させることができる(特徴⑧)。

特徴①～⑤は、TCP 通信を維持したまま TCP 通信高速化を行うネットワークアプライアンスの特徴であり、特徴⑥～⑧は、提案する RADIC-TCP の特徴である。特徴⑥⑦は 4.4.1 節にて、特徴⑧は 4.4.3 節にて詳細に説明する。

4.4.1 パケット廃棄率の変化率に基づく輻輳制御

提案する RADIC-TCP は、パケット廃棄率の変化率に基づいてネットワークの輻輳を検出して輻輳制御を行う。送信端末は、SACK や NACK から推定した廃棄パケットのデータサイズ積算値を、パケット送信時刻と ACK 到着時刻の差から得た RTT で除算することで、廃棄帯域(超過帯域)を計算する。廃棄帯域の計算は RTT 間隔で行い、直近 RTT の廃棄帯域を 2RTT～RTT 前の送信帯域で除算することでパケット廃棄率を計算する。パケット廃棄率の変化率が大きい場合に、ネットワークに輻輳が発生したと判断する。更に、輻輳の原因となった RTT 前の送信帯域から、輻輳発生時の直近の廃棄帯域を減算した値を残存帯域と見なして、送信帯域を減少させる。一方、パケット廃棄率の変化率が小さい場合、ネットワークに輻輳が発生していないと判断して、送信帯域を増加させる。最初は線形に増加させ、帯域遅延積に比例した時間に輻輳が発生しなければ、その後は指数的に増加させる。

ネットワークで発生するパケット廃棄は、2 つのタイプに分類できる。1 つは、他のトラフィックとの継続的な競合や、利用帯域が回線最大帯域を超過することにより発生する帯域使い過ぎによる継続的なパケット廃棄である。もう 1 つは、他のトラフィックとの一時的な競合や、無線ノイズによるビットエラーなどにより発生する一時的なパケット廃棄である。

RADIC-TCP の利用帯域が、図 46 に示すように回線最大帯域を超過すると、パケット廃棄率が急激に増加して、パケット廃棄率の変化率が大きくなり、予め定めた閾値を超過する。そのため、RADIC-TCP は、使い過ぎによる継続的なパケット廃棄が発生したと考えて、送信帯域を減少させる。一方、RADIC-TCP の利用帯域が回線最大帯域よりも小さいときに、無線ノイズによるビットエラーなどにより一時的なパケット廃棄が発生したとしても、一時的なパケット廃棄はパケット廃棄率を急増させないため、送信帯域を減少させない。そのため、RADIC-TCP の利用帯域は、図 46 に示すように回線最大帯域付近で安定する。

一方、RADIC-TCP を用いた通信が、図 47 に示すように、帯域利用率の変動の大きい背景トラフィック(平均利用率 20%)と競合した場合、背景トラフィックと一時的に競合しても、一時的なパケット廃棄が発生するだけなのでパケット廃棄率は増加せず、送信帯域を増加させる。一方、背景トラフィックと継続的に競合すると、継続的なパケット廃棄が発生するのでパケット廃棄率が増加し、送信帯域を減少させる。そのため、RADIC-TCP の帯域利

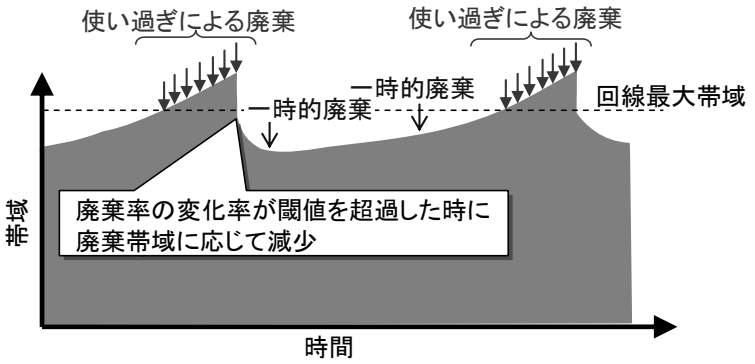


図 46　回線最大帯域を超過したときの RADIC-TCP の動作

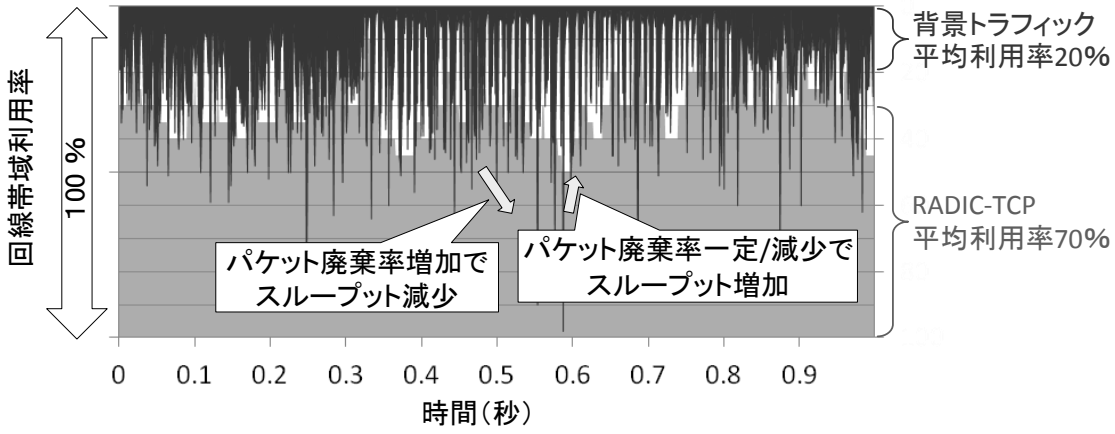


図 47　背景トラフィックと競合したときの RADIC-TCP の動作

用率は、図 47 に示すように、背景トラフィックの使い残した残存帯域(平均 80%)をやや下回る帯域利用率(平均 70%)で安定する。

RADIC-TCP は、パケットペア[142]のように調査パケットを用いる方式や、SABUL[135]や comet TCP[150]のように独自に定義した制御パケットを用いて受信端末が通知するパケット廃棄率を用いて送信帯域を制御する方式とは異なり、TCP パケットのフォーマットに準拠して輻輳制御に必要な情報(RTT、パケット廃棄率とその変化率、送信帯域と廃棄帯域の履歴)を取得し、パケット廃棄率の変化率を用いて輻輳を検出し、2RTT～RTT 前の送信帯域と、直近 RTT の廃棄帯域に基づいて送信帯域を制御する。

輻輳制御するために、独自に定義した調査・制御パケットを用いないので、付加的なトラフィックを発生させずネットワークに負荷を与えない(特徴⑥)。更に、パケット廃棄率の変化率を用いてネットワークの輻輳を検出することで、一時的なパケット廃棄に過敏に反応して帯域が過剰に減少するのを防ぎつつ、輻輳の原因となった 2RTT～RTT 前の送信帯域

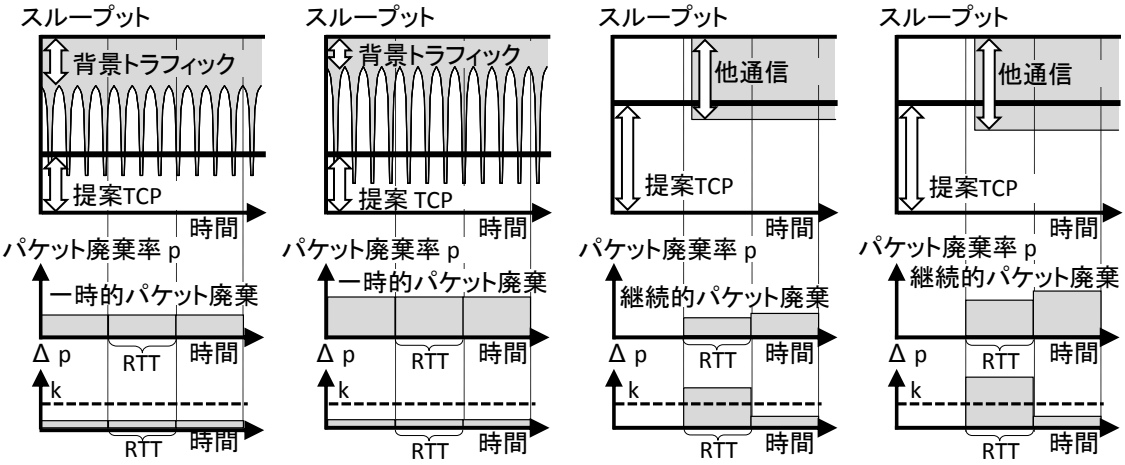


図 48　一時的廃棄と継続的廃棄の識別

と、輻輳発生時の直近 RTT 前の廃棄帯域に応じて送信帯域を減少させることで、回線帯域の利用効率を向上させることができる(特徴⑦)。

送信パケットに対するパケット廃棄発生のフィードバックは、RTT 後の ACK パケット受信で判明するので、計測間隔を RTT よりも大きくすると輻輳検出が遅れる。一方、計測間隔を小さくすると、計測パケット数減少により 1 パケットでも廃棄するとパケット廃棄率が増加しやすくなり、一時的なパケット廃棄による帯域過剰減少が起きやすくなる。計測間隔を主に RTT とし動的に変化させて最適化することで、輻輳検出までの時間を短縮しつつ、一時的なパケット廃棄による帯域過剰減少を抑制することが可能となる。加えて、動的に変化する時間間隔 RTT で計測したパケット廃棄率の変化率に基づいて輻輳を検出することで、一時的なパケット廃棄と継続的なパケット廃棄の識別が可能となる。以下に詳細を説明する。

背景トラフィックとの一時的な競合や、ノイズによるビットエラーが発生する無線環境などで一時的なパケット廃棄が起きる場合、パケット廃棄率 p は背景トラフィックの揺らぎの大きさに応じて変化するが、パケット廃棄率 p の変化率 Δp は小さい(図 48 左側 2 列)。一方、他のトラフィックとの競合により継続的なパケット廃棄が発生した場合、パケット廃棄率 p は競合トラフィックの大きさに依存するが、パケット廃棄率 p の変化率 Δp は大きい(図 48 右側 2 列)。

パケット廃棄率 p の大きさからは、一時的なパケット廃棄と継続的なパケット廃棄を識別できない。一方、パケット廃棄率 p の変化率 Δp は、一時的なパケット廃棄よりも継続的なパケット廃棄の方が大きく、予め定めた閾値 k と比較することで一時的なパケット廃棄と継続的なパケット廃棄の識別が可能となる。更に、継続的なパケット廃棄のみを輻輳発生と判断して送信帯域を減少させることで、一時的なパケット廃棄での送信帯域の過剰減少と、継続的なパケット廃棄での輻輳検出失敗によるパケット廃棄大量発生を防止し、WAN の帯域利用効率の向上が可能となる。

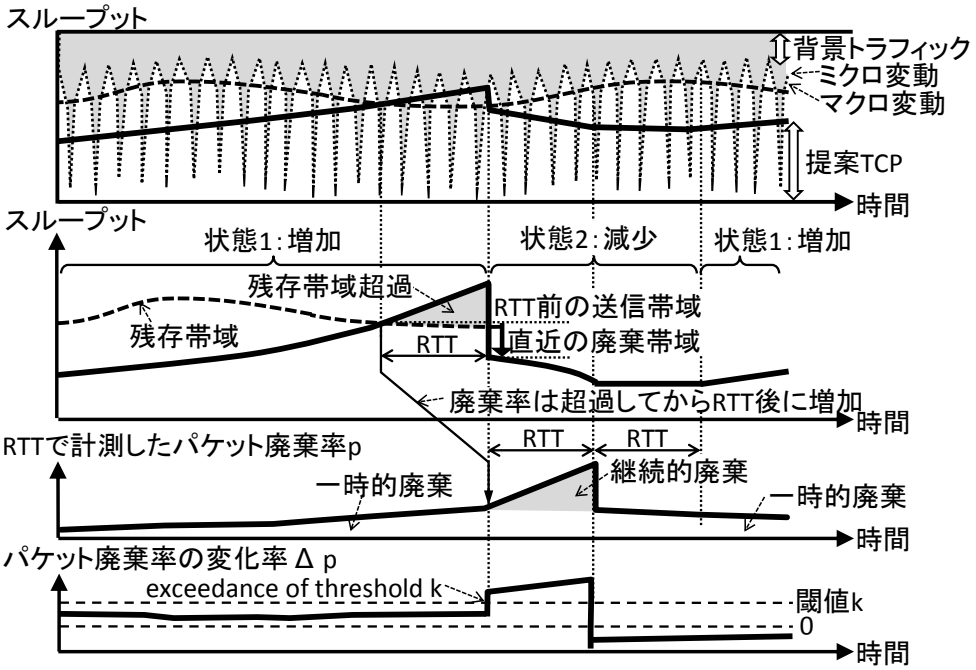


図 49　廃棄率の変化率を用いた輻輳制御

図 49 には輻輳制御の概念図を示す。計測間隔と比較して十分短い RTT 未満のミクロな背景トラフィックの帯域変動との競合により一時的なパケット廃棄が発生する時など、パケット廃棄率 p の変化率 Δp が小さく閾値 k 未満のケースでは、送信帯域を増加させる(ステート 1)。一方、RTT 以上のマクロな背景トラフィックの帯域変動との競合により継続的なパケット廃棄が発生する時など、パケット廃棄率 p の変化率 Δp が大きく閾値 k を超過するケースでは、RTT 前の送信帯域が残存帯域を超過してネットワークに輻輳が発生したと判断し、輻輳の原因となった RTT 前の送信帯域から、輻輳発生時の直近の廃棄帯域を減算した値を、新たな送信帯域とする(ステート 2)。ステート 2 において、パケット廃棄率の変化率が閾値未満となれば、送信帯域の増加を開始する(ステート 1)。

なお、廃棄帯域は、SACK や NACK から推定した廃棄パケットのデータサイズ積算値を、パケット送信時刻と ACK 到着時刻の差から得た RTT で除算することで得る。パケット廃棄率は、直近の廃棄帯域を RTT 前の送信帯域で除算することで得る。

提案 TCP と従来 TCP の輻輳制御についての比較を、輻輳検出方式，輻輳検出後の帯域減少方式，輻輳未検出時の帯域増加方式のそれぞれについて表 7 に示す。

提案 TCP の輻輳制御方式は、従来 TCP と大きく異なる 3 つの特徴を持つ。パケット廃棄率の変化率を用いる輻輳検出方式、パケット廃棄率の変化率が閾値を上回ると RTT 前の送信帯域から直近の廃棄帯域を減算する帯域減少方式、パケット廃棄率の変化率が閾値を下回ると帯域遅延積に比例する時間は線形増加を行い、その後、指数増加する帯域増加方式は、従来 TCP と大きく異なる特徴である。

表 7 輻輳制御まとめ

名称	輻輳検出方式	輻輳検出後の帯域減少方式	輻輳未検出時の帯域増加方式
RADIC-TCP [129][39][151][152]	パケット廃棄率の変化率	パケット廃棄率の変化率の閾値超過時に RTT 前の送信帯域から直近の廃棄帯域を減算	パケット廃棄率の変化率の閾値未達時に帯域遅延積に比例する時間は、線形増加を行い、その後、指数増加
TCP Reno[40]	パケット廃棄発生	パケット廃棄発生時に定率(50%)減少	線形増加(RTT あたり MSS*8/RTT)
High Speed TCP[44]		パケット廃棄発生時に定率(<50%)減少	線形増加(TCP Reno よりも大)
Scalable TCP[133]		パケット廃棄発生時に定率(12.5%)減少	指数増加
BIC TCP[134]		パケット廃棄発生時に定率(12.5%)減少	バイナリ探索に基づく増加と指数増加
CUBIC TCP[42]		パケット廃棄発生時に定率(10%)減少	3 次関数
TCP Hybla[137]		パケット廃棄発生時に定率(50%)減少	RTT 非依存な線形増加 (TCP Reno よりも大)
TCP Westwood[138]		パケット廃棄発生時に ACK 受信間隔に基づく推定帯域に減少	線形増加(TCP Reno と同一)
TCP Star[139]		パケット廃棄発生時に ACK 受信間隔に基づく推定帯域に減少	線形増加(TCP Reno よりも大)
Compound TCP[43]	パケット廃棄発生と通信遅延増加	パケット廃棄発生時に定率(50%)減少 通信遅延が閾値を超過した時に通信遅延増加に反比例して減少	通信遅延の閾値未達時に指数増加 通信遅延の閾値超過時に線形増加
SABUL[135]	パケット廃棄率	パケット廃棄率の閾値超過時に 1, 8, 16, 32, 64…番目のパケット廃棄が発生すると定率(12.5%)減少	パケット廃棄率の閾値未達時に指数増加
TCP Vegas[136]	通信遅延増加	通信遅延が閾値を超過すると線形減少 (RTT あたり MSS*8/RTT)	通信遅延が閾値を下回ると線形増加 (TCP Reno と同一)
Fast TCP[45]		通信遅延増加に反比例して減少	通信遅延減少に反比例して増加

4.4.2 トークンサイズを用いた送信量制御

提案する RADIC-TCP は、ウィンドウサイズ(RTT あたり送信量)の代わりに、4.4.1 節で推定した残存帯域に基づくトークンサイズ(1 ミリ秒あたりの送信量)を制限することで送信量を制御する。RTT が増加すると、RTT 毎の送信データサイズは RTT に応じて増加する。そのため、回線帯域が未使用にも関わらず送信できないアイドリング時間を発生させることなくデータを送信可能であり、送信帯域が増加する(図 50)。

トークンサイズを用いた送信量制御は、トークンバケツアルゴリズム[153]に基づいて送信量を制限する。本アルゴリズムは、1 ミリ秒あたりに送信可能なデータサイズに相当するトークンを、時間経過に応じてトークンバケツに蓄積する。トークンバケツに蓄積したトークンの量がパケット長を超過したら、パケットの送信を許可する。パケット送信後に、送信したパケット長に相当するトークンをトークンバケツから取り除く。本アルゴリズムを用いることで、一定の時間間隔でパケットを送信することが可能となる。トラフィックのスループットが平滑となるので、バースト転送によるパケット廃棄の発生を抑止することが可能となる。

4.4.3 NACK を用いた再送制御

提案する RADIC-TCP は、廃棄が発生すると、SACK の代わりに NACK (Negative

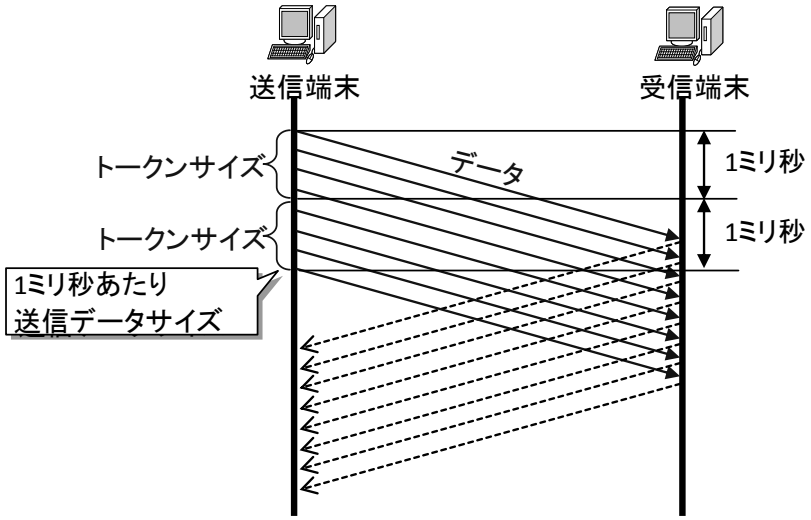


図 50 トークンサイズを用いた送信量制御

ACKnowledge) [154][155][156]を用いた再送を行う(図 51)。SACK と同じ TCP オプションフィールドを NACK として用いる。不連続に受信したデータの位置ではなく、不連続に廃棄したデータの位置を、受信端末から送信端末へ通知するために用いる。廃棄パケットの再送が完了するまで継続的に送信し続ける SACK と異なり、パケット廃棄を検出したら一度だけ NACK を送信して、廃棄したデータの位置を送信端末に通知する。

受信端末は、廃棄したパケットに後続するパケットを受信すると、廃棄したパケットのデータの先頭と末尾のシーケンス番号を、NACK を用いて送信端末に 1 回だけ通知する[129]。送信端末は、NACK 記載のシーケンス番号を用いて廃棄したパケットを再送する。受信端末は、NACK や再送パケットが廃棄するなどして、NACK 送信後に廃棄したパケットが一定期間(例えば 2RTT)受信できないと、NACK を再送する。

送信端末は、不連続に確認応答済みのデータ位置の管理を行わない。不連続に廃棄したパケットが増加しても、NACK 記載のシーケンス番号を用いて廃棄したパケットを再送するだけであり、廃棄したパケットのデータ位置を推定して再送を行うためのシーケンス番号比較が不要である。そのため、パケット廃棄率が高い時の演算量が少なく、スループットを向上させやすい(特徴⑧)。更に、演算量が少ないので、受信端末は、不連続に受信したデータの記録数を制限する必要がなく、送信端末に全ての廃棄箇所を高速に通知可能であり、送信端末は全ての廃棄箇所を 1RTT で再送することが可能となる(図 51)。パケット廃棄検出から再送完了までの時間が短縮され、送信端末と受信端末は再送が完了するまでデータを蓄積しておくためのバッファサイズを削減可能である。加えて、全廃棄箇所の迅速な通知により、送信端末は輻輳発生時の廃棄帯域を正確に推定することが可能となり、残存帯域の推定精度を向上させることができる。

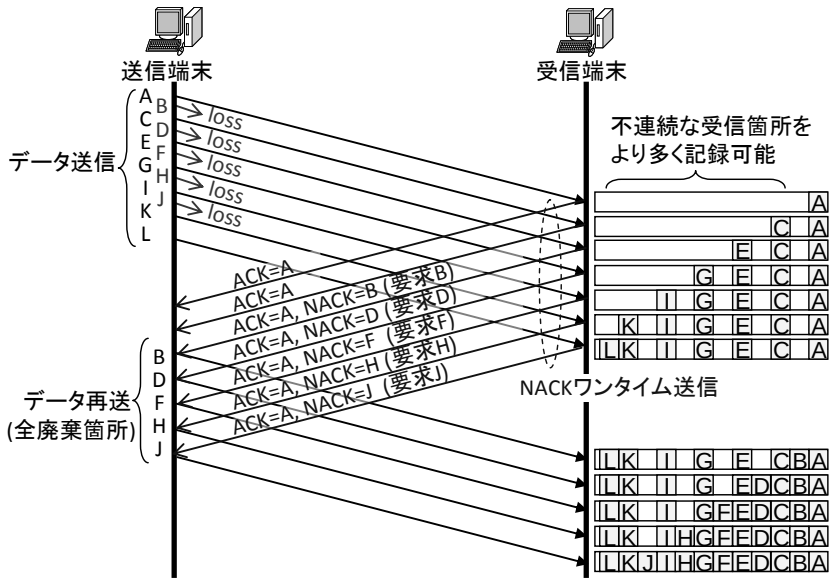


図 51　全廃棄箇所の高速再送

4.5　非対向設置による SACK 併用 RADIC-TCP

対向設置によるRADIC-TCPでは、図 52の左半分に示すように、受信側の装置がパケット廃棄を検出すると、廃棄したパケットのデータの位置を記載したNACKを送信側の装置に1回だけ送信する。送信側の装置は、NACKに基づいて廃棄したパケットを推定して再送する。受信側の装置は、NACK 送信後2RTT待っても再送パケットを受信しない場合は、NACKを再送することで廃棄したパケットの再送を促す。

一方、非対向設置によるRADIC-TCPでは、受信端末がパケット廃棄を検出すると、不連続な受信済データの位置を記載したSACKを、再送が完了するまで送信側の装置に送信し続ける。送信側の装置は、SACKに基づいて廃棄したパケットを推定して再送する(図 52右)。更に、送信側の装置は、SACK記載の不連続な受信済データの先頭と末尾のシーケンス番号から、廃棄したパケットのデータを推定して再送するために、不連続な確認応答済データの先頭と末尾のシーケンス番号を管理する。送信端末がSACKを受信する毎に、SACK記載の不連続な受信済データの先頭と末尾のシーケンス番号と、送信端末が管理する不連続な確認応答済データの先頭と末尾のシーケンス番号の比較を行い、廃棄したパケットのデータを推定して再送を行う。SACKを受信し続けている間は、シーケンス番号の比較を行い続ける必要があるため、演算量が増加してスループットが低下しやすい。

また、提案するRADIC-TCPが用いるパケット廃棄率の変化率に基づく輻輳制御では、送信端末においてパケット廃棄率の計算が必要となるが、パケット廃棄率の計算方法は、対向設置方式と非対向設置方式の間で異なる。

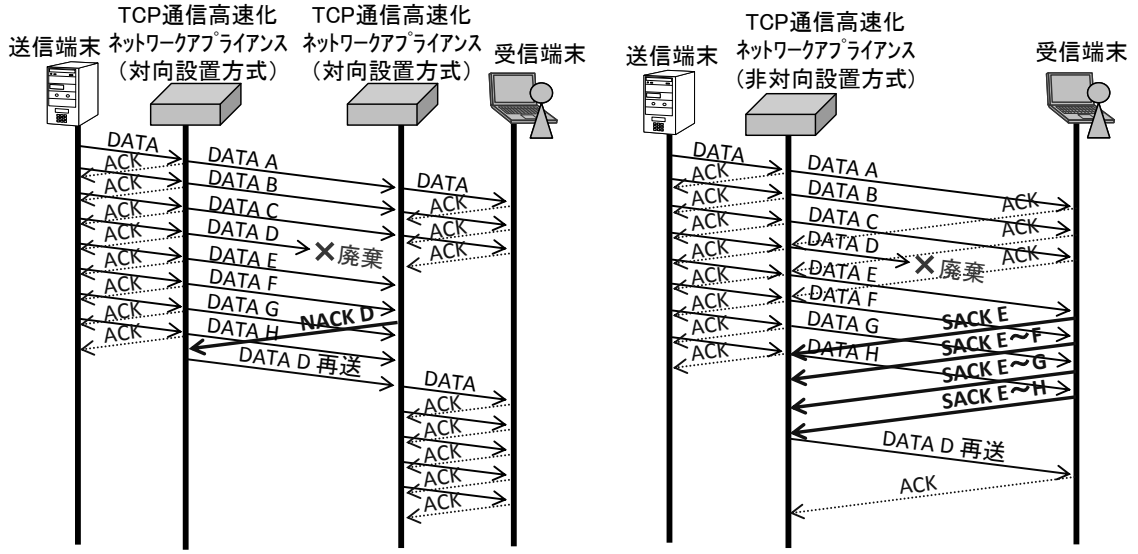


図 52　廃棄パケットの再送制御の比較

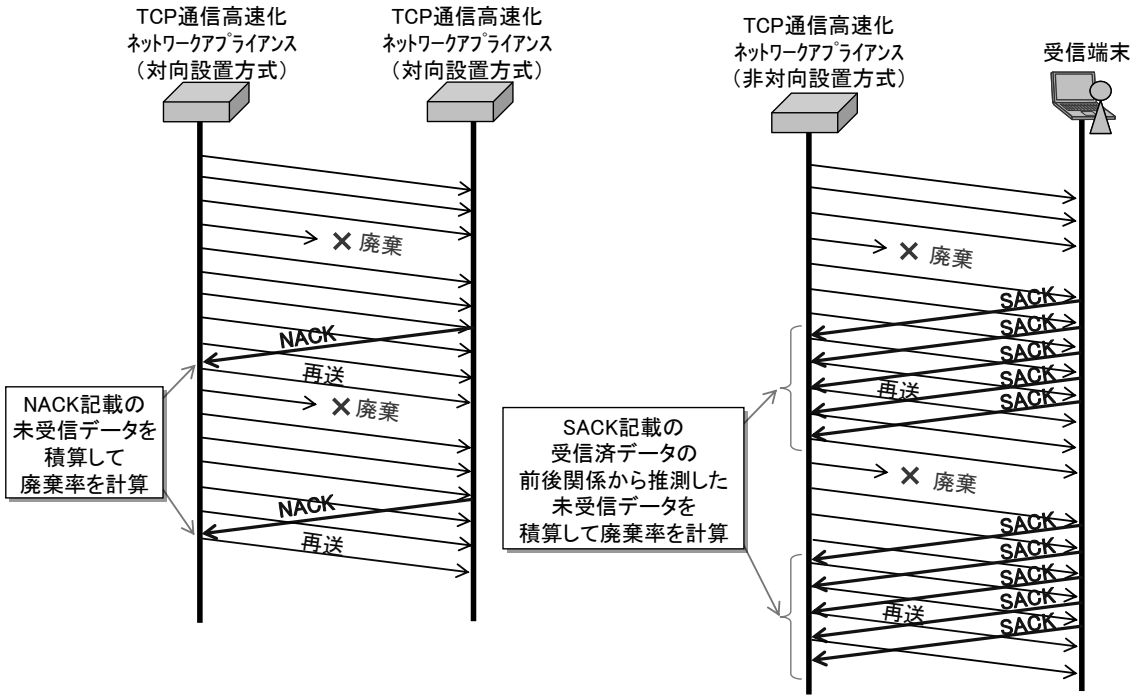


図 53　廃棄率の計算方法

対向設置方式は、対向装置から受信した NACK に記載の未受信データのサイズを積算することによりパケット廃棄率を計算する(図 53 左)。一方、非対向設置方式は、対向端末から受信した SACK に記載の受信済データの前後関係から推定した未受信データのサイズを積算することによりパケット廃棄率を計算する(図 53 右)。送信側の装置において、不連続な確認応答済データを記録しておき、SACK を受信する毎に SACK 記載の不連続な受信済

表 8　提案 TCP と従来 TCP の比較

	再送制御	輻輳制御			送信量制御
		輻輳検出	輻輳時の帯域減少	非輻輳時の帯域増加	
対向設置による RADIC-TCP [129][39][151]	NACK	パケット廃棄率の変化率	パケット廃棄率の変化率が閾値を上回ると、RTT 前の送信帯域から直近の廃棄帯域を減算	パケット廃棄率の変化率が閾値を下回ると帯域遅延積に比例する時間は線形増加を行い、その後、指数増加	トークンサイズ
非対向設置による SACK併用RADIC-TCP [152]	SACK				
TCP Reno[40]	SACK	パケット廃棄発生	パケット廃棄発生時に定率(50%)減少	線形増加 (RTT あたり MSS*8/RTT)	ウィンドウサイズ
High-Speed TCP[44]			パケット廃棄発生時に定率(<50%)減少	線形増加 (TCP Reno よりも大)	
Scalable TCP[133]			パケット廃棄発生時に定率(12.5%)減少	指数増加	
BIC TCP[134]			パケット廃棄発生時に定率(12.5%)減少	バイナリ探索に基づく増加と指数増加	
CUBIC TCP[42]			パケット廃棄発生時に定率(10%)減少	3 次関数	
TCP Hybla[137]			パケット廃棄発生時に定率(50%)減少	RTT 非依存な線形増加 (TCP Reno よりも大)	
TCP Westwood[138]			パケット廃棄発生時に ACK 受信間隔に基づく推定帯域に減少	線形増加 (TCP Reno と同一)	
TCP Star[139]			パケット廃棄発生時に ACK 受信間隔に基づく推定帯域に減少	線形増加 (TCP Reno よりも大)	
TCP Vegas[136]		通信遅延増加	通信遅延の閾値超過時に線形減少 (RTT あたり MSS*8/RTT)	通信遅延の閾値未達時に線形増加 (TCP Reno と同一)	
Fast TCP[45]			遅延増加に反比例して減少	遅延減少に反比例して増加	
Compound TCP[43]		パケット廃棄発生と通信遅延増加	パケット廃棄発生時に定率(50%)減少 通信遅延の閾超過時に遅延増加に反比例して減少	通信遅延の閾値未達時に指数増加 閾値超過時に線形増加	

データと比較して、未受信データを推定することが必要となるため、対向設置方式と比較して必要な演算量が増加し、TCP 通信高速化を行うネットワークアプライアンスの最大性能が減少する。その一方で、一般的な端末がサポートしている SACK を用いるので、受信側への装置設置が不要であり、スマートフォン端末やタブレット PC など不特定多数のモバイル端末と直接通信を行いつつ、送信方向の通信を高速化することが可能となる。

提案するRADIC-TCPと、非対向設置によるSACK併用RADIC-TCPと、従来TCPの比較を表 8に示す。RADIC-TCPの再送制御をSACKに対応させて、輻輳制御のパケット廃棄率の計算にSACKを用いるように変更することで、非対向設置によるSACK併用RADIC-TCPを実現した。非対向設置方式のRADIC-TCPを利用することで、一般的な端末と直接通信しつつ、受信側の端末やアプリケーションに依存せずにTCP通信を用いたデータ送信を高速化することが可能となった。

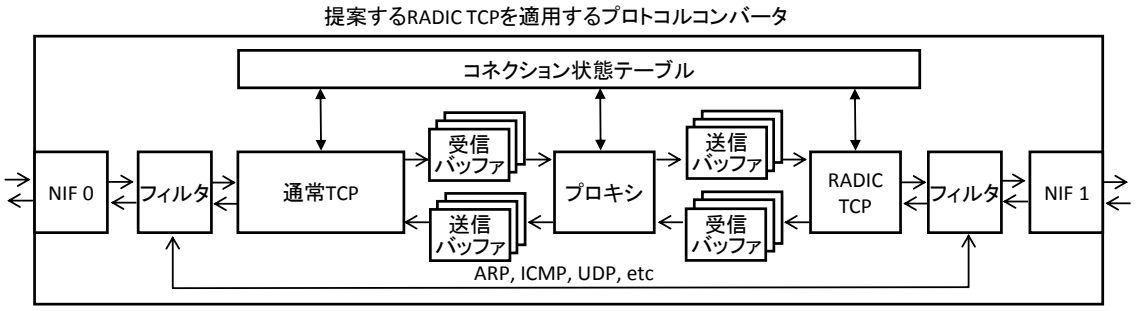


図 54　通常 TCP を提案する RADIC-TCP に変換するプロトコルコンバータ

4.6　評価実験

4.6.1　実験環境における評価

(1)通常 TCP を提案 TCP に変換するコンバータ装置

実験評価では、通常TCPを提案TCPに変換するプロトコルコンバータ(図 54)を実装して用いた。本装置は、TCP Renoを用いる従来TCPと、提案するRADIC-TCPと、送受信バッファと、コネクション状態テーブルと、フィルタと、2つのネットワークインターフェース(NIF)とプロキシから成る(図 54)。コネクション状態テーブルは、TCPコネクションとバッファを管理するための値を記録する。プロキシは、受信バッファの整列済みデータを送信バッファへ転送する。フィルタは、非TCPパケットをパススルーさせる。本装置は、他の端末からはブリッジのように動作しているように見える。NIF0側に接続する端末は、通常TCPを用いて通信しているつもりが、実際には提案するRADIC-TCPに変換して通信が行われる。通常TCP側の送受信バッファのサイズは64Kバイトに設定した。

(2)実験評価で使用した構成

実験評価で使用した基本構成は図 55に示す通りである。.提案するRADIC-TCPを一般的な端末でも利用可能にする2つのプロトコルコンバータ(標準的なPCを使用)と、プロトコルコンバータ間に設置して通信遅延とパケット廃棄を発生させることでWANを模擬するネットワークエミュレータと、プロトコルコンバータ経由でRADIC-TCPを用いて通信を行う2つのPC端末と、従来TCPを用いて通信を行う2つのPC端末から成る。従来TCPとして、IETF標準のTCP Renoと、Linuxカーネル2.6.18-128が標準実装するCUBIC-TCP と、商用Windows OSが標準実装するCompound TCPと、遅延ベースのTCPとして報告があるFast TCPと、帯域過剰減少を抑制する手法を持つ廃棄発生ベースのTCPであるTCP StarをSACKと共に用いた。CUBIC-TCPの送受信バッファのサイズは、手動で変更した。TCP Renoは、一般的な共通の環境を模擬するために、SACKと共にデフォルト設定で用いた。本構成を用いることで、RADIC-TCPとその他通常TCPの性能を評価した。

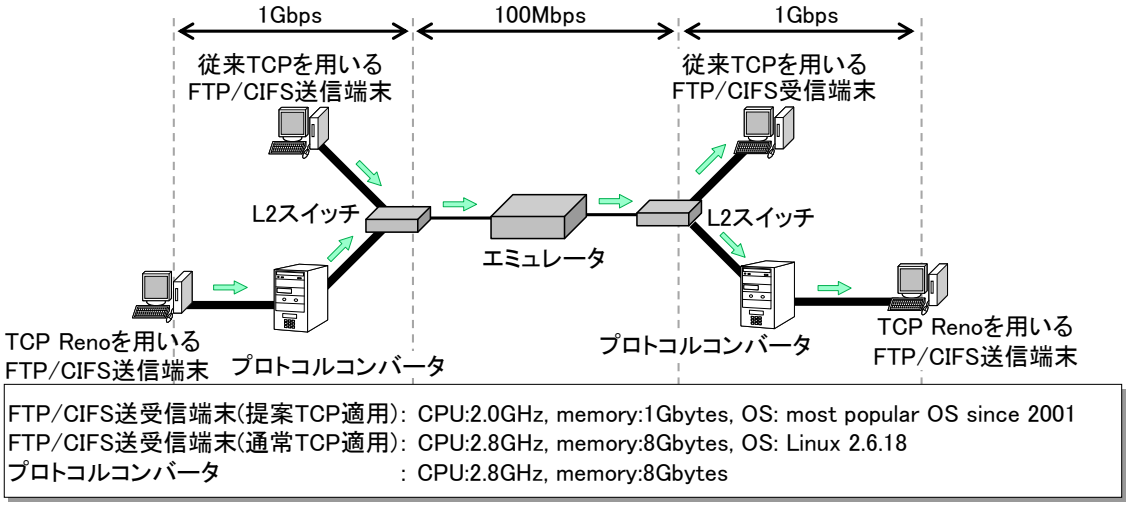


図 55　実験評価で使用した基本構成

表 9　実験評価の項目

#	項目	想定ケース	比較対象	比較対象の選択理由
(3)	スループットの RTTへの依存性	遠隔拠点との通信	CUBIC-TCP Compound-TCP	一般的なTCPが通信遅延に 依存することを示すため
(4)	スループットの 一時的なパケット廃棄への依存性	背景トラフィックとの 一時的競合	CUBIC-TCP Compound-TCP TCP-Star SABUL	パケット廃棄発生や パケット廃棄率ベースのTCPが 一時的なパケット廃棄による 理論上限に従うことを示すため
(5)	スループットの 継続的なパケット廃棄への依存性	背景トラフィックとの 継続的競合	Fast-TCP	通信遅延ベースのTCPが 継続的なパケット廃棄を 検出できない場合を示すため
(6)	スループットの 残存帯域変化への依存性	変化する背景トラフィック との競合	CUBIC-TCP	一般的なTCPの残存帯域への 追従が困難なことを示すため
(7)	スループットのボトルネックの キュー長への依存性	ボトルネックのある ネットワーク	CUBIC-TCP	従来TCPのウィンドウサイズ制御 がボトルネックのキュー長の 影響を受けることを示すため
(8)	スループットの ウィンドウサイズへの依存性	ウィンドウサイズを手動で 大きい値に設定したケース	CUBIC-TCP	従来TCPでウィンドウサイズが スループットに影響を 与えない場合を示すため
(9)	提案TCPと標準TCPの間の公平性	提案TCPと標準TCPの併用	TCP Reno	IETF標準のTCPであり、 公平性の基準となるため
(10)	異なるRTTを持つ通信間の公平性	複数の遠隔拠点との通信	—	—
(11)	多数通信間の帯域割当	多数の通信コネクション	—	—

実験評価を行った9つの項目を表 9に示す。各評価項目において、想定するケースや、比較対象として選択した従来TCPとその根拠も併せて示す。

まず初めに、遠隔拠点と通信するケース(3)を想定して、スループットのRTTへの依存性を評価した。通信遅延への依存性が一般的であることを示すため、一般的なTCPを比較対象として用いた。背景トラフィックとの一時的競合が発生するケース(4)の想定評価では、スループットの一時的なパケット廃棄への依存性を評価した。パケット廃棄発生やパケット廃棄率ベースのTCPが一時的なパケット廃棄による理論上限に従うことを示すため、パケット廃棄率ベースのSABULや、帯域過剰減少を抑制する手法を持つ廃棄発生ベースのTCPであ

るTCP StarをSACKと共に比較対象として用いた。背景トラフィックとの継続的な競合が発生するケース(5)の想定評価では、スループットの継続的なパケット廃棄への依存性を評価した。通信遅延ベースのTCPが継続的なパケット廃棄を検出できない場合を示すため、残存帯域への追従性能が最も高い通信遅延ベースのFast TCPを比較対象として用いた。帯域変化の激しい背景トラフィックとの競合が発生するケース(6)の想定評価では、スループットの残存帯域への追従性を評価した。従来TCPのスループットが、残存帯域へ追従しないことを示すため、一般的なTCPを比較対象として用いた。ボトルネックのあるネットワークを利用するケース(7)の想定評価では、スループットのボトルネックのキュー長への依存性を評価した。従来TCPのウィンドウサイズ制御がボトルネックのキュー長の影響を受けてスループットが減少しやすいことを示すために、一般的なTCPを比較対象として用いた。ウィンドウサイズを手動で大きい値に設定して通信するケース(8)の想定評価では、スループットのウィンドウサイズへの依存性を評価した。従来TCPでは、ウィンドウサイズがスループットに影響を与えない場合があることを示すために、一般的なTCPを比較対象として用いた。提案するRADIC-TCPと標準TCPを併用するケース(9)の想定評価では、提案TCPと標準TCPの公平性を評価した。公平性を評価するために、IETF標準のTCPを比較対象として用いた。複数の遠隔拠点と通信するケース(10)や、多数の通信コネクションが同時に通信するケース(11)の想定評価では、提案するRADIC-TCPが互いに公平性を確保して動作することを確かめた。

(3)スループットのRTTへの依存性

提案するRADIC-TCPのスループットのRTTへの依存性を評価するために、(2)の図 55に記載の実験環境を用いてRTTとスループットの関係を評価した。

日米間のインターネットの平均RTTは100～200msであり、日本とブラジルの間では300ms程度にまで増加する。ネットワークのルータやスイッチにおけるバッファリング時間を含めれば、200～400msになると推定される。これらの値に従い、エミュレータにおいて10～500 msのRTTを発生させた。図 56に、シングルコネクションのFTPを用いて146MBのファイルを転送する評価によって得られたRTTとスループットの関係を示す。

送受信バッファが 128KB のとき、CUBIC-TCP のスループットは、送受信バッファサイズを RTT によって除算することで得られる理論値に従う。送受信バッファが 4～16MB に増加すると、CUBIC-TCP のスループットは増加するが、RTT200ms 以上のときは、最大回線帯域 100Mbps の 70%に満たない。これは、RTT が大きいときのウィンドウサイズの増加スピードが遅く、転送開始からトップスピード到達までの時間が延びることで、平均値が減少するためである。更に、送受信バッファサイズに応じてウィンドウサイズが増加すると、バーストトラフィックが発生しやすくなり(図 44 参照)、回線帯域が 1Gbps から 100Mbps へと狭くなるボトルネック箇所において高頻度でパケット廃棄を引き起こす。その一方で、RADIC-TCP は、送受信バッファが 4～16MB のときに、CUBIC-TCP よりも大きいスループットを実現した。送受信バッファが 4MB で RTT が 500ms のとき、RADIC-TCP は理論値

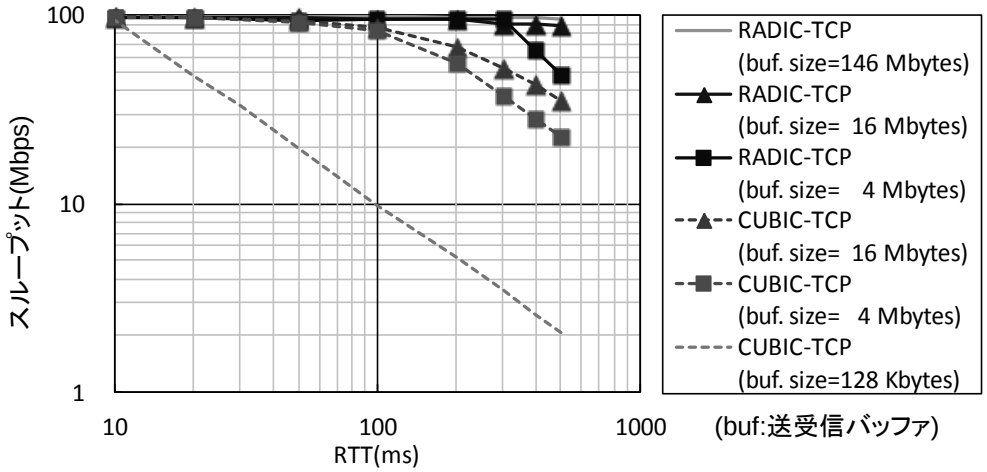


図 56　100Mbps 回線における RTT とスループットの関係

64Mbps の 77%に相当する 49Mbps のスループットを実現した。

次に、100Mbps 回線を 1Gbps 回線に変更して、代表的な汎用アプリ FTP と CIFS (Common Internet File System)のスループットの RTT への依存性を評価した。図 55 に示す評価構成において、エミュレータの帯域制御を OFF とし、WAN 環境でのパケット廃棄を模擬するため平均 0.01%のパケット廃棄を発生させた。RTT を 1～500ms に変化させて、Linux の FTP や、Windows 7 の CIFS を用いて 1Gbytes のファイルを転送したときの RADIC-TCP と CUBIC-TCP と Compound TCP (CTCP)のスループットを測定した。TCP のウィンドウサイズ(送受信バッファ)最大値を FTP では 16Mbytes、CIFS では 8Mbytes としたときの結果を図 57 と図 58 に示す。

CUBIC-TCP を用いた FTP は、RTT 増加に伴いスループットが 10Mbps まで低下する。CTCP を用いた CIFS は、RTT の増加に伴いスループットが 3Mbps まで低下する。一方、RADIC-TCP は RTT 増加に伴うスループット低下が緩やかで、ウィンドウサイズと RTT で定まる限界に近づいた。RADIC-TCP を用いた FTP は、RTT 50ms 以上で CUBIC-TCP の 10 倍以上、RADIC-TCP を用いた CIFS は、RTT 20ms 以上で CTCP の 10 倍以上のスループットを実現した。

(4)スループットの一時的なパケット廃棄への依存性

提案する RADIC-TCP の一時的なパケット廃棄率への依存性を評価するために、実験環境を用いてパケット廃棄率とスループットの関係を評価した。

日本と北米東海岸の間のインターネットの RTT は平均で約 200ms であり、パケット廃棄率は 0.1～1%程度であった。これらの測定値に基づいて、エミュレータにおいて、200ms の RTT と、往路と復路のそれぞれに 0.001 から 10%のランダム廃棄を発生させた。シングルコネクションの FTP を用いて 146MB のファイルを転送する評価により測定したパケット廃棄率とスループットの関係を図 59 に示す。

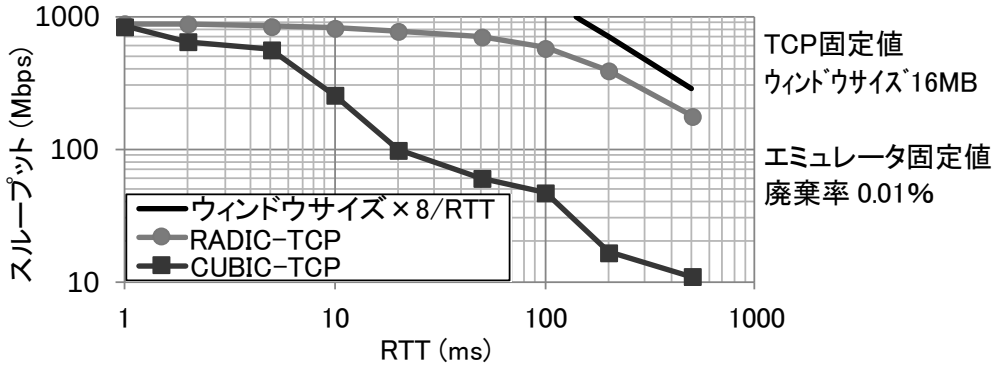


図 57　FTP スループットの RTT 依存性

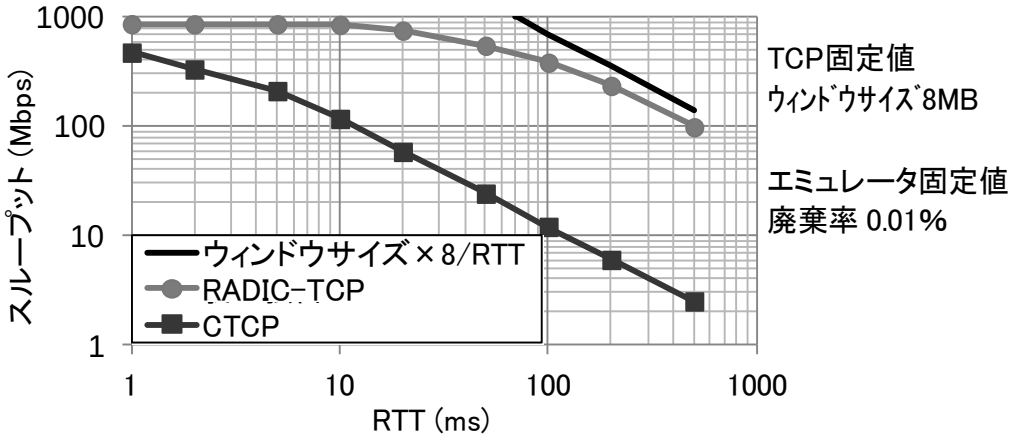


図 58　CIFS スループットの RTT 依存性

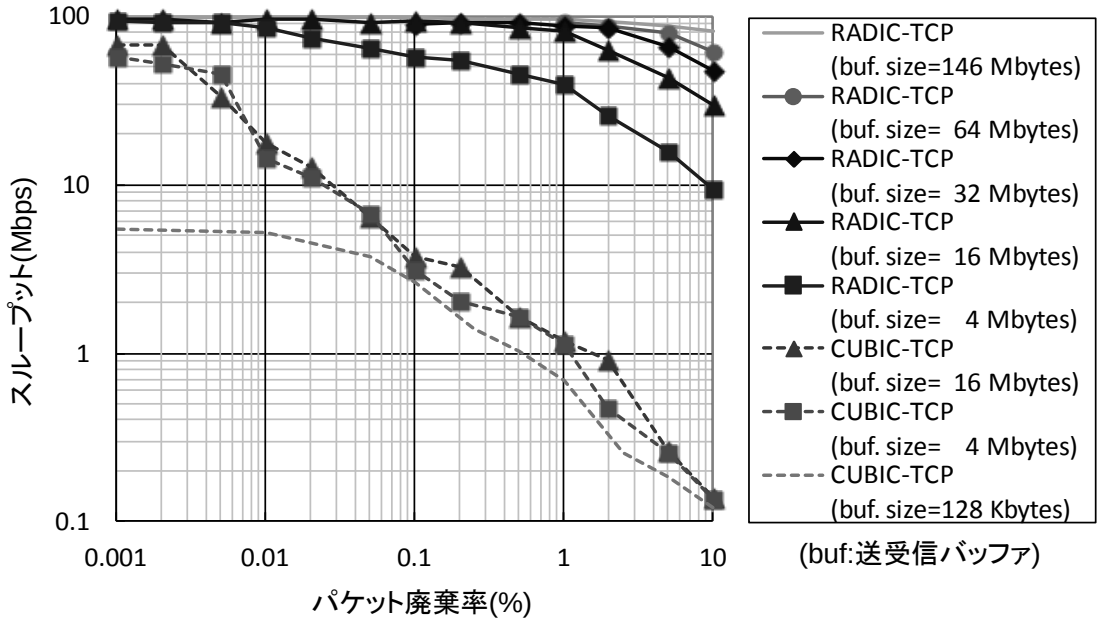


図 59　100Mbps 回線におけるパケット廃棄率とスループットの関係

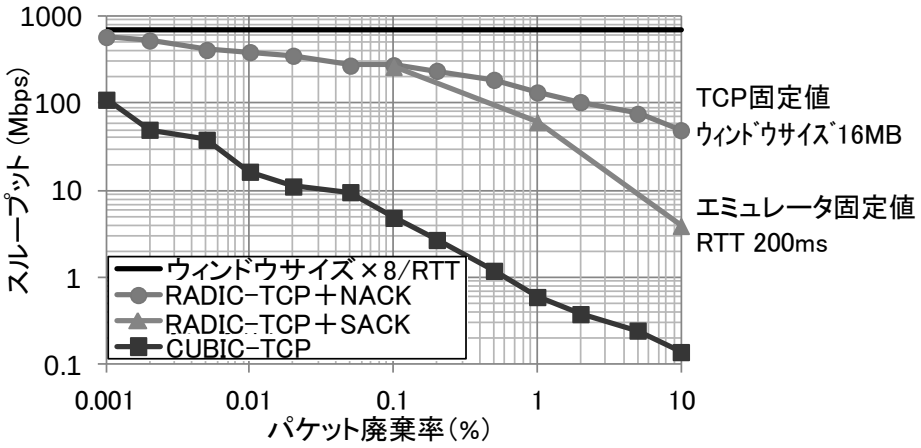


図 60　パケット廃棄率とスループットの関係

送受信バッファが 128KB のとき、CUBIC-TCP のスループットは、パケット廃棄率が 0.01%以上の領域で、パケット廃棄率が 10 倍になる毎に 3 分の 1 に減少した。送受信バッファが 4～16MB のときも、CUBIC-TCP のスループットは、パケット廃棄率が 0.001%以上の領域でパケット廃棄の増加に応じて減少した。パケット廃棄率が 0.1%のときは、送受信バッファを 128KB の 32, 128 倍に増加させても、スループットの改善効果は 2 倍未満となった。一方、提案する RADIC-TCP のスループットは、送受信バッファが 4～16MB のとき、パケット廃棄率がインターネットの平均に相当する 0.1～1%のときで、CUBIC-TCP より 10 倍以上大きかった。送受信バッファが 32, 64, 146MB へと増加するに従い、パケット廃棄率 1%以上の領域でスループットが増加した。

次に、100Mbps 回線を 1Gbps 回線に変更して、スループットの一時的なパケット廃棄への依存性を評価した。図 55 の評価構成において、エミュレータの帯域制御を OFF とし、日本－北米東海岸に相当する 200ms の RTT を発生させた。パケット廃棄率を 0.001～10%に変化させて、FTP を用いて 1Gbytes のファイルを転送したときの RADIC-TCP と CUBIC-TCP のスループットを測定した。TCP のウィンドウサイズ(送受信バッファ)最大値を 16Mbytes としたときの結果を図 60 に示す。

CUBIC-TCP はパケット廃棄率の増加に伴いスループットが 0.1Mbps まで低下する一方で、提案する RADIC-TCP はパケット廃棄率の増加に伴うスループット低下が緩やかで、ウィンドウサイズと RTT で定まる限界に近づいた。パケット廃棄率 0.01%以上で、CUBIC-TCP の 10 倍以上のスループットを実現した。

また、SACK による再送(4.3.3 節記載)を用いた RADIC-TCP と、NACK による再送(4.5 節の図 52 記載)を用いた RADIC-TCP を比較すると、パケット廃棄率が 0.1%を越えたときにスループット差が大きくなり、パケット廃棄率 10%でのスループット差は 10 倍となった。提案する RADIC-TCP は、SACK による再送を用いた非対向設置方式により、一般的な端末と直接通信しつつ、スループットを向上させることができる他、NACK による再送を用い

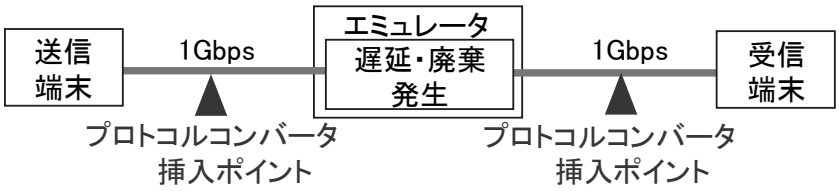


図 61　一時的なパケット廃棄の発生する環境を模擬する構成

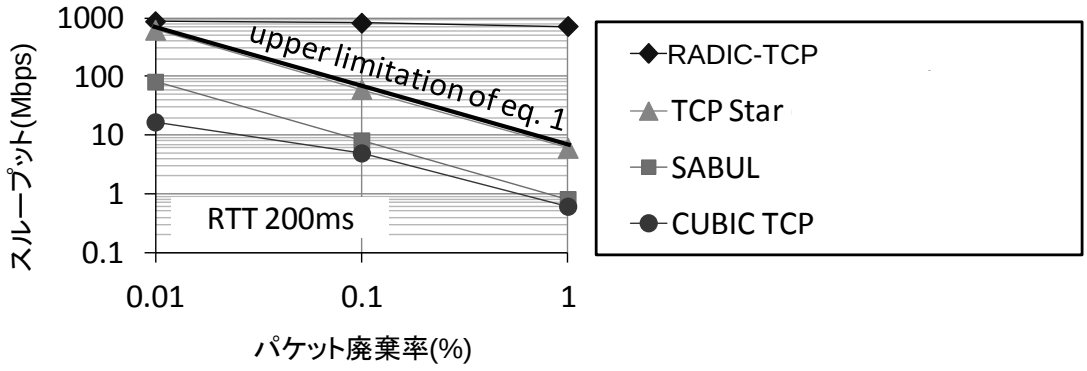


図 62　RADIC-TCP と従来 TCP のスループット比較

た対向設置方式により、スループットをより一層向上させることが可能であった。

最後に、パケット廃棄発生に基づいて輻輳を検出する従来 TCP のスループットと 4.3.1 節に記載の理論上限値を表す式(4-1)との一致性や、パケット廃棄率の変化率に基づいて輻輳を検出する提案 TCP の優位性を示すため、図 61 に示す構成を用いて実験評価を行った。全リンクを 1Gbps とし、エミュレータにて日米間 WAN に相当する往復 200ms の通信遅延を発生させた。標準 TCP を RADIC-TCP に変換するプロトコルコンバータは端末とエミュレータの間に挿入した。

従来 TCP として、文献[42][139][135]記載の CUBIC-TCP，TCP-Star，SABUL の制御方式を実装して評価に用いた。CUBIC-TCP は、パケット廃棄発生時に送信帯域を定率減少させる代表例であり、減少率が 10%と最も小さい。TCP-Star は、パケット廃棄発生時に送信帯域の減少幅を ACK 受信速度に応じて最適化する代表例である。SABUL は、パケット廃棄率に基づき輻輳を検出する代表例である。

TCP のウィンドウサイズ(送受信バッファ)最大値を 256MB として、往路と復路のそれぞれで廃棄を 0.01，0.1，1%でランダム発生させた時のスループットを図 62 に示す。MSS (Maximum Segment Size)は 1460 bytes とした。

TCP Star のスループットは、パケット廃棄発生に基づく輻輳検出方式の式(4-1)の上限値とほぼ一致し、CUBIC TCP と SABUL のスループットは、式(4-1)の上限値を下回った。パケット廃棄発生やパケット廃棄率に基づく輻輳検出方式は、一時的なパケット廃棄でも輻輳発生と判断するため、帯域を過剰に減少させ、回線帯域を使い切れないことを確認した。

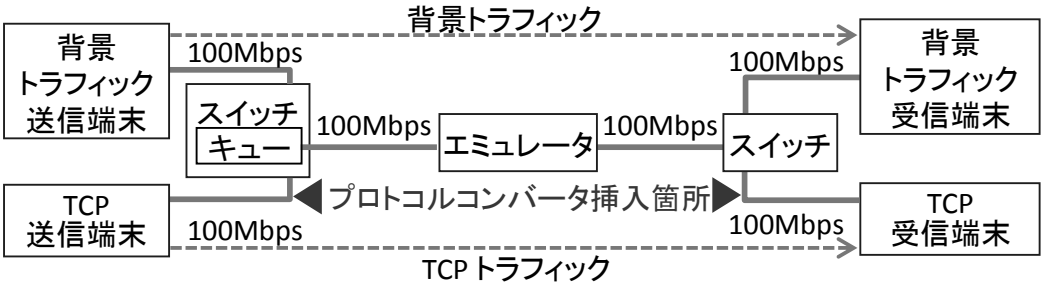


図 63　継続的なパケット廃棄を模擬する評価環境

一方、一時的なパケット廃棄を輻輳発生と見なさない RADIC-TCP のスループットは、712～860Mbps となり、4.3.1 節の式(4-1)の上限値の 1.4～116 倍となった。日米間 WAN に相当する RTT 200ms, 廃棄 0.1%において、従来 TCP と比較して帯域利用率を 13～165 倍向上させた。

(5)スループットの継続的なパケット廃棄への依存性

他のトラフィックとの競合により継続的なパケット廃棄が発生する環境でも、提案する RADIC-TCP が残存帯域に追従して送信帯域を変更できることを示し、通信遅延増加に基づき輻輳検出する従来 TCP よりも優位なことを示す。

通信遅延増加に基づき輻輳検出する従来 TCP として、文献[45]記載の Fast TCP の方式を用いた。TCP Vegas[136]よりも帯域増減幅が大きく、残存帯域への追従能力が優れているためである。送信帯域を決定するための式(4-2) (4.3.1 節)の変数 α は 0.05 とした。

図 63 に示す構成でシミュレーション評価を行った。競合によるパケット廃棄を模擬する合流箇所としてスイッチを備え、全回線を 100Mbps でリンクアップさせ、エミュレータにて、日米間 WAN に相当する往復 200ms の通信遅延を発生させた。標準 TCP を RADIC-TCP に変換するプロトコルコンバータを端末とスイッチの間に挿入した。

合流箇所のスイッチのキュー長を 1MB として、20Mbps の背景トラフィックと 1.6 秒毎に 0.4 秒間競合する環境をケース 1 とする。一方、合流箇所のスイッチのキュー長を 128KB として、50Mbps の背景トラフィックと 1.6 秒毎に 0.4 秒間競合する環境をケース 2 とする。ケース 1 は通信遅延が大きくてもパケット廃棄が少ないケースを模擬し、ケース 2 は通信遅延が小さくてもパケット廃棄が多いケースを模擬する。これら 2 つのケースについて TCP の送信帯域を評価した。ケース 1, 2 における RADIC-TCP と Fast TCP の送信帯域の時間推移を図 64 と図 65 に示す。

ケース 1 では、Fast TCP と背景トラフィックの競合により、RTT は最大で約 260ms となるがパケット廃棄は発生しない。パケット廃棄が発生しないにも関わらず、Fast TCP は輻輳発生と判断して、送信帯域を最小 73Mbps にまで低下させる。一方、ケース 2 では、Fast TCP と背景トラフィックの競合により、RTT は最大で 210ms となり、1 回の競合で約 2.4MB 分

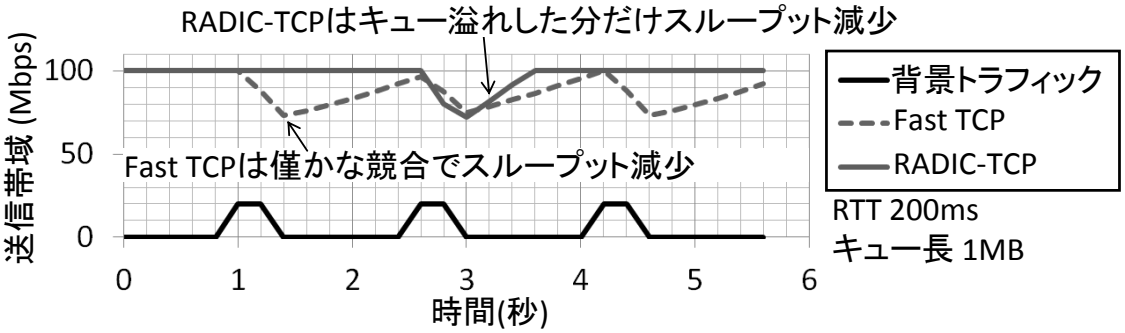


図 64　遅延が大きくて廃棄が少ないケース 1

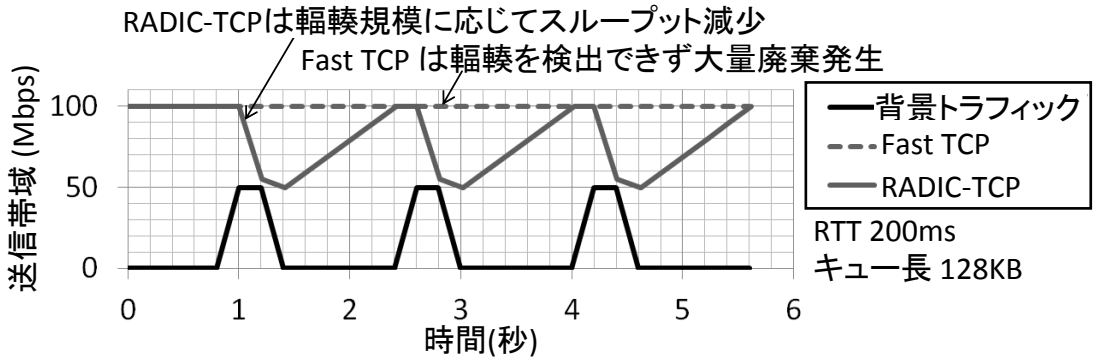


図 65　遅延が小さくて廃棄が多いケース 2

のパケット廃棄が発生する。大量のパケット廃棄が発生するにも関わらず、Fast TCP は輻輳発生と判断せず、送信帯域を減少させない。

上記のように、Fast TCP は、キュー長が大きいときの背景トラフィックとの一時的な競合による送信帯域の過剰減少や、キュー長が小さいときの背景トラフィックとの競合によるパケット廃棄の大量発生を引き起こす場合がある。

一方、RADIC-TCP は、ケース 1 の 1, 3 回目の競合ではスイッチのキューが溢れず、パケット廃棄が発生しないので、パケット廃棄発生ベースの TCP 同様に、送信帯域を減少させない。ケース 1 の 2 回目の競合では、スイッチのキューが溢れて、パケット廃棄の発生によりパケット廃棄率が増加するので、廃棄帯域の大きさに応じて送信帯域を減少させる。ケース 2 の 1, 2, 3 回目の競合では、スイッチのキューが溢れて、パケット廃棄の発生によりパケット廃棄率が増加するので、廃棄帯域の大きさに応じて送信帯域を減少させる。

上記のように、RADIC-TCP は、一時的なパケット廃棄が発生したときに送信帯域を過剰に減少させない一方で、継続的なパケット廃棄が発生したときは輻輳発生と判断して廃棄帯域に応じて送信帯域を減少させることでパケット廃棄の大量発生を防ぎ、帯域利用率を向上させる。

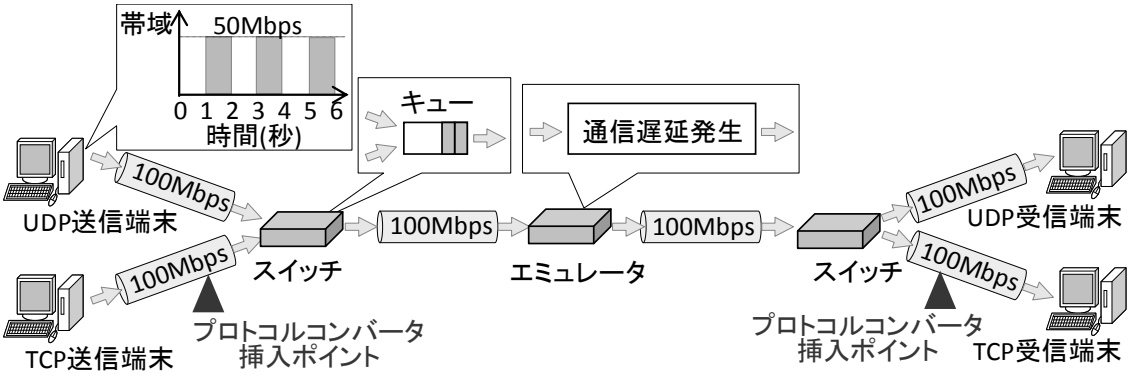


図 66　残存帯域が 1 秒毎に変化する評価環境

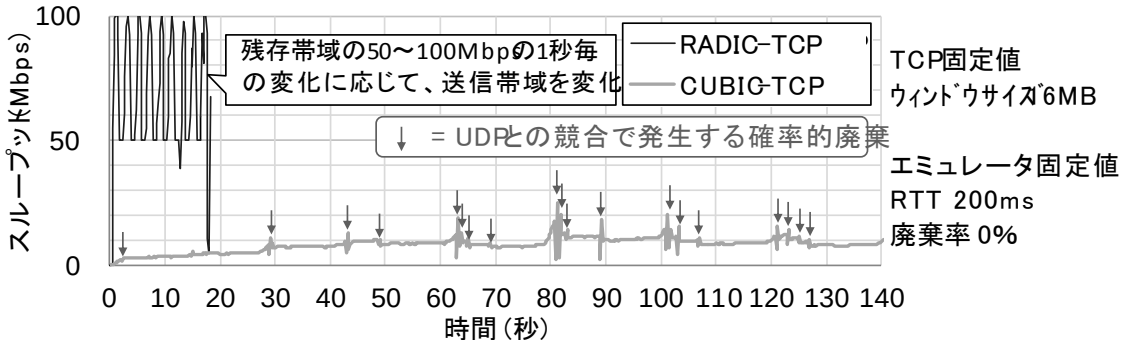


図 67　残存帯域変化時のスループット

(6)スループットの残存帯域変化への依存性

提案する RADIC-TCP が、残存帯域が急激に変化する環境において、残存帯域を使い切る能力を評価した。評価構成を図 66 に示す。エミュレータで往復 200ms の通信遅延を発生させ、UDP トラフィックと TCP トラフィックが合流するスイッチのキュー長を 128KB、フローコントロールを OFF とした。UDP トラフィックの 50Mbps での送信と停止を 1 秒毎に交互に行うことで、残存帯域が 50Mbps と 100Mbps の間で 1 秒毎に交互に変化する環境を作成した。RADIC-TCP によるプロトコルコンバータは端末とスイッチの間に挿入した。

FTP を用いて 146Mbytes のファイルを転送したときの RADIC-TCP と CUBIC-TCP のスループットの時間変化を測定した。TCP のウィンドウサイズ(送受信バッファ)最大値を 16Mbytes としたときの結果を図 67 に示す。図 68 には、RADIC-TCP のスループット詳細を示した。

従来 TCP の CUBIC-TCP は、廃棄発生に基づく輻輳制御を用いるため、UDP トラフィックとの競合で発生する一時的なパケット廃棄により、送信帯域を過剰に減少させやすく、平均 75Mbps ある残存帯域のうち 10Mbps(13％)程度の使用となった(図 67)。一方、

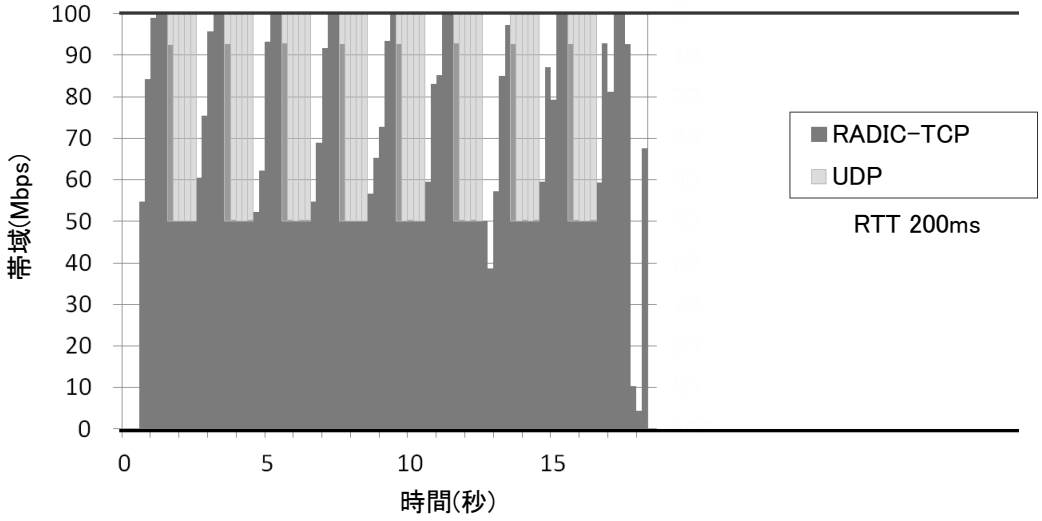


図 68　残存帯域変化時の RADIC-TCP のスループット詳細

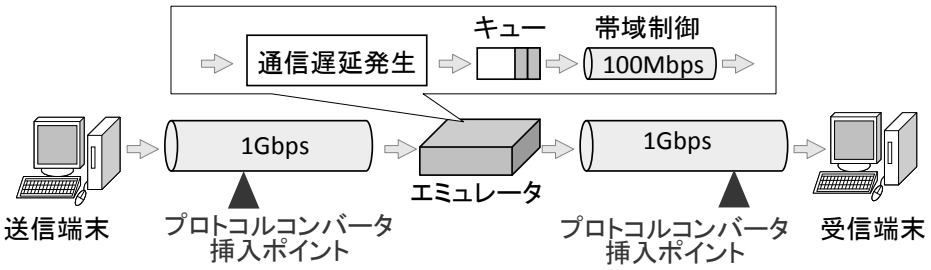


図 69　ボトルネックの存在する評価環境

RADIC-TCP は、パケット廃棄率の変化率に基づく輻輳制御方式を用いることで、一時的なパケット廃棄に依存せず、UDP トラフィックの使い残した残存帯域の 50～100Mbps の 1 秒毎の変化に応じて、送信帯域を変化させ、平均 75Mbps ある残存帯域のうち 68Mbps(90％)を使い切ることができた(図 67)。図 68 からは、RADIC-TCP が、競合の発生から RTT 後に送信帯域を減少させ、競合が解消してから RTT 後に送信帯域を増加させている動作を確認した。提案する RADIC-TCP が、背景トラフィックとの競合時に想定通りの動作をしていることを確認した。

(7)スループットのボトルネックのキュー長への依存性

提案する RADIC-TCP のスループットのボトルネックのキュー長への依存性を評価した。評価構成を図 69 に示す。送受信端末とエミュレータを 1Gbps でリンクアップさせ、エミュレータにて 100Mbps に帯域制御して、往復 200ms の通信遅延を発生させた。標準 TCP を RADIC-TCP へ変換するプロトコルコンバータは端末とエミュレータの間に挿入した。キュ

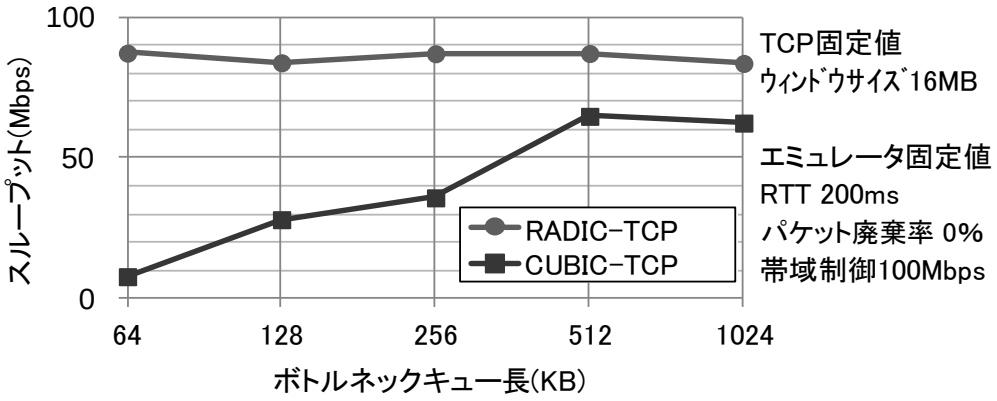


図 70　キュー長とスループットの関係

一長を 64～1024Kbytes に変化させて、FTP を用いて 100Mbytes のファイルを転送した時の RADIC-TCP と CUBIC-TCP のスループットを測定した。TCP のウィンドウサイズ(送受信バッファ)最大値を 16Mbytes とした時の結果を図 70 に示す。

CUBIC-TCP は、ウィンドウサイズに基づく送信量制御により、RTT 毎にバーストトラフィックを発生させるので(図 44 参照)、キュー長が小さいほどパケット廃棄を引き起こしやすい。パケット廃棄が発生すると、CUBIC-TCP のパケット廃棄発生に基づく輻輳制御により、送信帯域を過剰に減少させるため、回線帯域を使い切れなかった。一方、RADIC-TCP は、キュー長に依存せず 80～90Mbps の間で安定したスループットを実現した。

(8)スループットのウィンドウサイズへの依存性

スループットのウィンドウサイズ依存性を評価した。図 69 の評価構成において、エミュレータの帯域制御を OFF とし、200ms の RTT と 1%のパケット廃棄率を発生させた。FTP を用いて 1Gbytes のファイルを転送したときの RADIC-TCP と CUBIC-TCP のスループットを測定した。TCP のウィンドウサイズ(送受信バッファ)最大値を 16～256Mbytes に変化させたときの結果を図 71 に示す。

ウィンドウサイズを大きくすれば､TCP のスループットを増加可能と考えられがちだが、WAN やインターネットのように他のトラフィックとの競合によるパケット廃棄が発生する環境では、図 71 に示すように、ウィンドウサイズ最大値を大きくしても CUBIC-TCP のスループットは 0.6Mbps のまま大きくならない。一方、RADIC-TCP のスループットは、ウィンドウサイズ最大値を大きくすることで 130Mbps から 700Mbps まで増加した。WAN やインターネットのように他のトラフィックとの競合によるパケット廃棄が発生する環境における TCP 通信のスループットは、ウィンドウサイズだけではなく、輻輳制御のアルゴリズムに大きく依存する。

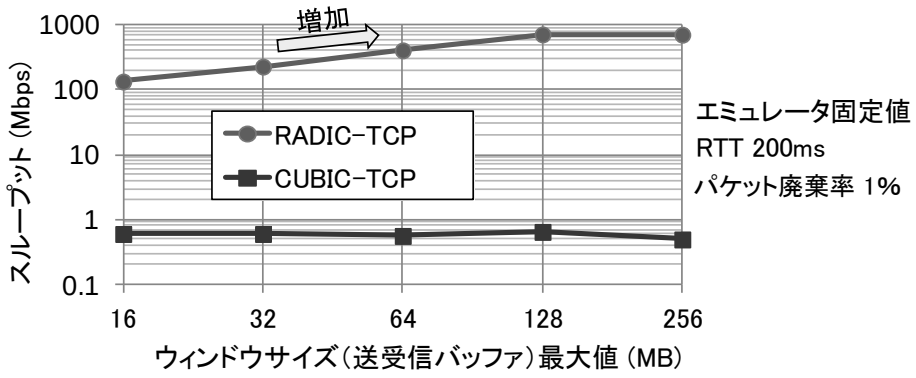


図 71　ウィンドウサイズとスループットの関係

(9)提案 TCP と標準 TCP の間の公平性

提案する RADIC-TCP は、WAN やインターネットを他の TCP と共に利用する場合、帯域を使い切りつつ標準的な TCP の帯域を保護することが重要である。RADIC-TCP と標準 TCP の間の公平性を評価するため、本研究では、図 55 の構成を用いて、RADIC-TCP と標準 TCP が 100Mbps 回線を共有する試験を行った。RTT を 2, 20, 200ms として、RADIC-TCP と標準 TCP のスループット推移を計測する評価を行った。標準 TCP として IETF 標準の TCP Reno を SACK と共に用いた。TCP Reno の送受信バッファサイズは、非力なモバイル端末やタブレット PC などを想定した 128K バイトとした。一方、RADIC-TCP の送受信バッファサイズは 16 M バイトとした。RADIC-TCP が、非輻輳発生時に送信帯域を線形増加させる時間は、帯域遅延積を 40*MSS で除算した値とした。

TCP Reno が FTP 上で 146MB のファイル転送を開始してから、RADIC-TCP が FTP 上で 146MB のファイル転送を開始したときのスループットを、RTT 2, 20, 200ms のそれぞれのケースについて図 72, 図 73, 図 74 に示した。

RTT が 2ms のときは、TCP Reno と RADIC-TCP の間でスループットを 55Mbps、45Mbps に分け合い、TCP Reno の最大帯域の方が大きかった。RTT が 20ms のときは、TCP Reno と RADIC-TCP の間でスループットを 52Mbps、48Mbps に分け合い、TCP Reno の最大帯域の方が大きかった。以上、RTT 20ms 以下では、RADIC-TCP が帯域を使い切りつつ標準的な TCP の帯域を保護できることを確認した。RTT が 200ms のときは、TCP Reno の最大帯域は、競合が発生していない時でも 12Mbps となり、回線帯域を使い切ることができなかった。RADIC-TCP が通信を開始して競合が発生し始めると、TCP Reno の最大帯域の減少は 12Mbps から 10Mbps への 17%に留まり、十分な帯域を確保して通信すること可能であった。RADIC-TCP は、残りの帯域を使い切った。提案する RADIC-TCP は、標準的な TCP の帯域への影響を最小限に抑えつつ、回線帯域を使い切ることを確認した。

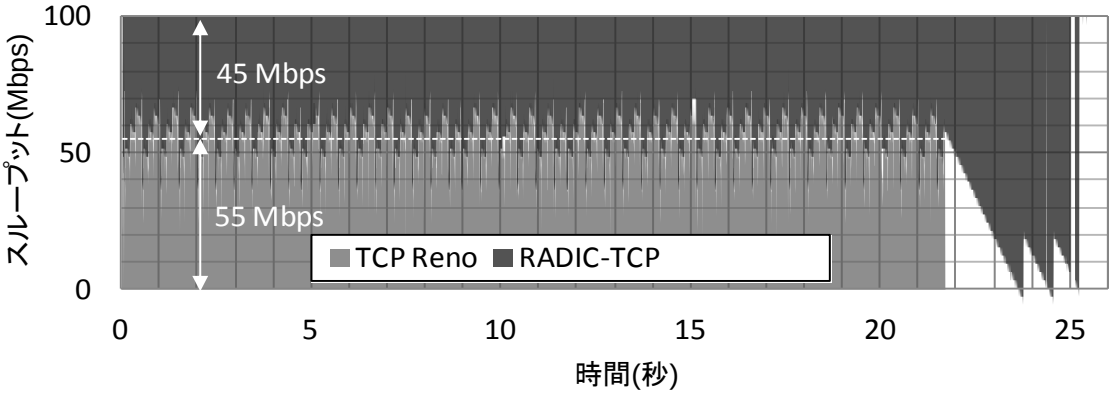


図 72　RADIC-TCP と TCP Reno の間の公平性(RTT 2ms)

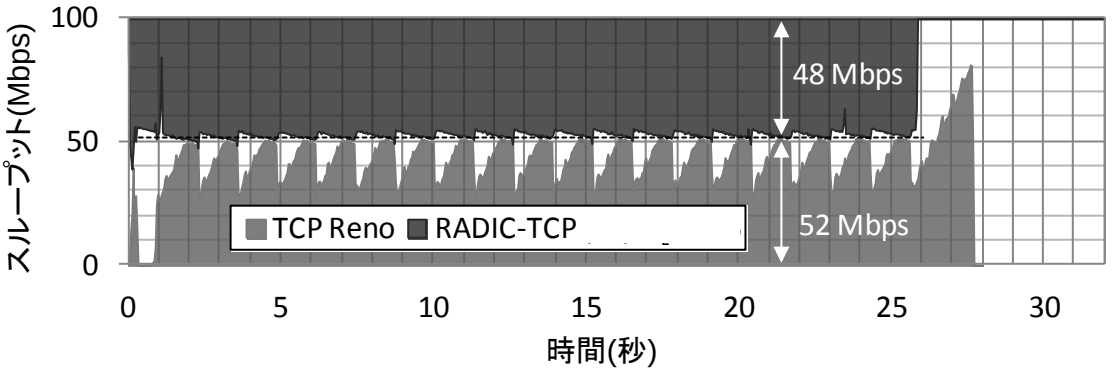


図 73　RADIC-TCP と TCP Reno の間の公平性(RTT 20ms)

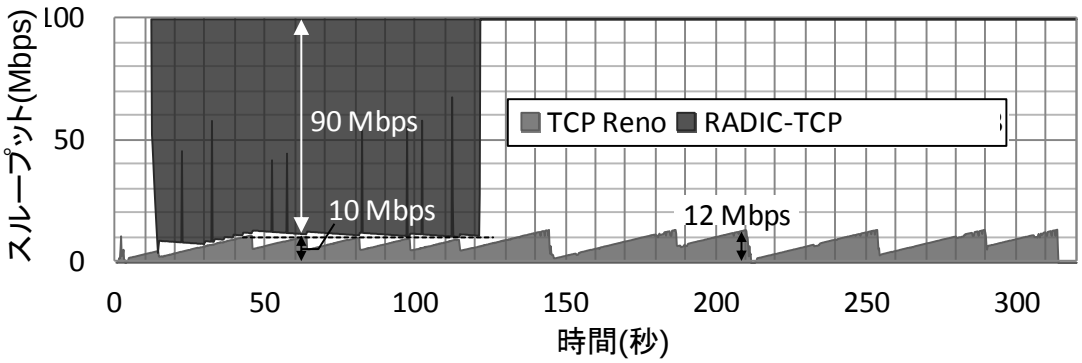


図 74　RADIC-TCP と TCP Reno の間の公平性(RTT 200ms)

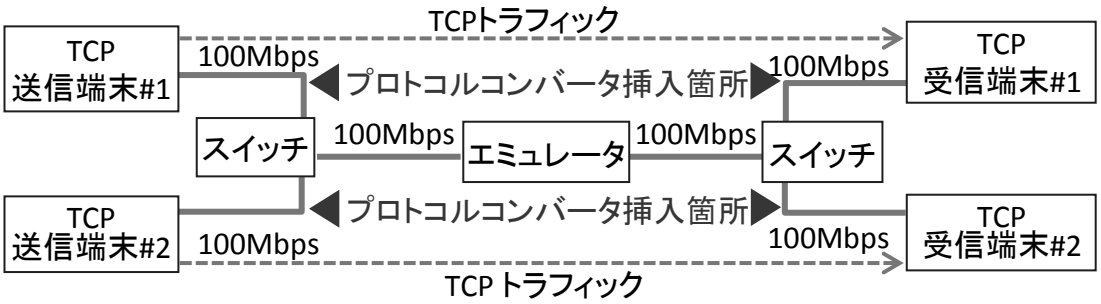


図 75　異なる RTT を持つ TCP セッション間の帯域分割の評価構成

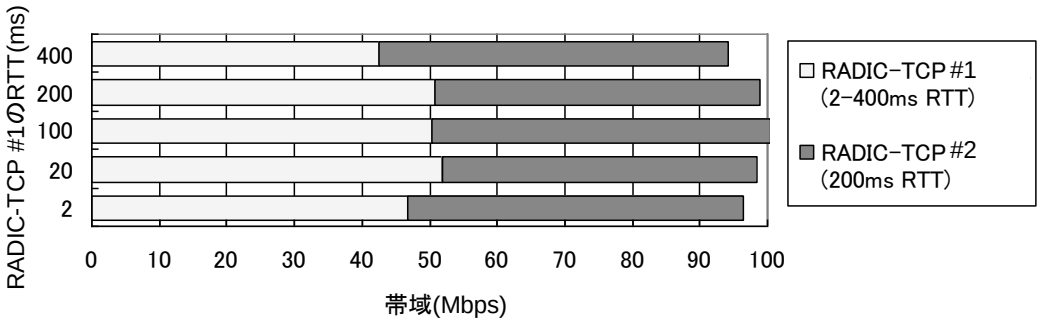


図 76　異なる RTT を持つ TCP セッション間の帯域分割

(10)異なる RTT を持つ通信間の公平性

複数の遠隔拠点との通信を同時に実施するユースケースを想定して、互いに異なる RTT を持つ 2 つの RADIC-TCP の通信コネクションが、100Mbps の回線帯域を共有する際の帯域分配を、図 75 の実験構成を用いて評価した。

本評価では、送信端末#1-2 と、受信端末#1-2 と、通信遅延とパケット廃棄を発生させるエミュレータ、スイッチを 100 Mbps の回線で接続する構成とした。各送受信端末とスイッチの間にプロトコルコンバータを挿入することで、送信端末#1 と受信端末#1 の間と、送信端末#2 と受信端末#2 の間に、RADIC-TCP の通信コネクション#1 と#2 を生成する。通信コネクション#2 の RTT を 200 ms に固定して、通信コネクション#1 の RTT を変化させた。

各通信コネクションにおいて、146 MB のファイルを FTP 転送した際の各コネクションの通信帯域を図 76 に示す。通信コネクション#1 の RTT が変化しても、各通信コネクションの帯域は回線帯域の 50%に近い 42Mbps 以上の帯域を確保している。帯域の小さな通信コネクションに優先的に帯域を分配しつつ、RTT に大きく依存しない帯域分配を実現したと考えられる。提案する RADIC-TCP は、帯域利用率の向上に加え、通信コネクション間の帯域の公平な分配を実現できたと考えられる。

(11)多数通信コネクション間の帯域割当

複数の RADIC-TCP の通信コネクションが利用する帯域を、予め定めた上限帯域以下に制

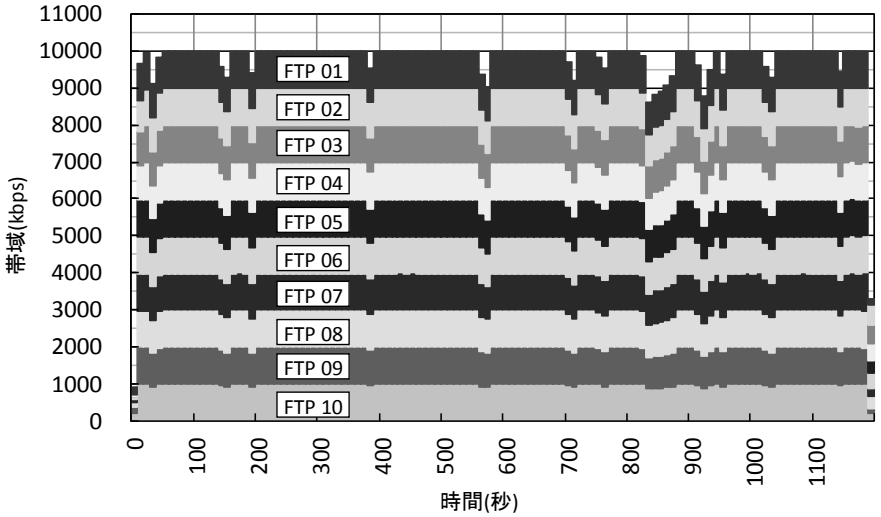


図 77　10 本の FTP コネクションで 10Mbps を共有した場合の帯域分割

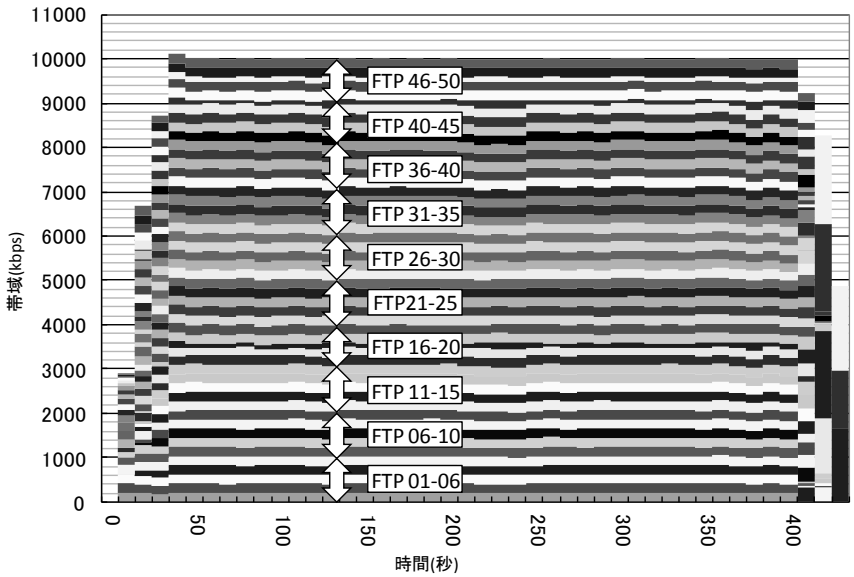


図 78　50 本の FTP コネクションで 10Mbps を共有した場合の帯域分割

御しつつ公平に帯域を分配することも可能である。図 55 の構成を用いて、予め定めた上限帯域を複数の RADIC-TCP の通信コネクションが公平に分割できることを評価した。プロトコルコンバータで設定する上限帯域を 10Mbps として、エミュレータで 200ms の RTT を発生させて、RADIC-TCP を用いる端末間で 10 本、または 50 本の FTP コネクションを生成して、それぞれの FTP コネクションでファイルを転送する試験を行った。

10 本の FTP コネクションを生成して、各コネクションで 146MB のファイルを同時に転送した際の帯域分割の時間推移を図 77 に、50 本の FTP コネクションを生成して、各コネクションで 10MB のファイルを同時に転送した際の帯域分割の時間推移を図 78 に示した。

10 本の FTP コネクションで 10Mbps をシェアする場合は 1 コネクションあたり 1Mbps、

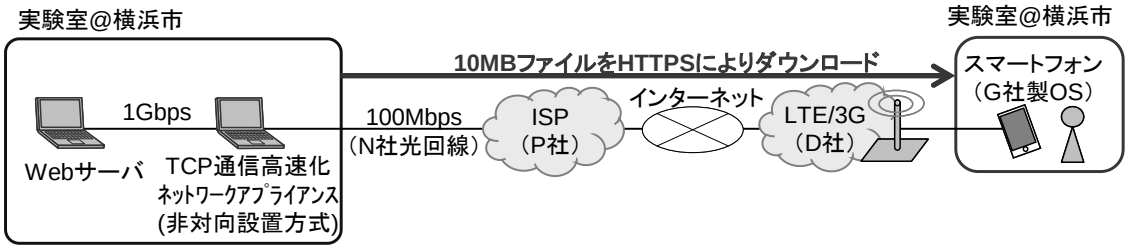


図 79　LTE 網または 3G 網を用いた評価構成

表 10　実験緒元

機材	詳細
Web サーバ	CPU：動作周波数 2.7-3.4GHz, キャッシュ 4MB [158] メモリ：DDR3-1333MHz 8G バイト(デュアルチャネル) OS：パソコン用の商用 OS [159]
TCP 通信高速化 ネットワークアプライアンス (非対向設置方式)	CPU：動作周波数 2.7-3.4GHz, キャッシュ 4MB [158] メモリ：DDR3-1333MHz 8G バイト(デュアルチャネル) OS：オープンソースの OS [160]
スマートフォン	CPU：汎用プロセッサ OS：スマートフォン OS(G 社製[161]または A 社製[162])

50 本でシェアする場合は 1 コネクションあたり 200kbps の帯域が公平に割り当てられた。同時に、全コネクションの合計帯域は、上限帯域の 10Mbps 以内に制限されることを確認した。

4.6.2　実際の無線ネットワークとスマートフォンを用いた評価

提案する RADIC-TCP が、実験環境だけでなく実際のネットワーク環境でも、期待通りに動作することを検証する必要がある。実際のネットワーク環境は、パケット廃棄の発生間隔や、通信遅延の揺らぎや、端末の送受信バッファなどの点で、実験環境と大きく異なると考えられる。そこで、本研究では、パケット廃棄の発生間隔や通信遅延の揺らぎが、有線ネットワークよりも大きく変動すると考えられる無線ネットワークを用いて、通常の PC 端末よりも小さい送受信バッファを持つスマートフォン向けにデータを転送する試験を実施した。本試験により、非対向設置による SACK 併用 RADIC-TCP の実際のネットワーク環境における高速化効果を評価した。無線ネットワークの種類(LTE, 3G, WiFi など)、通信キャリアおよび端末、ネットワーク経路、ネットワークの混雑度など、様々な要素を変化させて高速化効果を評価した。

(1) LTE/3G 回線を用いた評価

神奈川県横浜市の実験室において、通信キャリア D 社[157]の商用 LTE・3G 回線を用いて、スマートフォン端末向けのデータ転送実験を行った。評価構成と実験緒元を図 79 および表 10 に示す。

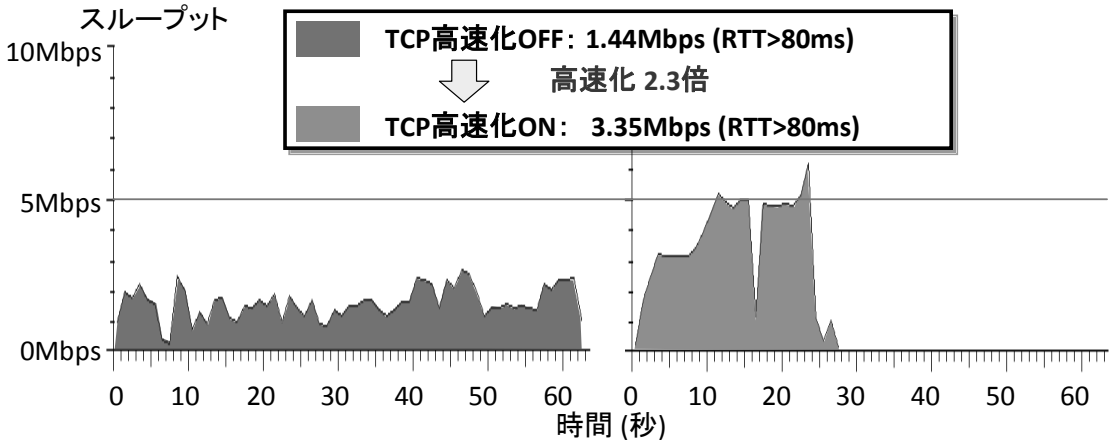


図 80　3G 回線における TCP 通信高速化効果

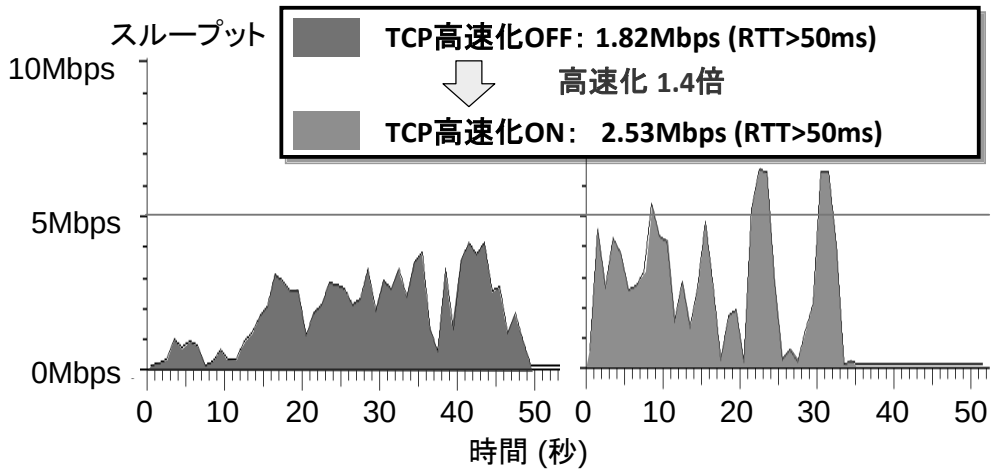


図 81　 LTE 回線における TCP 通信高速化効果

実験室内に、データ送信用の汎用プロセッサ[158]と商用 OS[159]を搭載した Web サーバと、非対向設置による SACK 併用 RADIC-TCP を用いた TCP 通信高速化ネットワークアプライアンスを設置し、1Gbps でリンクアップさせた。RADIC-TCP が、非輻輳発生時に送信帯域を線形増加させる時間は、帯域遅延積を 40*MSS で除算した値とした。TCP 通信高速化ネットワークアプライアンスの先は 100Mbps の商用光回線(N 社[163])を利用して商用 ISP(P 社[164])ネットワークに繋がり、インターネットに接続する。実験室内において、G 社製 OS[161]を搭載したスマートフォンから Web サーバの 10MB データに HTTPS を用いてアクセスを行い、データ転送が完了するまでの平均スループットを測定した。Web サーバは、TCP 通信高速化機能が OFF の場合、Compound TCP[43]を用いて通信する。

通信キャリア D 社の 3G 回線および LTE 回線を使用して、TCP 通信高速化機能を ON・OFF しながら、スマートフォン向けデータ転送を実施した時のスループットの推移を、3G 回線および LTE 回線のそれぞれについて図 80 と図 81 に示した。

TCP 通信高速化を ON することにより、3G 回線ではスループットが 1.44Mbps から

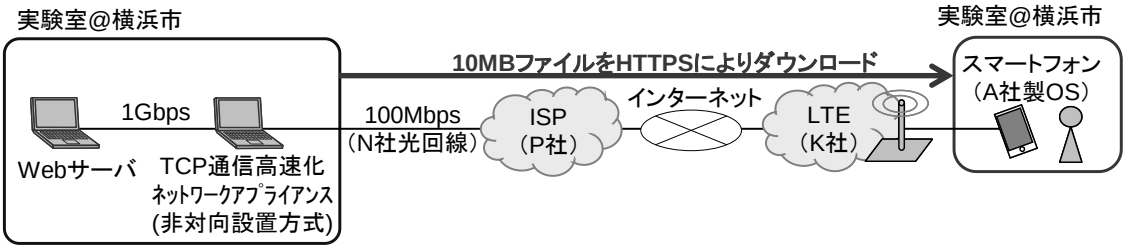


図 82　通信キャリアおよびスマートフォン端末を変更した評価構成

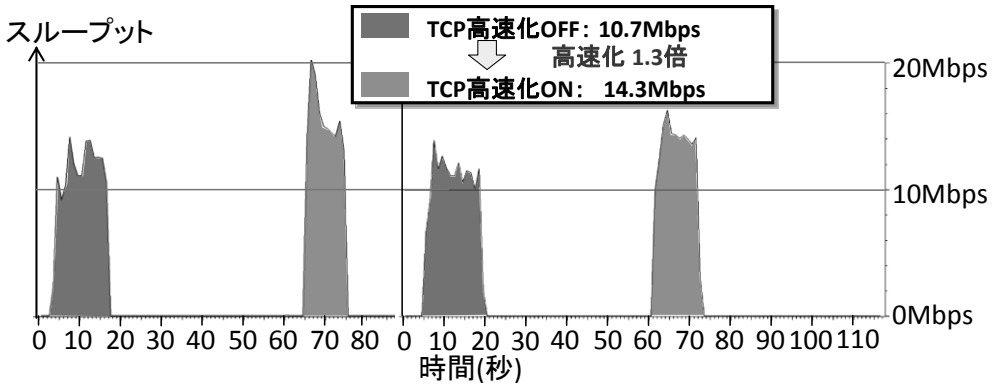


図 83　通信キャリアとスマートフォン端末を変更した時の TCP 通信高速化効果

3.35Mbps へと 2.3 倍に高速化、LTE 回線ではスループットが 1.82Mbps から 2.53Mbps へと 1.4 倍に高速化することを確認した。回線種別に依存せず、高速化効果が得られた。

新規格 LTE の RTT は 50ms と、旧規格 3G の 80ms と比較して短かったが、新規格 LTE の最大スループットは 2.53Mbps と、旧規格 3G の 3.35Mbps と比較して小さかった。最大スループットは、回線規格だけでなく、その地域における通信キャリアの設備投資状況、や利用ユーザ数や、利用方法に依存して大きく変化すると推定される。

(2)他の通信キャリアや端末を用いた評価

無線回線の通信キャリアを D 社[157]から K 社[165]に、スマートフォンを G 社製 OS[161]搭載端末から A 社製 OS[162]搭載端末に変更して、(1)と同様の方法を用いて RADIC-TCP の高速化効果の評価を実施した。評価構成を図 82 に示す。

通信キャリア K 社[165]の LTE 回線を使用して、TCP 通信高速化の機能を交互に ON・OFF しながら、スマートフォン向けデータ転送を 2 回繰り返した時のスループットの推移を図 83 に示す。RADIC-TCP が、非輻輳発生時に送信帯域を線形増加させる時間は、帯域遅延積を 40*MSS で除算した値とした。

通信キャリア K 社[165]の LTE 回線、および G 社製 OS 搭載スマートフォンを用いても、TCP 通信高速化を入れることにより、スループットが 10.4～10.9Mbps から 13.5～15.0Mbps へ 1.3～1.4 倍に高速化することを確認した。非対向設置方式による TCP 通信高速化機能は、キャリアや端末 OS に依存しない高速化が可能であった。

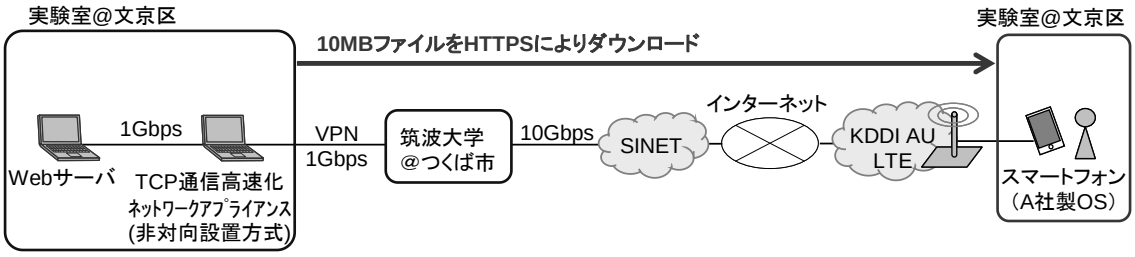


図 84　経由するネットワークを SINET へ変更した評価構成

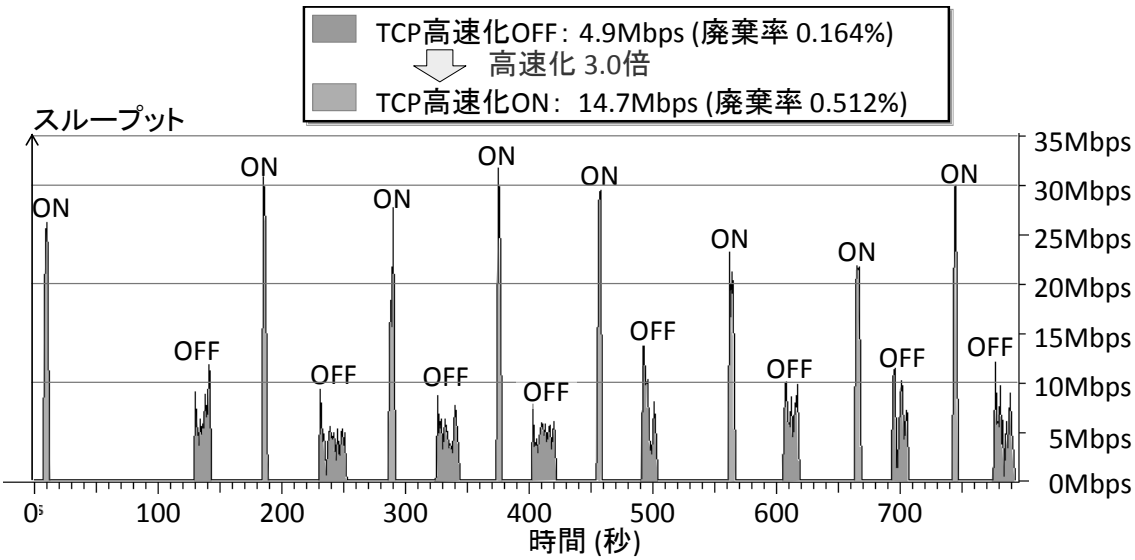


図 85　経由するネットワークを SINET へ変更した時の TCP 通信高速化効果

(3)SINET を用いた評価

Web サーバや、TCP 通信高速化を行うネットワークアプライアンスの設置場所と、スマートフォンからサーバにアクセスする場所を、横浜市の実験室から、文京区の実験室に移して、同様の評価を実施した。評価構成を図 84 に示す。

文京区の実験室に、データ送信用の Web サーバと、非対向設置方式の TCP 通信高速化を行うネットワークアプライアンスを設置し、1Gbps でリンクアップさせた。ネットワークアプライアンスの先は 1Gbps の VPN 回線を経由して、つくば市の筑波大学キャンパス LAN を通り、10Gpbs 回線で学術情報ネットワークである Science Information Network (SINET) [166]に接続している。その先は、SINET 経由でインターネットに接続する。文京区の実験室において、G 社製 OS[161]搭載スマートフォン端末から Web サーバの 10MB データに HTTPS を用いてアクセスを行い、データ転送が完了するまでの平均スループットを測定した。

キャリア K 社[165]の LTE 回線を使用して、TCP 通信高速化の機能を交互に ON・OFF しながら、サーバからスマートフォン向けのデータ転送を 8 回繰り返した時のスループットの推移を図 85 に示す。

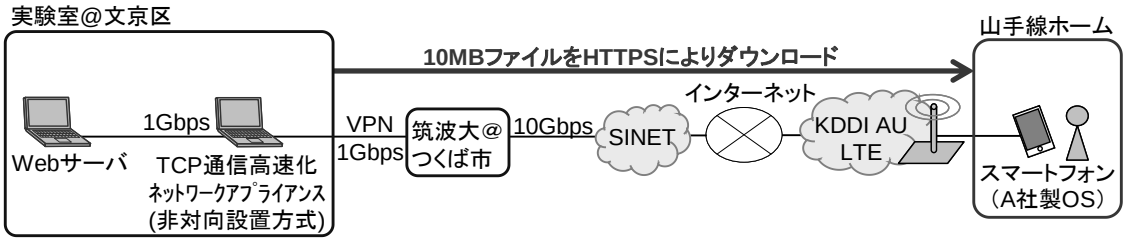


図 86　無線回線の混雑度を変更した評価構成

Web サーバや、TCP 通信高速化を行うネットワークアプライアンスの設置場所や、スマートフォンからサーバにアクセスする場所を、横浜市の実験室内から文京区の実験室内に移して、経由するネットワークを商用 ISP から SINET に変更しても、TCP 通信高速化の機能を ON することにより、スループットが 4.9Mbps から 14.7Mbps へ 3.0 倍に高速化することを確認した。非対向設置方式による TCP 通信高速化機能は、経由するネットワークに依存しない高速化が可能であった。

通信中のパケット廃棄率は､TCP 通信高速化機能を ON にすることで､0.164%から 0.512%に上昇した。これは、スループットが 3 倍に増加することで、他のトラフィックと競合しやすくなったためと考えられる。パケット廃棄率の増加分は 0.348%と、廃棄帯域の増加分は 51kbps 程度であり、最大スループット 32Mbps の 0.16%を占めるに過ぎず、無線ネットワークに過度の輻輳を引き起こしていないと考えられる。

同一条件の実験環境における高速化効果は、TCP 通信高速化 OFF，RTT 40ms，loss 0.16%の時の 6.9Mbps に対して、TCP 通信高速化 ON，RTT 40ms，loss 0.51%の時で 23Mbps と約 3 倍であり、実網評価と実験評価で同様の高速化効果が得られた。

(4)混雑した無線ネットワークにおける評価

スマートフォンからサーバにアクセスする場所を、文京区の実験室内から、東京駅、品川駅、渋谷駅、新宿駅の山手線ホームへと無線回線がより混雑する場所へと移し、同様の評価を実施した。評価構成を図 86 に示す。評価は、混雑度が多いと考えられる時間帯等で実施した。

東京駅，品川駅，渋谷駅，新宿駅の山手線ホームにおいて、キャリア K 社の LTE 回線[165]を使用して、TCP 通信高速化機能を交互に ON・OFF しながら、スマートフォン向けデータ転送を複数回繰り返した時のスループットの推移を図 87、図 88、図 89、図 90 に示す。東京駅と渋谷駅では 2 回に分けて評価を実施し､新宿駅では 3 回に分けて評価を実施した。

スマートフォンからサーバにアクセスする場所を、無線回線の混雑した場所に変更しても、TCP 通信高速化の機能を ON することにより、スループットが 1.2～2.3 倍に高速化することを確認した。非対向設置方式による TCP 通信高速化機能は、無線回線の混雑度に依存しない高速化が可能であった。

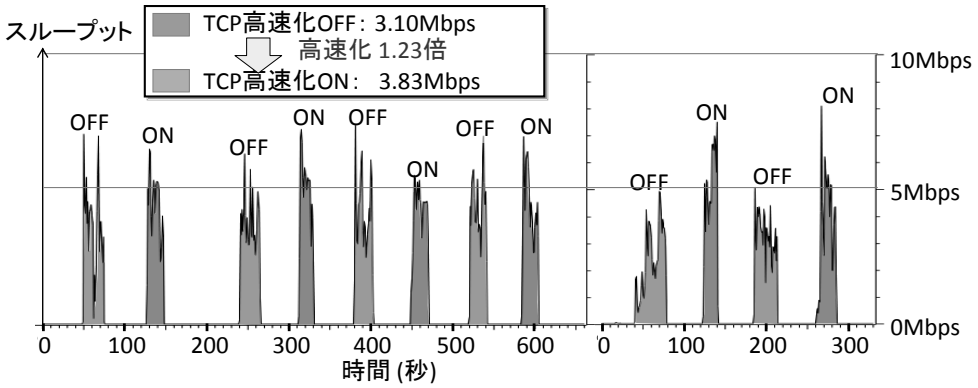


図 **87**　東京駅からアクセスした時の **TCP** 通信高速化効果

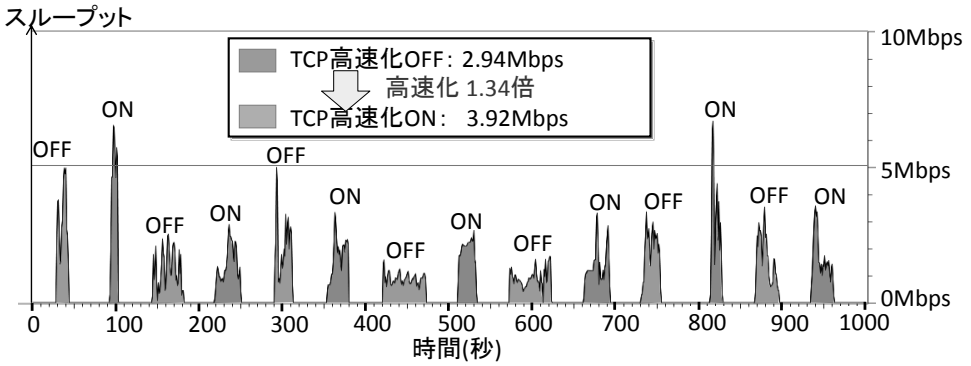


図 **88**　品川駅からアクセスした時の **TCP** 通信高速化効果

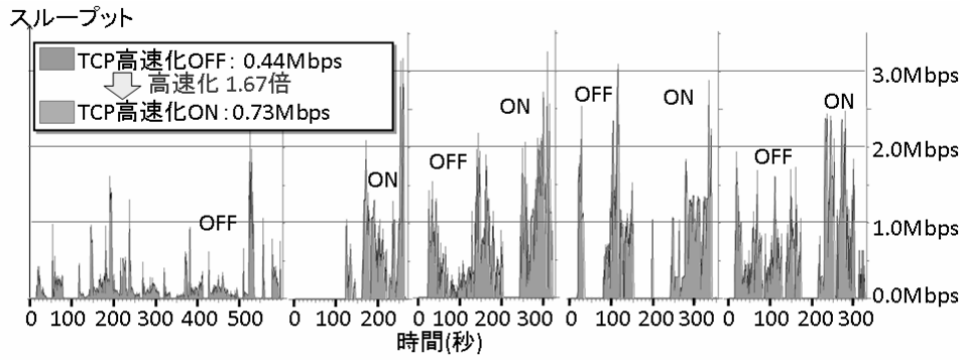


図 **89**　渋谷駅からアクセスした時の **TCP** 通信高速化効果

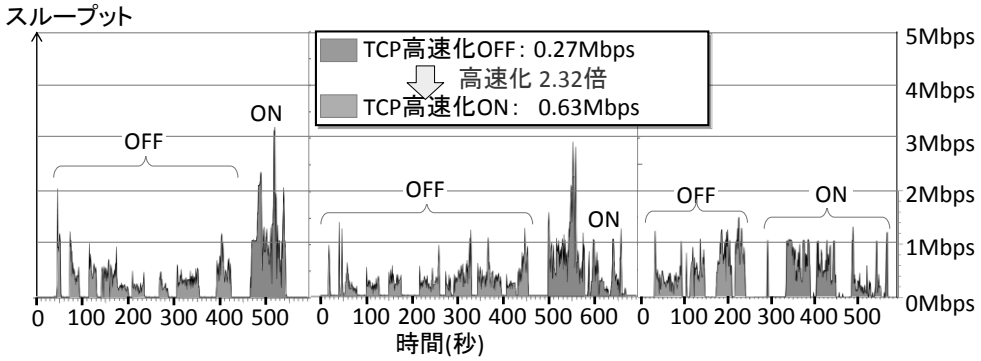


図 **90**　新宿駅からアクセスした時の **TCP** 通信高速化効果

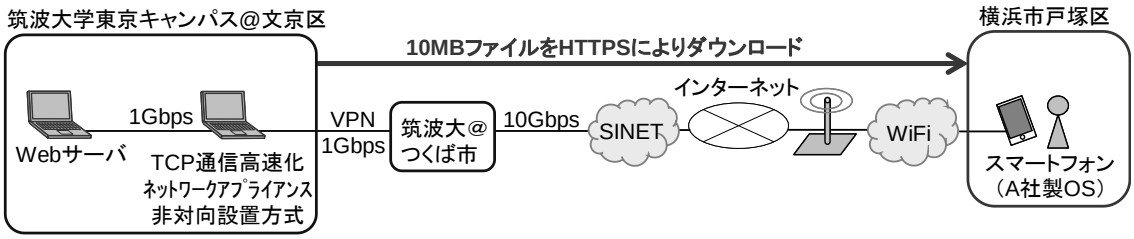


図 91　Wi-Fi ネットワークを用いた評価構成

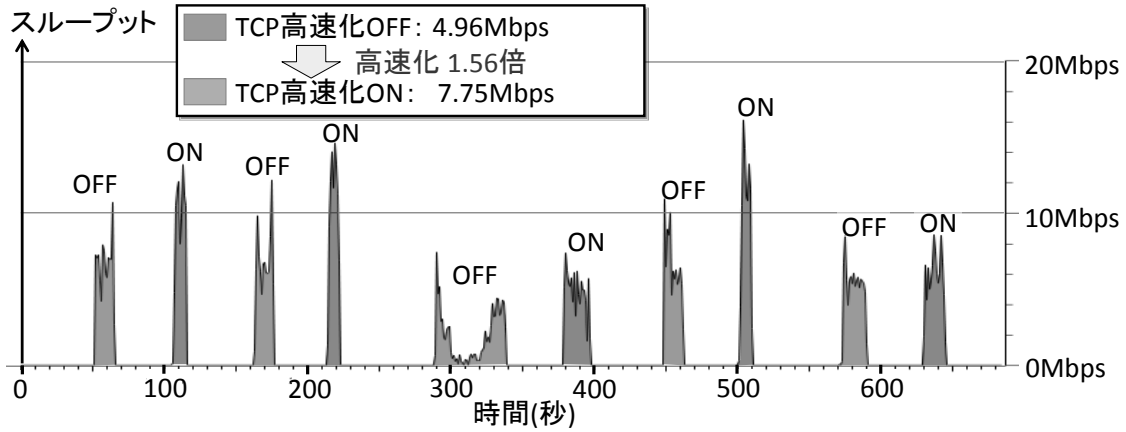


図 92　WiFi 回線を利用してアクセスした時の TCP 通信高速化効果

なお、渋谷駅や新宿駅では、スループットが 0Mbps に低下している時間帯が見られた。これは、渋谷駅や新宿駅の週末の混雑度が東京駅や品川駅よりも高く、基地局のユーザ収容数上限値を超過したことにより通信できない時間が発生したためだと推定される。

(5)WiFi ネットワークを用いた評価

スマートフォンからデータアクセスするために利用する無線回線を、基地局エリアの大きな LTE 回線から、基地局エリアの小さい WiFi 回線に変更して、同様の評価を実施した。評価構成を図 91 に示す。WiFi 回線として、物理的な最大帯域が 54Mbps である 802.11g を用いた。

横浜市において、WiFi 回線を使用して、TCP 通信高速化の機能を交互に ON・OFF しながら、サーバからスマートフォン向けデータ転送を 5 回繰り返した時のスループットの推移を図 92 に示す。

スマートフォンからサーバにアクセスする際に利用する無線回線を、LTE から WiFi に変更しても、TCP 通信高速化の機能を ON することにより、スループットが 1.56 倍に高速化することを確認した。非対向設置方式による TCP 通信高速化機能は、WiFi でも高速化が可能であった。

4.7 結言

本研究では、通信遅延が大きく、パケット廃棄率の大きいネットワークにおいて、TCP のスループットが劣化する問題を解決するために、以下 3 つの特徴を持つ TCP 通信高速化技術 RADIC-TCP を提案した。

① パケット廃棄率の変化率に基づく輻輳制御
② トークンサイズを用いた送信量制御
③ NACK を用いた再送制御

RADIC-TCP は、パケット廃棄率の変化率を用いてネットワークの輻輳を検出して、送信帯域を残存帯域に追従するよう動的に制御することで、一時的なパケット廃棄の影響を抑えて WAN やインターネットの帯域を有効活用する。トークンサイズに応じた数のパケットを一定間隔で送信することで、パケット廃棄の発生を抑制しつつ、RTT の影響を緩和する。廃棄箇所を特定し易い NACK を用いることで、パケット廃棄率が高い時のパケット再送の演算コストを削減する。

更に、パブリッククラウドを利用する際の、クラウド利用ユーザの端末向けの TCP 通信を高速化するために、RADIC-TCP と SACK (Selective ACKnowledgement)[130]を併用することで、データセンタ側に設置するだけで、モバイル端末をはじめとする多様な端末向けの通信を高速化可能な非対向設置による SACK 併用 RADIC-TCP を提案した。

非対向設置による SACK 併用 RADIC-TCP は、NACK を用いた再送制御の代わりに、通常の SACK を用いた再送制御を用いることで、提案した RADIC-TCP と同様の輻輳制御アルゴリズム[39][129] [151]をサポートしつつ、送信側に設置するだけで、受信側拠点への装置設置や、受信端末へのソフトウェアのインストールを必要とせずに、不特定多数の端末への送信方向の TCP 通信を高速化可能である。不特定多数のユーザに様々な ICT サービスを提供するクラウド事業者は、非対向設置方式による TCP 通信高速化を用いることで、より快適なサービスを提供することが可能となる。

実験評価では、標準 TCP を RADIC-TCP へ変換するプロトコルコンバータを用いて、RADIC-TCP のスループットを評価した。残存帯域が 1 秒毎に変化する環境において、RADIC-TCP が残存帯域を使い切ることを示した。更に、ボトルネックのキュー長に依存しないスループットを実現し、RTT やパケット廃棄率が大きな環境(RTT 200ms, パケット廃棄率 0.01%以上)でも、従来 TCP 比 10 倍のスループットを実現した。パケット廃棄が発生する環境でウィンドウサイズを増加させても、従来 TCP のスループットは向上しないのに対し、提案 TCP のスループットは向上した。提案 TCP は、FTP だけでなく CIFS のスループットも向上させた。更に、上限帯域が 10Mbps のときに、複数のコネクション間で公平に帯域を分割して利用することを確認した。

更に、実際の無線ネットワーク回線を用いて、パケット廃棄の発生間隔や通信遅延の揺

らぎ方が実験環境と異なる実ネットワーク環境での高速化効果を評価した。非対向設置による SACK 併用 RADIC-TCP を用いた TCP 通信高速化ネットワークアプライアンスの試作機を用いて、実際の無線ネットワークを経由して、性能に制約があるスマートフォン向けデータ転送実験を行い、高速化効果を実証した。RTT 40ms の LTE 回線を用いた評価では、実験環境同様の約 3 倍の高速化を確認した。更に、無線回線規格，キャリアおよびスマートフォン端末，経由するネットワーク，無線回線の混雑度などを変化させて評価したところ、1.2〜3.0 倍の高速化効果を確認した。実験環境と同様に高速化効果が得られることを確認した。

以上により、クラウドデータセンタへの非対向設置形態も取り得る、多様な端末に対応する汎用性の高い TCP 通信高速化方式を用いることで、ユーザが利用する多様なアプリケーションと端末に対応する汎用性の高い通信高速化を実現できることを示した。

第５章　クラウドサービス遠隔利用時の応答高速化

5.1　緒言

近年、通信ネットワークを介したクラウドデータセンタと多数のセンサデバイス(温度・位置情報取得装置、設備監視装置、監視カメラ、車載端末、等)の間の M2M (Machine to Machine)リアルタイム通信や、クラウドデータセンタと遠隔端末の間の画像・映像などの大容量データ通信が増加しつつある。それに伴い、クラウド事業者とクラウド利用ユーザの双方から、クラウドデータセンタとセンサデバイス／遠隔端末の間の通信データを高速に捌いて、クラウドサービスの応答速度を高速化することが求められている。特に、センサデバイスからのリアルタイムな情報収集と解析を行い、解析結果に基づいて多数のセンサデバイスに制御データを配信するクラウドサービスにおいて、応答高速化へのニーズが高まりつつある。

ところが、遠隔地のセンサデバイスや端末からクラウドにアクセスする際の往復通信時間が長く、遠隔地からコマンド要求を送信してクラウドデータセンタからリプライ応答が帰ってくるまでのコマンド往復時間が長い場合に、クラウドサービスの応答時間が遅くなる課題があった。

そこで、本章では、図 93 に示すように、クラウドサービスを肩代わりするネットワークアプライアンスを、ネットワークエッジに分散配置して、クラウドサービスの応答時間を高速化することを提案する。

提案するネットワークアプライアンスは、センサデバイス近くのネットワークエッジに設置するユースケースが想定されるため、設置場所を選ばないよう小型省電力な装置として実現する必要がある。また、多数センサデバイスからの大量の通信データの収集・解析を可能とする高い演算性能が求められる。更に、クラウドサービスをアプリレベルで代替する性格上、クラウドサービスを代行して応答を高速化するクラウドサービス高速化系の機能(1) (2.5 節)だけでなく、公衆網からの攻撃トラフィックを排除するセキュリティ系の機能(2) (2.3 節)も備える必要がある。クラウドサービス高速化系の機能(1)と、セキュリティ系の機能(2)としては、第２章に記述した通り、以下の機能を備える必要がある。

(1)クラウドサービス高速化系の機能
- キャッシュ系：クエリトランザクション、キャッシュ、VOD
- オフロード系：SSL / IPSec，HTML / XML テキスト解析・変換
- 負荷分散系　：ロードバランサ

(2)セキュリティ系の機能
- 統計情報系：対異常通信防御(IDS, IPS)、DPI
- パターン系：Firewall，NAT / NAPT

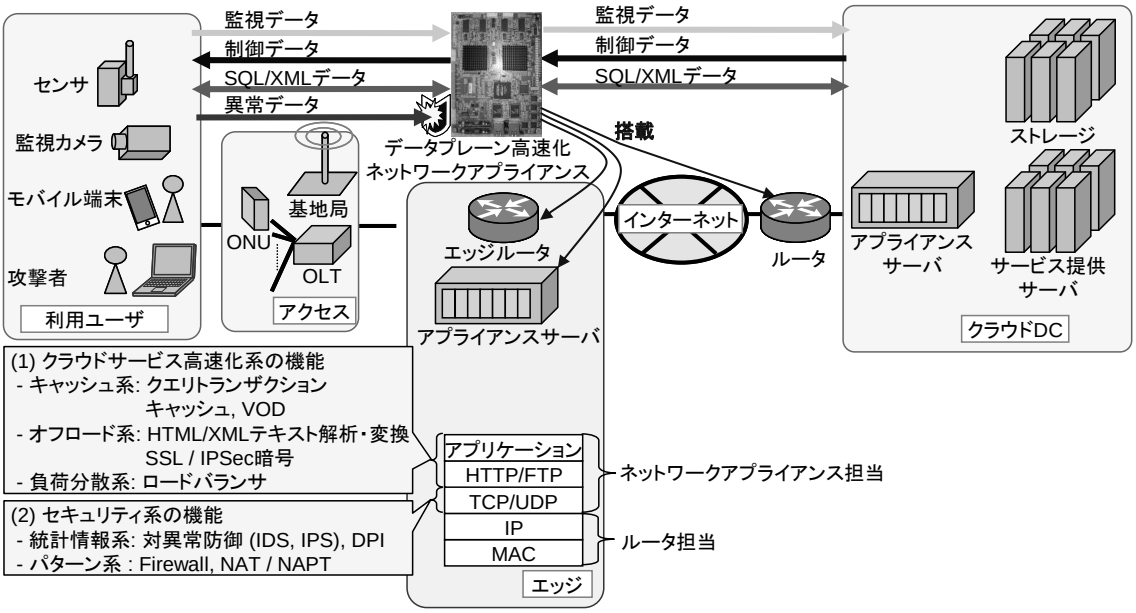


図 93　ネットワークアプライアンスを用いたクラウドサービス高速化

上記の要件を満たすネットワークアプライアンスには、小型省電力にてパケットを高速演算する性能と共に、ユーザやセンサデバイス毎に提供する機能(1)(2)を高速応答にて切替える柔軟性が要求される。

小型省電力での高速演算性能と、高速応答での多機能切替を両立するに当たり、従来の汎用プロセッサや ASIC (Application Specific Integrated Circuits)はいずれか一方を欠くという問題があった。その一方で、ネットワークプロセッサ(Network Processor: NP)、FPGA (Field Programmable Gate Array)、動的再構成 LSI [112][113][114][115]は、以下に示すように、小型省電力による高い演算性能と、機能を切り替えられる柔軟性を持つ。

a) ネットワークプロセッサ

ネットワークプロセッサ [167][168][169]は、異常通信を防止する機能(2)や、トランスポート層のプロトコルを制御するネットワークアプライアンスとして用いられている[118][170]。ネットワークプロセッサは、マイクロエンジン(Micro Engine (ME))が実行中の命令キャッシュを変更することで機能の切り替えを実現する。機能の切り替えには、外部メモリを用いて複数の ME を同期させる複雑なプログラミングを要するため時間を要し、機能(1)(2) を高速応答にて切替える柔軟性が要求されるネットワークアプライアンス用途に向かない。加えて、ネットワークプロセッサの性能(24 GOPS) は動的再構成 LSI の性能(28 GOPS)よりも低く、ネットワークプロセッサの電力消費量(30W)は動的再構成 LSI の電力消費量(7W)よりも大きいので[114][115][168]、小型省電力での高速演算性能が要求されるネットワークアプライアンス用途に向かない。

b) FPGA

FPGA は、頻繁に機能を再構成することを前提とした設計になっておらず、プログラムを変更しようとすると、動的再構成 LSI の変更時間(1 クロックサイクル)よりも長い時間を必要とする。再構成用のサポートツールも不十分との報告がある[171]。

c) 動的再構成 LSI

動的再構成LSIは、多数の演算エレメント(PE: Processing Element)を含む演算マトリックスを備える。演算エレメントは、例えば、実行エレメント(EXE)、遅延エレメント(DEL)、メモリエレメント(ME)などの種類がある[114][115]。これらのエレメントは、セレクタ経由で互いに接続しており、低い動作周波数と消費電力でも、並列演算を用いることでASICに迫る演算速度を実現する。省電力で高い演算性能を実現可能なので、発熱が少なく小型化が容易である。また、動的再構成LSI は、実行エレメントの役割や演算エレメント間の接続を1クロックサイクルで変更することで、搭載論理を短時間で変更できる高い柔軟性を実現する。これらの、小型省電力での高速演算性能と、高速応答での機能切替を使いこなすことで、機能(1)(2)を実現するネットワークアプライアンスの報告がある[36][72][172]。

上記a)～c)により、動的再構成LSIは、小型省電力でのパケット高速演算性能と、機能(1)(2)を高速応答に切替える柔軟性と、を同時に要求するネットワークアプライアンス向けプロセッサとして有望と考えられる。その一方で、アーキテクチャ次第で、得られるプロセッサ性能が異なる、という問題点があった。

そこで本章では、下記 3 方式を用いたアーキテクチャを持つ動的再構成 LSI 搭載ネットワークアプライアンス装置を提案する。

i. メモリ非経由ダイレクトパケット I/O
ii. プロセッサの階層接続
iii. 通信状態に基づくパケット毎自己再構成

これら 3 つの方式を用いたアーキテクチャを備えることで、小型省電力での高速演算性能(方式 i-iii 利用)と、高速応答での多機能切替(方式 iii 利用)を実現することを目指した。更に、上記 3 方式を用いたネットワークアプライアンスを試作して、セキュリティ機能(2)の対異常通信防御の一部(IPS 機能(3.2.2 参照))と、クラウドサービス高速化機能(1)の SQL を用いた DB 向けクエリトランザクションの一部と、クラウド DC のデータのキャッシュの一部を実装して、実験評価を行った。実験評価では、HTTP 上の SQL を用いて DB 向けクエリトランザクションを継続的に実行する評価を行った。1 秒間あたりに実行可能なトランザクション数や機能切替頻度、周波数から見積持った演算性能を評価した。

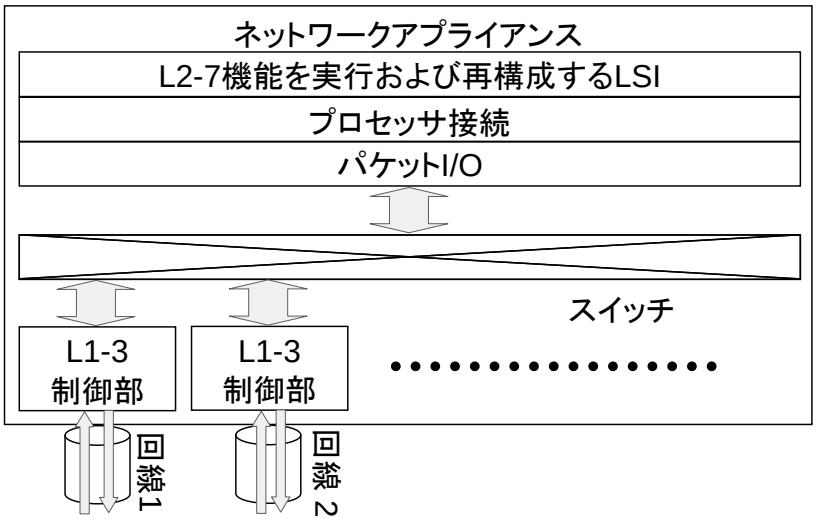


図 94　ネットワークアプライアンスのアーキテクチャ

5.2　クラウドサービス高速化ネットアークアプライアンスの概要

本研究で対象とするネットワークアプライアンスは、パケット交換用スイッチと、パケットルーティング用L1-3制御部に加えて、パケットI/O、プロセッサ接続、L2-7機能を実行、および再構成するプロセッサの3つの要素から成る(図 94)。

①パケットI/O(Input/Output)

スイッチとプロセッサ間でパケットを交換する。パケットバッファを持ち、スイッチまたはプロセッサからのパケット読込/書込のタイミングを調整する。TCP/UDPコネクションの状態管理を行う

②プロセッサ接続

スイッチ，パケットI/O，プロセッサ間のパケット経路を決定する。

③L(layer)2-7機能を実行および再構成するプロセッサ

5.1節に記載の(1)(2)から成る多様なネットワークアプライアンス機能を再構成および実行する。通信データの内容に応じて通信コネクション毎に異なる処理を行う。

5.3 節では、従来のアーキテクチャの 3 つの要素において用いる方式について説明する。5.4 節では、提案するアーキテクチャの 3 つの要素において用いる方式(5.1 節の i, ii, iii)について詳細に説明する。

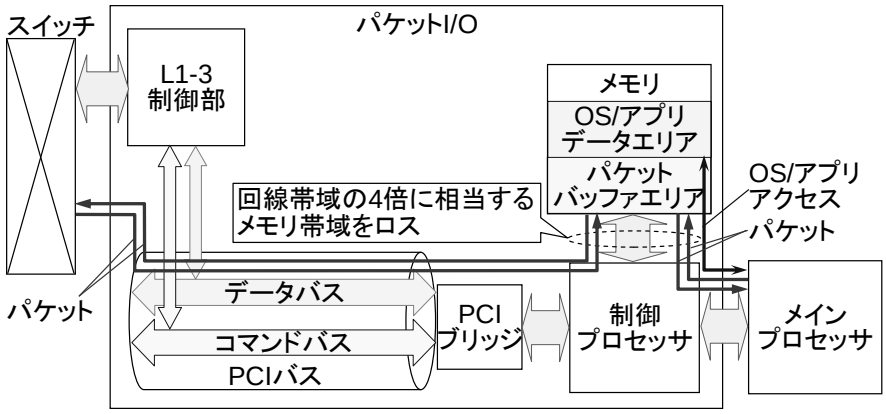


図 95　メモリ経由のパケット I/O

5.3　従来のクラウドサービス高速化技術

5.3.1　外付けメモリ経由のパケット I/O

一般に、汎用CPUと通信回線の間のパケットデータ入出力は、プロセッサで実行するプログラムの処理実行タイミングと、パケット入出力のタイミングを揃えるために、外部メモリを経由して行うプログラムが多い(図 95)。動的再構成LSIを用いるネットワークアプライアンスにおいても、外部メモリ経由のパケットI/Oを持つ方式の報告がある[72]。

パケットルーティング用L1-3制御部に到着したパケットは、メモリのパケットバッファエリアに転送され、最後にプロセッサが読み出す。更に、プロセッサが生成した新たなパケットは、メモリのパケットバッファ領域に蓄積され、パケットルーティング用L1-3制御部へと転送される。そのため、本方式は、図 95に示すように、パケットの入出力にて、入力パケット帯域の4倍に相当するメモリ帯域を消費する。

パケットI/Oと、プロセッサからのアプリケーションアクセスの両方がメモリ帯域をシェアする。そのため、本方式には、メモリ帯域の空時間を待つためのCPUアイドル時間が増大し、演算性能が低下するという問題があった。

5.3.2　プロセッサの直列接続

従来の動的再構成LSI搭載装置[36]は、プロセッサを直列に接続する方式を用いていた(図 96)。

パケットI/Oが、内部のパケットバッファに全てのパケットを蓄積して、動的再構成LSIへと転送する。動的再構成LSIが生成した新規データは、隣のパケットI/Oへと移動する。

本方式をネットワークアプライアンス向けに適用した場合、動的再構成LSIで実行する機能がパケット毎に異なるにもかかわらず、全パケットが全ての動的再構成LSIを通過する。そのため、処理を必要とするパケットが動的再構成LSIを通過している間、通過中のパケッ

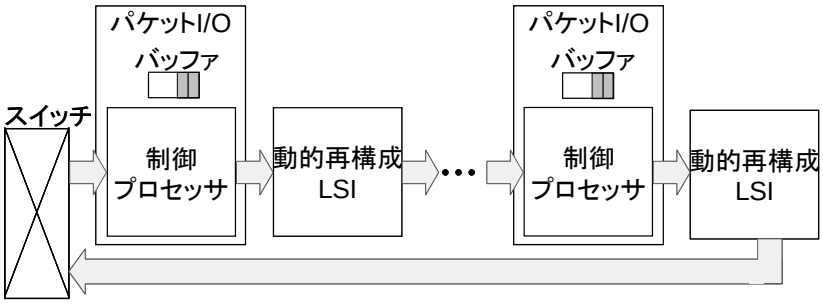


図 96　プロセッサの直列接続

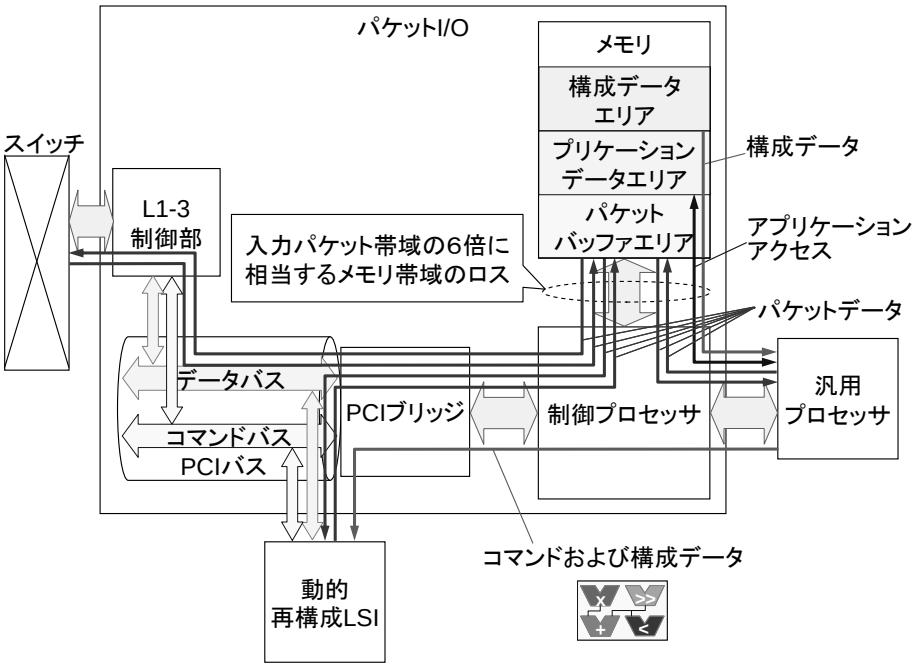


図 97　外部コマンドに基づく自動再構成

トの処理が終わるまで、処理が不要な後続パケットはパケットI/Oのバッファで待機する。アイドリングタイムが生じ、演算性能や応答速度が劣化する問題があった。

5.3.3　外部コマンドに基づく機能の手動切替

従来の動的再構成 LSI 搭載ネットワークアプライアンス装置は、プロセッサ外部からのコマンドをトリガとして、搭載論理の再構成を行う。マニュアル再構成[36][172] (ケース 1)と、汎用プロセッサを用いた外部コマンドに基づく自動再構成[72] (ケース 2: 図 97)の 2 種類がある。

ケース 1 のマニュアル再構成方式は、数日おきに進化する攻撃手法に対応せねばならない対異常通信防御用のネットワークアプライアンス装置において、数日おきに機能をアップデートする場合に使用する。ケース 2 の自動再構成方式は、動的再構成 LSI において、

暗号化等、アプリケーションの一部を高速化する場合に用いる。

本研究で提案するネットワークアプライアンスは、多様な機能(1)(2)を高速応答にてパケット毎に自動的に切替えて提供できる柔軟性を必要とする。ケース 1 のマニュアル再構成方式では、コマンドを手動で入力する速度が遅く、再構成に時間がかかるため、パケットの到着間隔に追いつかないという問題があった。また、ケース 2 の自動再構成方式では、パケットはメモリのパケットバッファエリアへ蓄積した後、汎用 CPU と動的再構成 LSI の両方がパケットの読み出しと書き込みを行う。そのため、図 97 に示すように、パケット入出力が回線帯域の 6 倍に相当するメモリ帯域を消費することになり、メモリや PCI バス帯域が不足して演算速度が減少する。更に、汎用プロセッサの使用により消費電力が増加するため、廃熱のための空気の通り道やファン等が必要となり筐体サイズが増加するという問題があった。

5.4 提案するクラウドサービス高速化技術

従来方式を用いたネットワークアプライアンスの問題を解決するため、パケット I/O，プロセッサ接続，プロセッサの機能再構成について、それぞれ以下の 3 つの方式を用いることを提案した。

i. メモリ非経由ダイレクトパケット I/O
ii. プロセッサの階層接続
iii. 端末間の通信状態に基づくパケット毎自己再構成

これらの提案方式および効果について、以下に詳細を説明する。

5.4.1 メモリ非経由のダイレクトパケット I/O

提案するメモリ非経由ダイレクトパケットI/O方式では、動的再構成LSIとスイッチの間のパケット入出力を、パケットを直接交換することが可能な演算マトリックスのダイレクトI/O経由で行う(図 98)。パケットバッファの中味を全て動的再構成LSIに渡すことなく、演算に必要なデータとポインタのみを渡すことによりI/Oを減らす。

本アーキテクチャは、パケットバッファとして外部メモリを使用しないため、メモリ帯域がパケットI/Oに使用されず、全メモリ帯域をPE(演算)マトリックスやRISCからのOS/アプリケーションアクセスに使用することが可能である。そのため、プロセッサがメモリ帯域の解放を待つアイドル時間が減少し、演算速度が向上する。

5.4.2 プロセッサの階層接続

提案するプロセッサの階層接続方式は、図 99 に示すように、パケット I/O の分類/分割器において、入力パケットを、後段の動的再構成 LSI における演算が必要なパケットまたは

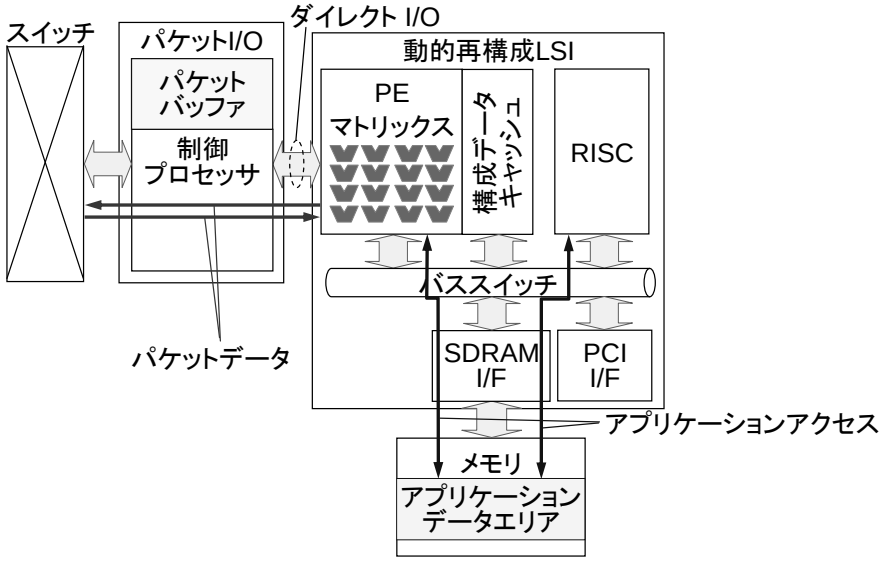


図 98　ダイレクトパケット I/O

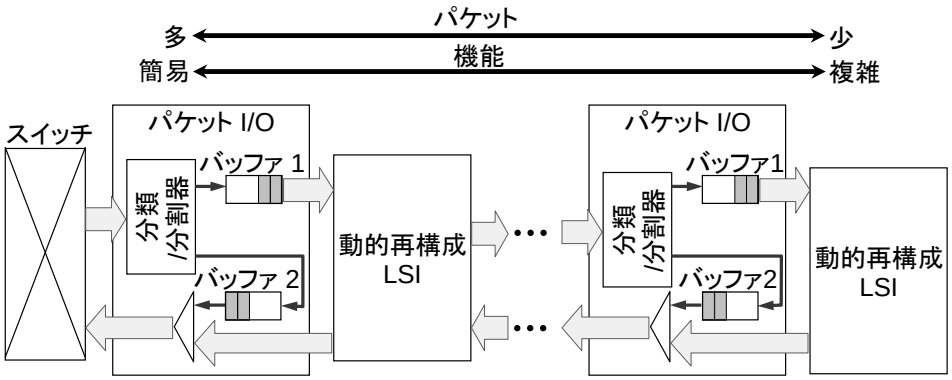


図 99 プロセッサの階層接続

小データ群と、演算が不要なパケットまたは小データ群へと分類・分割する。演算が必要なパケット・小データ群は、バッファ 1 に蓄積してから、演算が不要なパケット・小データ群へのポインタ情報と共に、隣の動的再構成 LSI へと出力する。演算が不要なパケット・小データ群は、バッファ 2 に蓄積してから、隣の動的再構成 LSI が出力する新規データと結合、あるいは新規生成パケットと合流して、スイッチや前段の動的再構成 LSI へと出力する。図 99 に示すように、複数組のパケット I/O と動的再構成 LSI を、同一ボード内に配置することで演算性能を向上させることも可能である。

本アーキテクチャは、パケットの要する処理が簡単な程、プロセッサ間のホップ数が減少し、パケットのプロセッサ通過時間が短くなる。入力パケットに対して、最短時間で必要とする機能による演算処理を実行して、演算処理の結果として新規生成パケットをスイッチに出力するので、クラウドサービス向けデータ処理の応答速度を向上可能である。加

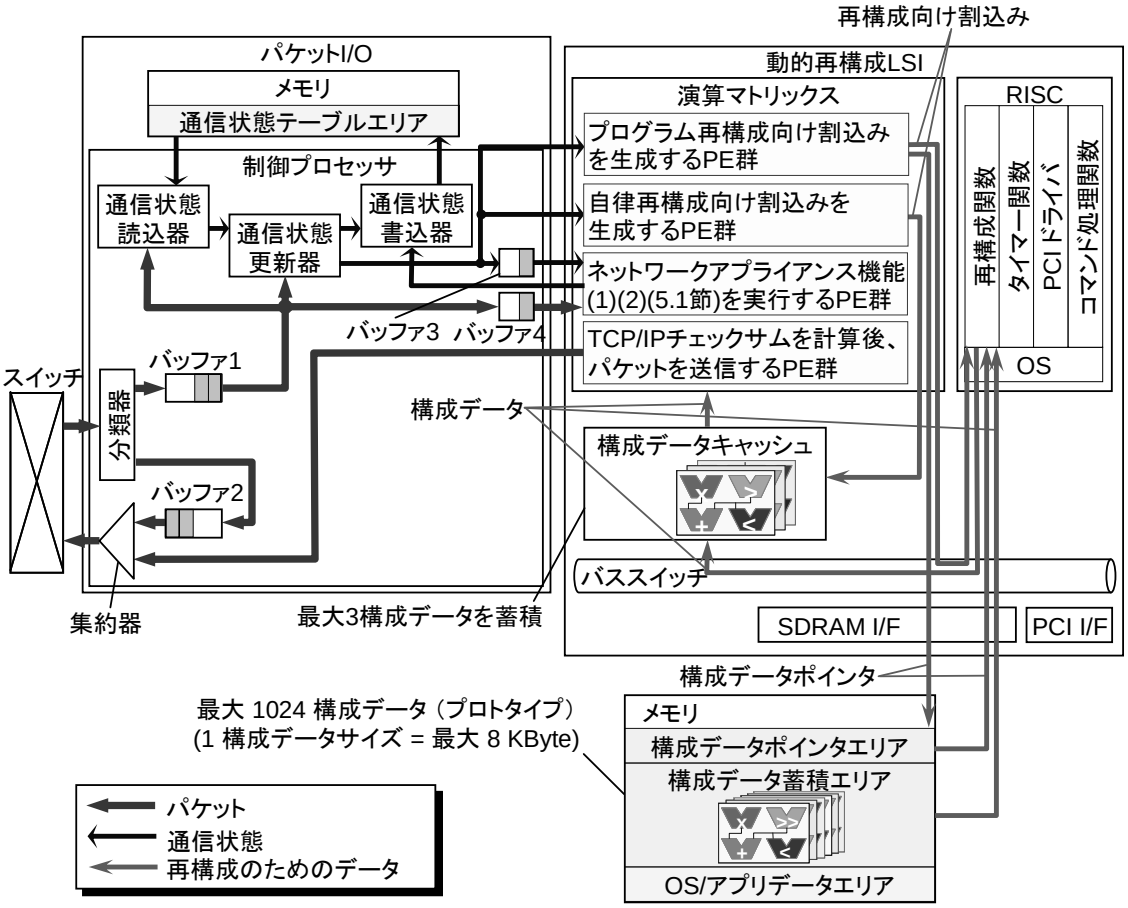


図 100　通信状態に基づくパケット毎自己再構成

えて、動的再構成 LSI で実施する機能の演算処理が複雑になるほど、通過するパケット数が減少し、動的再構成 LSI が受け取る演算データ量は減少する。複数の動的再構成 LSI に演算負荷を分散させることができるので、演算速度を向上させやすい。

5.4.3　通信情報に基づく機能の自己再構成

提案する通信情報に基づく機能の自己再構成方式では、動的再構成 LSI は、端末間の通信状態に基づいて自律的に回路を再構成する。

パケット I/O は、端末間の通信状態を蓄積した通信状態テーブルと、通信状態テーブルの通信状態を読み込んだり書き込んだり更新したりする通信状態書込/読込/更新器と、分類器とバッファ 1, 2 (5.4.2 節記載)を備える(図 100)。

通信状態読込器は、バッファ 1 が出力するパケット記載の送信元および宛先の IP アドレスと TCP ポート番号に基づき、通信状態テーブルから通信状態を読み出す。通信状態更新器は、通信状態読込器の読み出した通信状態を、バッファ 1 が出力するパケットに基づき更新する。通信状態書込器は、通信状態更新器の更新した通信状態を、通信状態テーブル

に書き込む。

更新した通信状態は、自律再構成向け割込みを生成する PE (Prosessing Element)群と、プログラム再構成向け割込みを生成する PE 群へと出力される。演算マトリックスの再構成が完了するまで、更新した通信状態と、バッファ 1 が出力したパケットは、一時的にバッファ 3 およびバッファ 4 にて待機する。

自律再構成向けに、PE 群は通信状態を使って割込みを生成する。生成した割込みをトリガとして、現在の構成データが予め指定する次の構成データが、構成キャッシュから演算マトリックスに 1 クロックサイクルでロードされ、実行される。自律再構成向け割込みが生成されない場合、PE 群はプログラム再構成向けに割込みを生成する。PE 群は次の構成データを特定する構成データポインタをメモリに書き込む。

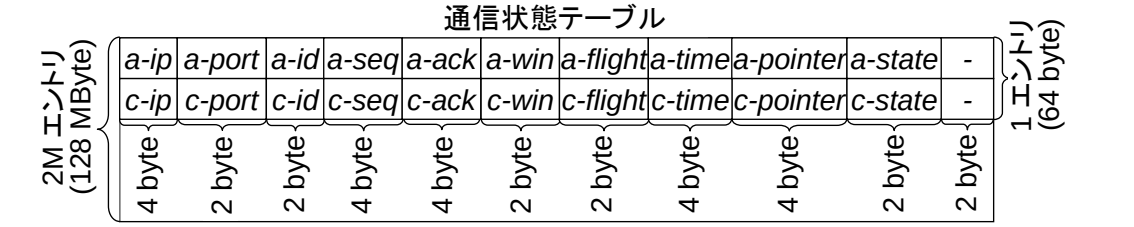
RISC 上で稼動している OS は、演算マトリックスからプログラム再構成向け割込みを受け取ると、再構成実行関数を呼び出す。再構成実行関数は、メモリ内の構成データポインタを参照して、構成データキャッシュが次の構成データを蓄積しているかを判定する。構成データキャッシュが次の構成データを蓄積していない場合、再構成実行関数はメモリ内の構成データエリアから構成データキャッシュに次の構成データを転送する。更に、再構成実行関数は、次の構成データを構成データキャッシュから演算マトリックスにロードして実行する。構成データをロードして実行するまでの間、演算マトリックスにおける演算処理は停止する。本プログラム再構成は、構成データキャッシュが構成データを蓄積している場合で 1000 クロックサイクルを、構成データキャッシュが構成データを蓄積しておらず外部メモリに蓄積している場合で 20000 クロックサイクルを要する。なお、演算マトリックスにおける演算が停止している間でも、他の部分の演算は継続する。

再構成が完了すると、一時的にバッファ 3 とバッファ 4 で待機していたパケットと通信状態は、演算(PE)マトリックス上のクラウドサービス高速化系やセキュリティ系のネットワークアプライアンス機能(1)(2)(5.1 節)を実行する PE 群に移動する。ネットワークアプライアンス機能(1)(2)を実行する PE 群での演算処理が終了すると、演算処理の結果として新たに生成したパケットを、TCP/IP チェックサム計算後にパケットを送出する PE 群を経由してパケット I/O に送信する。

上記のアーキテクチャにより、各端末間の TCP のフィードバック制御に応じて、少しずつデータを交換したり、コマンドを実行したりすることが可能となる。

以下の項では、本研究において開発したネットワークアプライアンスの試作機の例を用いて、通信状態のフォーマットおよびシーケンスと、様々な構成データ間での再構成サイクルを詳細に説明する。

開発したネットワークアプライアンスの試作機は、5.1 節記載のセキュリティ系の機能(2)の対異常通信防御の一部(IPS 機能(3.2.2 参照))と、クラウドサービス高速化系の機能(1)の HTTP に載せた SQL コマンドにより DB を操作するクエリトランザクションの一部と、クラウド DC のデータを一時的に蓄積するキャッシュの一部を実装している。



a/c-ip: アクセス/クラウドDC端末のIPアドレス
a/c-port: アクセス/クラウドDC端末のTCPポート
a/c-id: アクセス/クラウドDC端末に送った最後のIPパケットのID番号
a/c-seq: アクセス/クラウドDC端末のコネクションのTCPセグメントシーケンス番号
a/c-ack: アクセス/クラウドDC端末のコネクションのTCPセグメントACK番号
a/c-win: アクセス/クラウドDC端末のコネクションのTCPウィンドウサイズ
a/c-flight: アクセス/クラウドDC端末へのパケット送信時に使用するTCPウィンドウサイズ
a/c-time: アクセス/クラウドDC端末からパケットが到着した最新の時刻
a/c-pointer: アクセス/クラウドDC端末からのパケット処理時に使用するポインタ
a/c-state: アクセス/クラウドDC端末の通信コネクションの通信状態番号

図 101　通信状態テーブルのフォーマット

<u>通信状態</u>

本研究において開発したネットワークアプライアンスの試作機は、アクセス側端末との通信、および、クラウド DC 端末との通信の 2 種類の通信を取り扱う(図 93 参照)。2 種類の通信状態を管理するため、通信状態テーブルは、アクセス側端末とクラウド DC 端末の IP アドレス(*a/c-ip*)と TCP ポート番号(*a/c-ip*)毎に 2 つの通信状態(*a/c-state*)を蓄積する(図 101)。

通信状態のフォーマットの一例を図 101 に示す。この例は、HTTP に載せて SQL コマンドを用いた DB 操作を行う時の構成例である。2 つの通信状態を、*a/c(access/cloud)-state* で示す。1 つのエントリは、20 個のアイテムから成る。*a/c-ip*，*a/c-port* の 4 個のアイテムは、通信を特定するためのアイテムである。通信状態読込器(図 100)は、パケットヘッダに記載の通信を特定する 4 個のアイテムを持つエントリを読み出す。*a/c -id*，*a/c -seq*，*a/c -ack*，*a/c -win*，*a/c -flight* の 10 個のアイテムは、新しいパケットを返信する際に、パケットヘッダを生成するために使用する。*a/c -time* は、通信状態を消去するかリフレッシュするかを判定するために使用する。*a/c -pointer* は、動的再構成 LSI がメモリ内のアプリケーションデータを読み出すために使用する。*a/c -state* は、通信状態を特定するための番号である。

通信状態(*a/c -state*)は、バッファ 1 から読み出したパケット記載の情報や、処理の進捗状況に基づき遷移する(図 102)。更新済み通信状態と、バッファ 1 からのパケットは、演算マトリックスの再構成が完了するまで、バッファ 3 とバッファ 4 で待機する(図 100)。更に、更新済み通信状態は、自律/プログラム再構成向け割込みを生成する PE 群に移動する。PE 群は、通信状態に応じて自律/プログラム再構成向け割込みを生成し、構成データポインタを書き換える。自律再構成向け割込みが発生すると、使用中の構成データが予め指定する

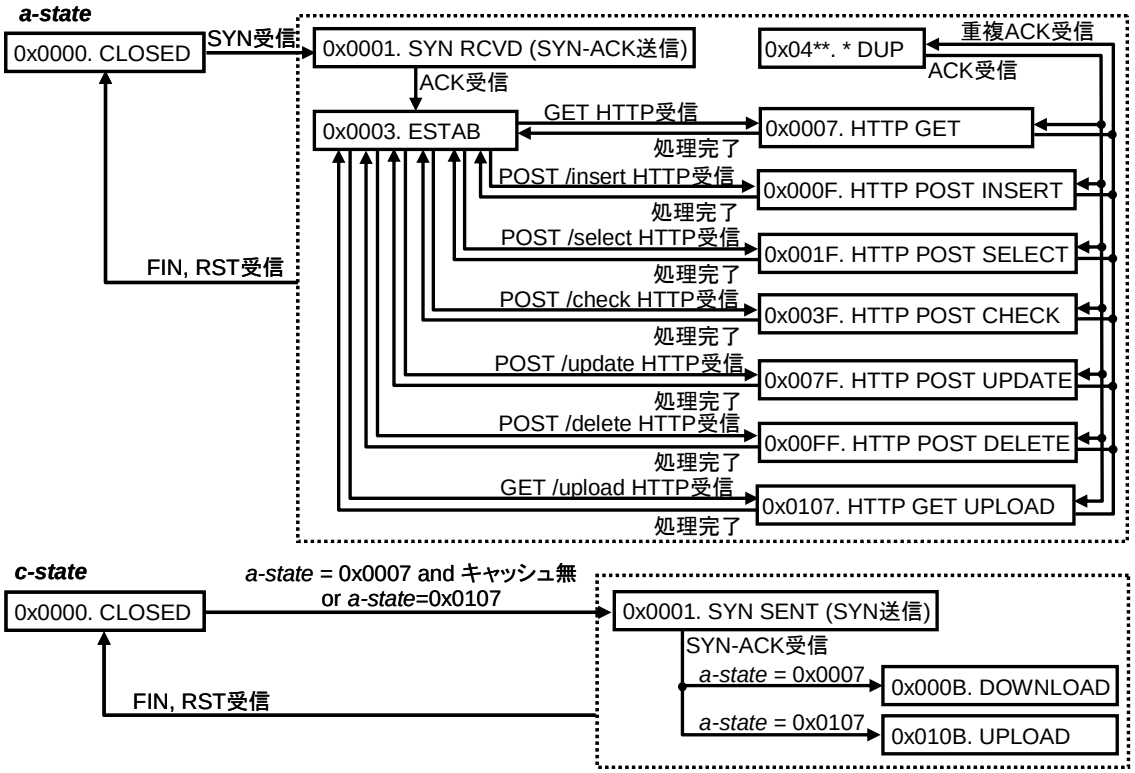


図 102　通信状態(*a/c-state*)の遷移

次の構成データが、構成データキャッシュから演算マトリックスに 1 クロックサイクルでロードされ、実行される。

この例では、アクセス側の端末との通信状態 *a-state* は、パケットデータと、通信状態テーブル記載の通信状態 *a-state* に応じて変化する(図 102)。

最初の段階では、通信状態 *a-state* は TCP コネクション状態における”CLOSED” [121]を意味する’0x0000’である。SYN パケットを受信すると(図 102 における rcv SYN)、通信状態 *a-state* は、”SYN RCVD” [121]を意味する’0x0001’に変化する。更に、ACK パケットを受信すると(図 102 における rcv ACK)、通信状態 *a-state* は、コネクション確立”ESTAB” [121]を意味する’0x0003’に変化する。

通信状態 *a-state* は、コネクション確立を表す’0x0003’に遷移した後、到着パケットのペイロード内の HTTP (HyperText Transfer Protocol)の GET/POST コマンドに応じて変化する。パケットのペイロードが HTTP の GET コマンドを含み他のクエリを含まない場合、通信状態 *a-state* は、クライアントが要求するファイルの送信を要求する”HTTP GET”を意味する’0x0007’に変化する。パケットのペイロードが HTTP の POST コマンドを含み、引数として”/insert”、 ”/select”、 “/check”、 “/update”、 “/delete”クエリを持つ POST コマンドを含む場合、通信状態 *a-state* は、ネットワークアプライアンスの SQL データベースに対して、クライアントが送ってきたアイテムデータの挿入/選択/確認/更新/削除を要求する” HTTP POST INSERT/SELECT/CHECK/UPDATE/DELETE”を意味する他の番号 0x000F、0x001F、

0x003F、0x007F、0x00FF に変化する。パケットのペイロードが HTTP の GET コマンドを含み、引数として”/upload”クエリを持つ場合、通信状態 *a-state* は、アプライアンスのデータベースのアイテムデータをクラウドサーバへアップロードすることを要求する”HTTP GET UPLOAD”を意味する’0x0107’に変化する。

更に、重複 ACK[173]を持つパケットが来ると、TCP 輻輳制御の高速再送と高速回復[173]を要求する”DUP”を意味する’0x0400’を通信状態 *a-state* に加算する。通信状態 *a-state* が 0x0007、0x000F, 0x001F, 0x003F, 0x007F, 0x00FF、0x0107 になった後で、HTTP の GET/POST コマンドが要求する処理が全て終了すると、*a-state* は’3’に戻る。なお、FIN-ACK/RST-ACK パケットが到着すると、通信状態 *a-state* はそれまでの値に関わらず強制的に’0’に戻る。

クラウド DC 側の端末との通信状態 *c-state* は、サーバデータをキャッシュにダウンロードしたり、キャッシュに蓄積したデータベースの内容をサーバにアップロードしたりする要求を受けたときに変化する。例えば、通信状態 *a-state* がファイルの送信を要求する’0x0007’の時に、アクセス側のクライアント端末が要求するファイルを、ネットワークアプライアンスのメモリが蓄積していない場合、通信状態 *c-state* は”SYN SENT” [121]を意味する’0x0001’に変化する。アクセス側のクライアント端末から、キャッシュに蓄積したデータベースの内容をサーバにアップロードする要求を受けて通信状態 *a-state* が’0x0107’に変化した時も、通信状態 *c-state* は”SYN SENT” [121]を意味する’0x0001’に変化する。通信状態 *c-state* の変化後、ネットワークアプライアンスは SYN パケットをクラウド DC のサーバ端末に送信する。その後、SYN-ACK パケットがサーバ端末から到着すると、通信状態 *a-state* がファイルの送信を要求する’0x0007’の場合、通信状態 *c-state* は、サーバからネットワークアプライアンスへのファイルダウンロードを要求する”DOWNLOAD”を意味する’0x000B’に変化する。通信状態 *a-state* がアップロードを要求する’0x0107’の場合、通信状態 *c-state* は、アプライアンス内のメモリにキャッシュしたデータベースをサーバへアップロードすることを要求する”UPLOAD”を意味する’0x010B’に変化する。FIN-ACK/RST-ACK パケットが到着すると、通信状態 *c-state* はそれまでの値に関わらず強制的に’0’に戻る。

<u>再構成サイクル</u>

本研究において開発したネットワークアプライアンスの試作機が搭載する動的再構成 LSI は、14 個の構成データを使用して、HTTP に載せた SQL コマンドにより DB を操作するクエリトランザクションを行うクラウドサービス高速化系(1)の機能(5.1 節)と、対異常通信防御を行うセキュリティ系(2)の機能(5.1 節)を実現する。14 個の構成データのうち、構成データキャッシュに蓄積可能な構成データは 3 つであり、残りの 11 個は外部メモリの構成データ蓄積エリアに蓄積する。構成データは 1 つずつシーケンシャルに実行可能である。

対異常通信防御を行うセキュリティ系(2)の機能は、構成データキャッシュに常駐する全パケット共通の構成データと、対異常通信防御機能を実現する構成データを、1 クロックサイクルでの自律再構成により切り替えてシーケンシャルに使用することにより実行する(図

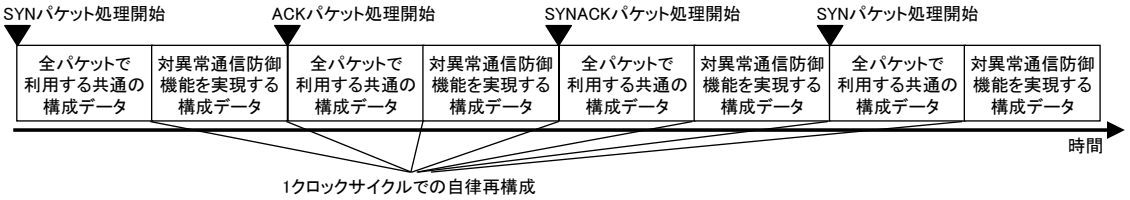


図 103　対異常通信防御を行う機能を実現する構成データの利用方法

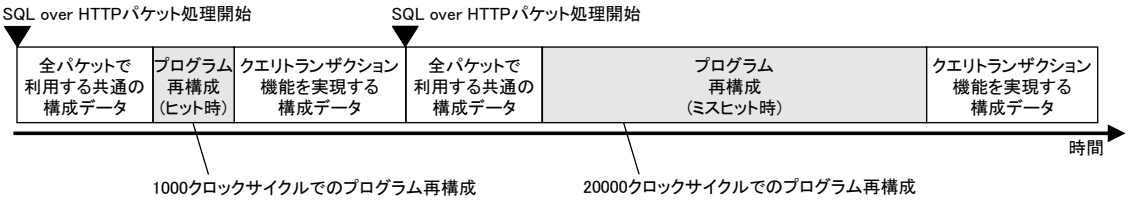


図 104　クエリトランザクションを行う機能を実現する構成データの利用方法

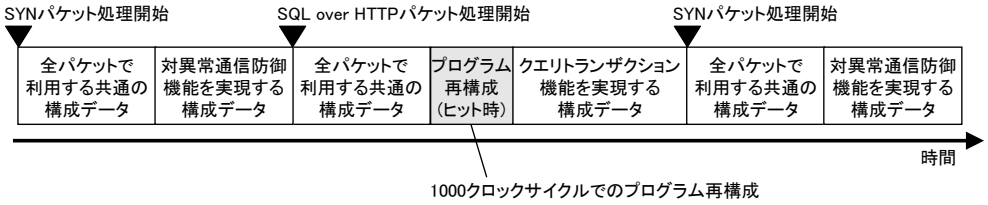


図 105　対異常通信防御とクエリトランザクションを実行する場合の構成データ利用方法

103)。SYN パケットや SYN-ACK パケット等に対して演算処理を行う。

一方、HTTP 上で SQL を使用してデータベースを操作するクエリトランザクションを行うクラウドサービス高速化系(1)の機能は、構成データキャッシュに常駐する全パケット共通の構成データと、構成データキャッシュまたは外部メモリの構成データ蓄積エリアのいずれかに存在するクエリトランザクション実行用の構成データをシーケンシャルに使用して実行する。全パケット共通の構成データは、1 クロックサイクルでの自律再構成により切り替えて使用するが、クエリトランザクション実行用の構成データは、キャッシュヒット時で 1000 クロックサイクル、キャッシュミスヒット時で 20000 クロックサイクルでのプログラム再構成により切り替えて使用する(図 104)。プログラム再構成中は、動的再構成 LSI における演算処理は停止するので、キャッシュミスヒットが続くと性能が低下する。

また、図 105 に示すように、対異常通信防御を行うセキュリティ系(2)の機能を必要とするパケットが到着した後に、SQL を使用してデータベースを操作するクエリトランザクシ

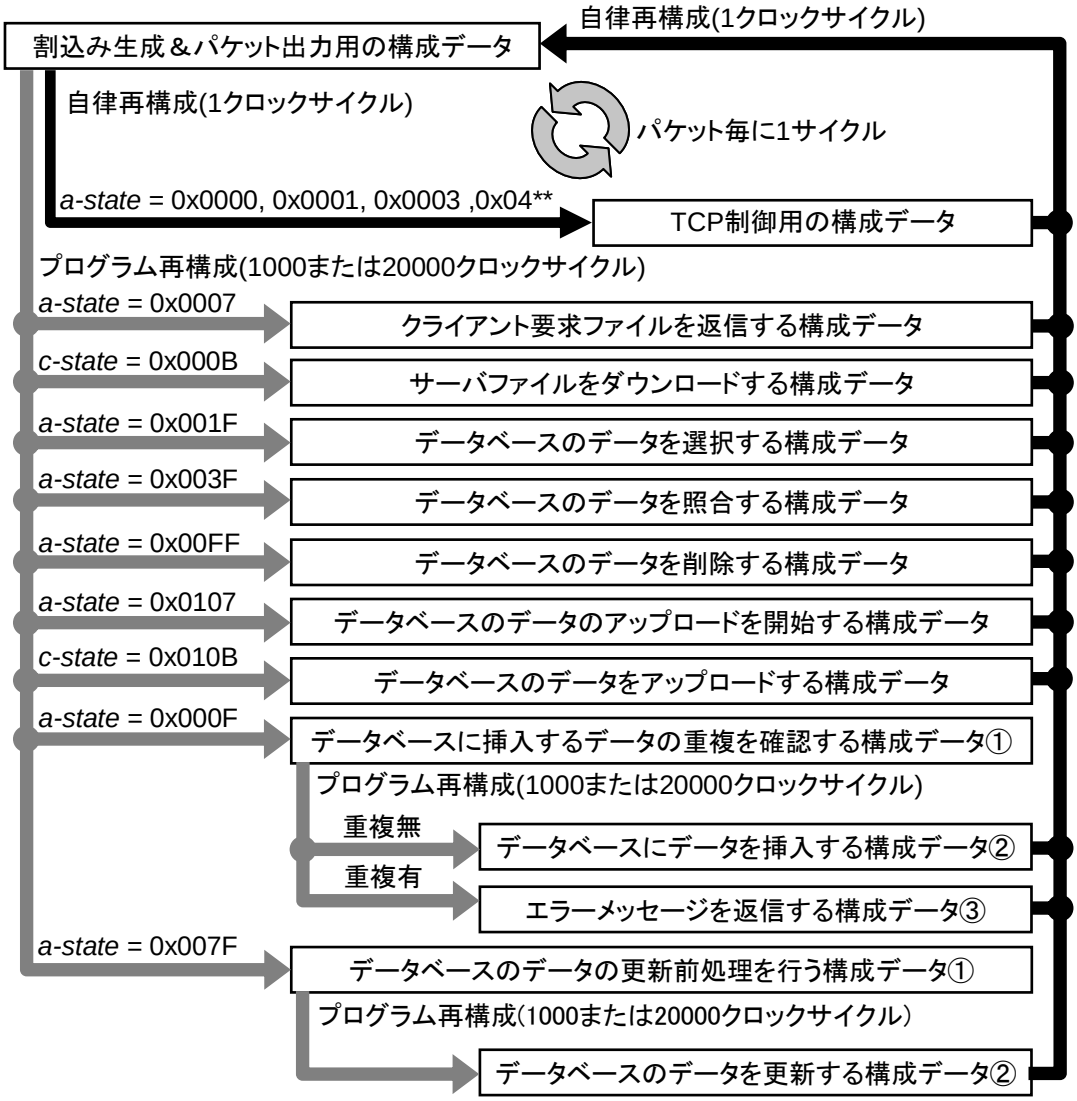


図 106　再構成サイクル

ョンを行うクラウドサービス高速化系(1)の機能を必要とするパケットが到着した場合は、最初に到着したパケットに対して対異常通信防御を行うセキュリティ系(2)の機能を、自律再構成により実行する。その後に到着したパケットに対しては、SQL を使用してデータベースを操作するクエリトランザクションを行うクラウドサービス高速化系(1)の機能を、プログラム再構成により実行する。それぞれの機能をパケット毎にシーケンシャルに実行する。以下に構成データを使い分ける方法の詳細を説明する。

ネットワークアプライアンスの試作機は、図 106 に示した再構成サイクルを用いて、通信状態(*a/c-state*)の値に応じて 14 個の構成データを使い分け、パケット毎に 2～3 回、演算マトリックスの論理を動的に再構成する。これにより、5.1 節記載のセキュリティ系の機能(2)の対異常通信防御や、クラウドサービス高速化系の機能(1)の SQL を使用してデータベー

スを操作するクエリトランザクション、キャッシュ等の様々な機能を高速応答に提供する柔軟性を実現することが可能となる。多くの演算量を必要とするトランザクション(データベースへのデータ挿入、データベースのデータ更新、等)程、使用する構成データ数が多くなる。

以下に、本研究で試作機に実装した SQL over HTTP 等から成る 11 種類のトランザクション処理を例に挙げ、図 106 を用いて詳細に説明する。

■割込み生成とパケット出力を行う構成(図 106)

動的再構成 LSI は、パケット I/O から通信状態を受信する時は常に本構成を実行する。本構成を用いた演算は、全てのパケットに対して実行され、割込みを生成する処理と、TCP/IP チェックサムを計算して新たに生成したパケットを送信する処理を実行する。本構成は、全パケットで利用する共通の構成データであり、使用頻度が最も高いため、構成データキャッシュに常に蓄積しておき、自律再構成を用いて 1 クロックサイクルで再構成する。

■TCP コネクション制御を行う構成(図 106)

アクセス側端末との通信状態 *a-state* が、TCP セッションを確立したり廃棄パケットを再送したりするための状態'0x0000'，'0x0001'，'0x0003'，'0x04**'である場合に、動的再構成 LSI は TCP コネクション制御に特化した本構成を使用する。本構成は、異常な TCP セグメントシーケンス/確認番号を持つパケットを廃棄する他、通信状態が"SYN RCVD"の時に SYN-ACK パケットの生成と、通信状態が"CLOSED"の時に RST/FIN-ACK および ACK パケットの生成と、パケット生成の時に通信状態の *f/b-id* の変更を行う。本構成は、対異常通信防御を行うための構成データであり、使用頻度が高いため、常に構成データキャッシュに蓄積しておき、自律再構成を用いて 1 クロックサイクルで再構成する。

■HTTP GET コマンド受信時のファイル送信トランザクションを実行する構成(図 106)

アクセス側端末からファイル送信を要求する GET コマンドを受信して、通信状態 *a-state* が'0x0007'となった時に、動的再構成 LSI は、本構成を使用して、クライアント要求ファイルを外付けメモリ内のキャッシュ領域に蓄積しているか否かを判定する。クライアント要求ファイルをキャッシュ領域に蓄積している場合、クライアント向けに、キャッシュに蓄積した要求ファイルのデータを含むパケットを生成する。データのサイズが MSS (Maximum Segment Size)を超える場合は、複数のパケットに分割して生成する。クライアント要求ファイルをキャッシュ領域に蓄積していない場合は、通信状態 *c-state* を'0x0001'にしてクラウド DC 側のサーバ端末と新規 TCP コネクションを張るための SYN パケットを生成する。その後、クラウド DC のサーバ端末から SYNACK パケットを受信してコネクションが確立すると、通信状態 *c-state* を、クラウド DC 側のサーバ端末からファイルをダウンロードするための状態'0x000B'に設定する。

■キャッシュミスヒット時のファイルダウンロードを実行する構成(図 106)

アクセス側端末との通信状態 *a-state* が、クラウド DC 側のサーバ端末からファイルをダウンロードするための状態’0x000B’である場合、動的再構成 LSI は本構成を使用する。本構成は、サーバからパケットを受信すると、ネットワークアプライアンスの外部メモリにファイルをダウンロードして蓄積する。ファイルサイズが MSS を超える場合は、最後尾のデータを蓄積してからキャッシュを有効にする。

■HTTP POST コマンド内の SQL 選択(select)コマンドを実行する構成(図 106)

アクセス側端末との通信状態 *a-state* が、SQL の選択コマンドを実行するための状態’0x001F’である場合、動的再構成 LSI は本構成を使用する。本構成は、パケットの選択コマンドが指定する内容に応じてアプライアンスのメモリ内の DB からアイテムデータを選択して、HTML/XML テキスト形式に翻訳したアイテムデータからなるパケットを生成する。アイテムデータのサイズが MSS を超える場合は、複数のパケットに分割して生成する。

■HTTP POST コマンド内の SQL 挿入(insert)コマンドを実行する構成①～③(図 106)

アクセス側端末との通信状態 *a-state* が、SQL の挿入コマンドを実行するための状態’0x000F’である場合、動的再構成 LSI は 1 枚目の構成①を使用する。本構成①は、ネットワークアプライアンスの外部メモリ内の DB が、クライアントが挿入しようとするデータを既に登録しているかを判定する。判定結果に基づき、クライアントが挿入しようとするデータを DB が登録していない場合、動的再構成 LSI は 2 枚目の構成②を使用する。構成②は、アプライアンスのメモリ内の DB にクライアントが送付したアイテムデータを挿入して、データの挿入が正しく完了したことをクライアントに通知するパケットを生成する。一方で、クライアントが挿入しようとするデータを DB が既に登録している場合、動的再構成 LSI は 3 枚目の構成③を使用する。構成③は、エラーメッセージを出力するパケットを生成する。

■HTTP POST コマンド内の SQL 照合(check)コマンドを実行する構成(図 106)

アクセス側端末との通信状態 *a-state* が、SQL の照合コマンドを実行するための状態’0x003F’である場合、動的再構成 LSI は、本構成を使用する。本構成は、ネットワークアプライアンスのメモリ内の DB が、クライアントが指定したアイテムデータを登録しているか否かを判定して判定結果をアクセス側端末へ通知するパケットを生成する。

■HTTP POST コマンド内の SQL 削除(delete)コマンドを実行する構成(図 106)

アクセス側端末との通信状態 *a-state* が、SQL の削除コマンドを実行するための状態’0x00FF’である場合、動的再構成 LSI は、本構成を使用する。本構成は、パケットの削除コマンドの引数としてクライアントが指定したアイテムデータをアプライアンスのメモリ内の DB から削除して、削除の結果を通知するパケットを生成する構成を使用する。

■HTTP POST コマンド内の SQL 更新(update)コマンドを実行する構成①②(図 106)

アクセス側端末との通信状態 *a-state* が、SQL の更新コマンドを実行するための状

態’0x007F’である場合、動的再構成 LSI は、1 枚目の構成①を使用する。構成①は、クライアントが更新対象に指定した古いアイテムデータを DB から削除する。その後で、プログラム再構成により、構成②に切替えて使用する。構成②は、クライアントが更新する新規アイテムデータを DB に登録する。

■HTTP GET コマンド内の SQL アップロードコマンドを実行する構成(図 106)

アクセス側端末との通信状態 *a-state* が、SQL のアップロードコマンドを実行するための状態’0x0107’である場合、動的再構成 LSI は、本構成を使用する。本構成は、クラウド DC 側端末との通信状態 *c-state* を’0x010B’に設定して、クラウド DC 側のサーバ端末向けに SYN パケットを生成し、更に、アクセス側のクライアント端末向けにアップロードを開始したことを通知するパケットを生成する。

■ネットワークアプライアンスの DB のデータをアップロードする構成(図 106)

クラウド DC 側端末との通信状態 *c-state* が、ネットワークアプライアンスの DB のデータをアップロードするための状態’0x010B’である場合、動的再構成 LSI は、本構成を使用する。本構成は、アプライアンスのメモリがキャッシュした DB の内容をクラウド DC 側のサーバにアップロードする。

通信状態毎に異なるトランザクション処理が完了すると、演算マトリックスは自律再構成により、割込み生成とパケット出力を行う構成に 1 クロックサイクルで再構成する。

割込み生成とパケット出力を行う構成データと、TCP コネクション制御を行う構成データは常に構成データキャッシュに蓄積され、1 クロックサイクルで自律再構成する。これら以外の構成データは、構成データキャッシュまたは動的再構成 LSI の外付けメモリに蓄積され、プログラム割込みを使用して演算マトリックスにロードされる。再構成する際に、構成データキャッシュが構成データをキャッシュしている場合で 1000 クロックサイクル、構成データキャッシュが構成データをキャッシュしておらず外部メモリからダウンロードする場合で 20000 クロックサイクルの時間を要する。

以上述べた再構成サイクルを用いた通信状態に基づくパケット毎自己再構成を行うことで、ネットワークアプライアンスの多様な機能(1)(2)を通信状態に応じて高速応答にて再構成することができ、パケット毎に最適化した並列演算が可能となる。

5.5 評価実験

5.5.1 評価に用いた試作機

本研究では、提案方式をテストするために試作機を開発した。試作機とそのアーキテクチャを図 107 に示す。

メインプロセッサ向け動的再構成 LSI(図 107 の 1)は、376 個の演算エレメント PE(168 個の実行エレメント EXE と、136 の遅延エレメント DEL と、608KByte のメモリエレメント

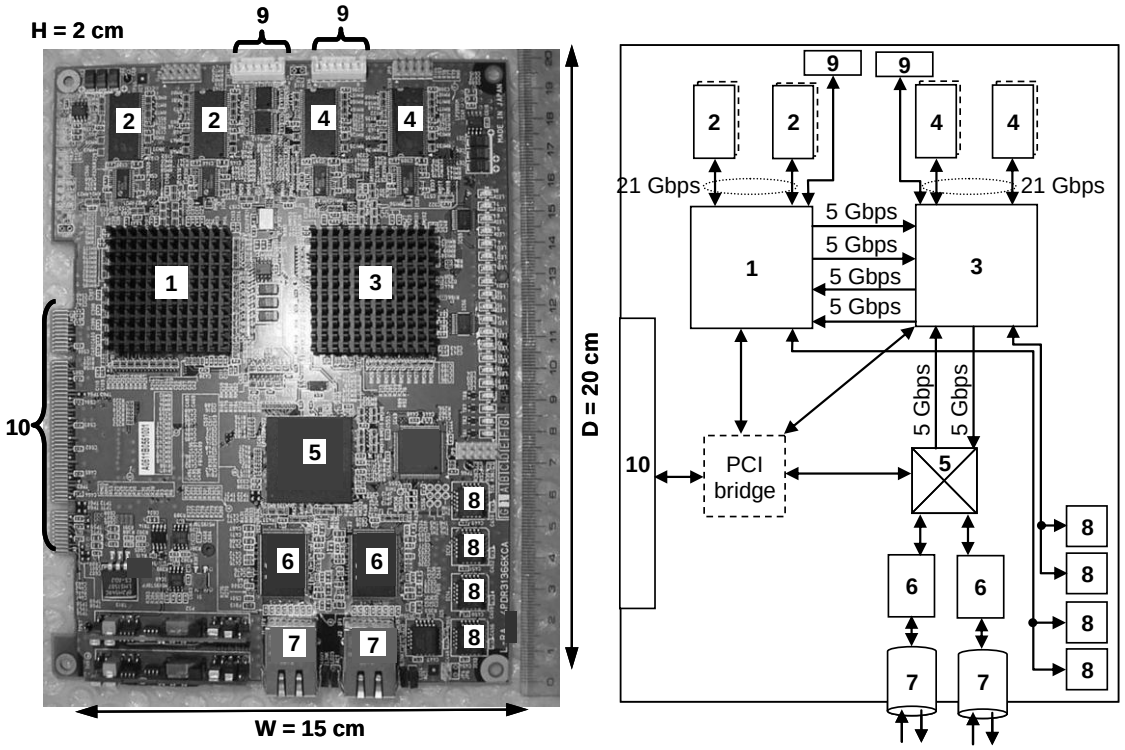


図 107　ネットワークアプライアンスの試作機

RAM と、8 個のダイレクト I/O と 8 個のメモリ I/O を含む)から成る PE マトリックスを備える[114][115]。動作周波数 166MHz で稼動し、21Gbps(64-bit 166-MHz DDR)のメモリ帯域を持つ。OS、アプリケーション、構成データは、プログラムを記憶するための EEPROM(図 107 の 8)に書き込み、シリアルポート(図 107 の 9)や PCI バス(図 107 の 10)経由でアップデートする。外部メモリ(図 107 の 2)は 256MByte あり、OS、アプリケーション、構成データ、データベース向けのデータを一時的に蓄積する。

パケットが 2 本の 1-Gbps 回線(図 107 の 7)から到着すると、スイッチ(図 107 の 5)を経由して、L1-2 制御部(図 107 の 6)からパケット I/O の制御プロセッサ(図 107 の 3)へとルーティングされる。パケット I/O の制御プロセッサには、メインプロセッサと同一の動的再構成 LSI を使用する。パケットの通信状態は、256MByte の外部メモリ(図 107 の 4)内の通信状態テーブルエリアに蓄積する。

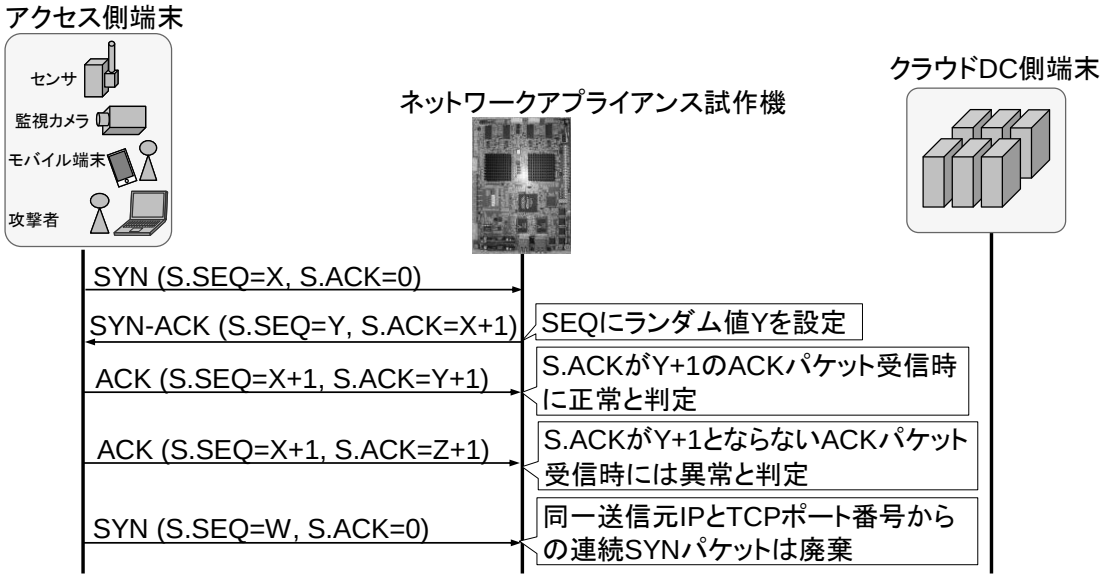


図 108　対異常通信防御の一部である IPS 機能

5.5.2　試作機に実装した機能

クラウドサービスを高速化するネットワークアプライアンスの試作機は、セキュリティ系の機能(2)の対異常通信防御の一部(IPS 機能(3.2.2 参照))と、クラウドサービス高速化系の機能(1)のクエリトランザクションおよびキャッシュの一部として、以下の 4 つの機能を実装した。実装したクエリトランザクションは、SQL over HTTP を用いた一部の SQL コマンドのみに対応する。また、SQL コマンドは 1 パケットに収まるサイズを想定する。

① 対異常通信防御の一部である IPS 機能(セキュリティ機能)
② クライアント要求ファイルの返信とキャッシュ(クラウド DC のデータのキャッシュ)
③ データベースに対するデータ選択/登録/削除/更新(DB 向けクエリトランザクション)
④ キャッシュした DB 内容をサーバへアップロード(DB 向けクエリトランザクション)

本研究では、これらの機能について実験評価を行った。以下、試作した機能について詳細を説明する。実験評価の結果は、5.5.3 節にて説明する。

対異常通信防御の一部である IPS 機能

ネットワークアプライアンスの試作機は、図 108 に示すように、接続要求パケットである SYN パケット[121]を受信すると、ランダムな値 Y を送信シーケンス番号 S.SEQ に付与した SYN-ACK パケット[121]を返信する。その後、受信シーケンス番号 S.ACK が Y+1 である ACK パケット[121]を受信した時点で、正常な通信と判定する(図 108)。

ネットワークアプライアンスの試作機が、受信シーケンス番号 S.ACK として正しい値 Y+1 を持たない ACK パケットを受信した場合は、異常通信と判定してパケットを廃棄する

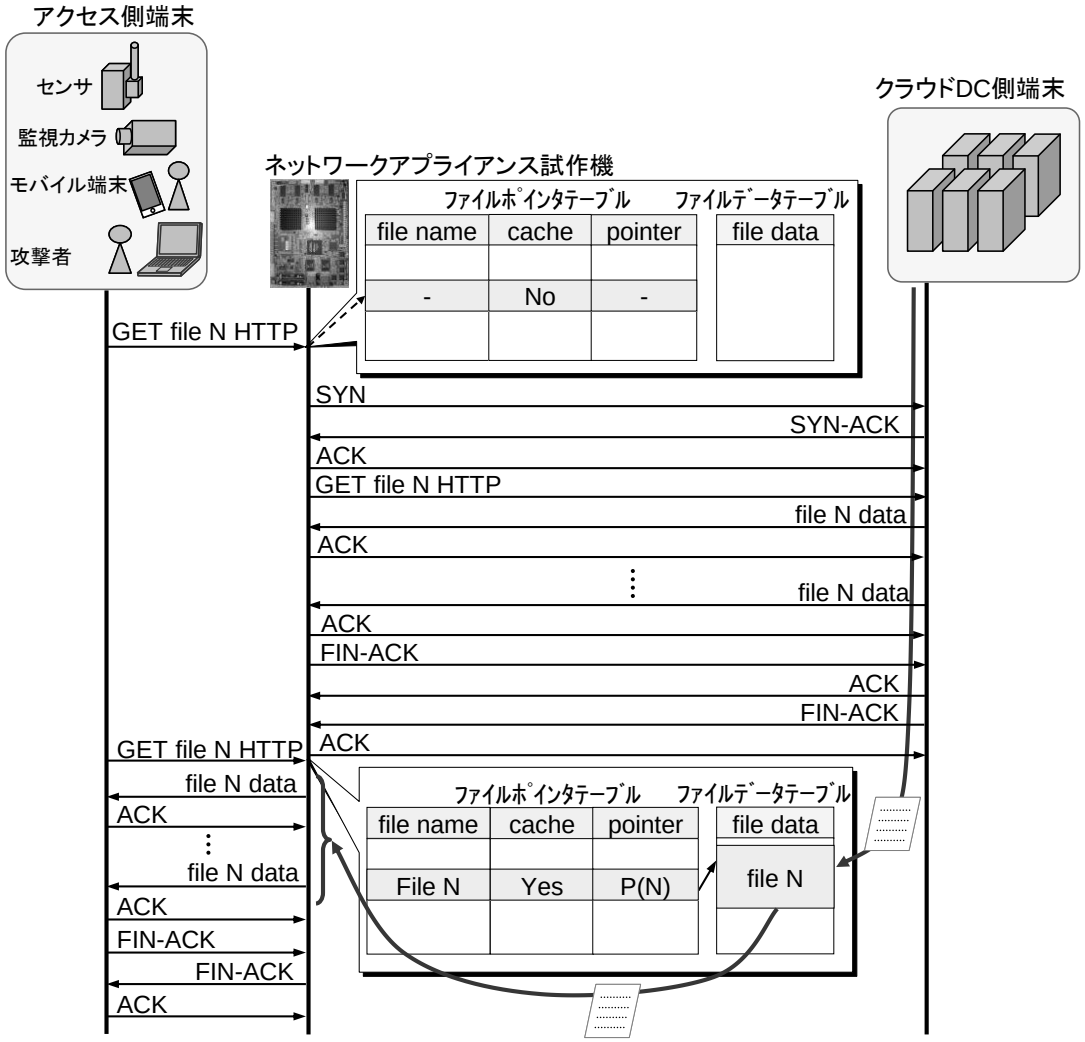


図 109　クライアント要求ファイルの返信と自動キャッシュ

(図 108)。その他、同一送信元 IP および同一送信元ポート番号から連続して短期間に大量の SYN パケットを受信した場合は、DoS 攻撃と判定してパケットを廃棄する(図 108)。

クライアント要求ファイルの返信とキャッシュ

ネットワークアプライアンスの試作機は、外付けメモリ(図 107 の 2)において、クラウド DC 側のサーバ端末からダウンロードしたファイルをキャッシュするファイルデータテーブルと、ファイル名毎にキャッシュファイルへのアドレスポインタを記録するファイルポインタテーブルを備える(図 109)。

ネットワークアプライアンスの試作機が、アクセス側端末から、クラウド DC 側のサーバ端末に対してファイル N のダウンロードを要求する HTTP GET コマンドを持つパケットを受信すると、ファイル名 N を用いてファイルポインタテーブルを検索し、ファイルデータテーブルが GET コマンドの要求ファイル(ファイル N)をキャッシュしているかを判定する。

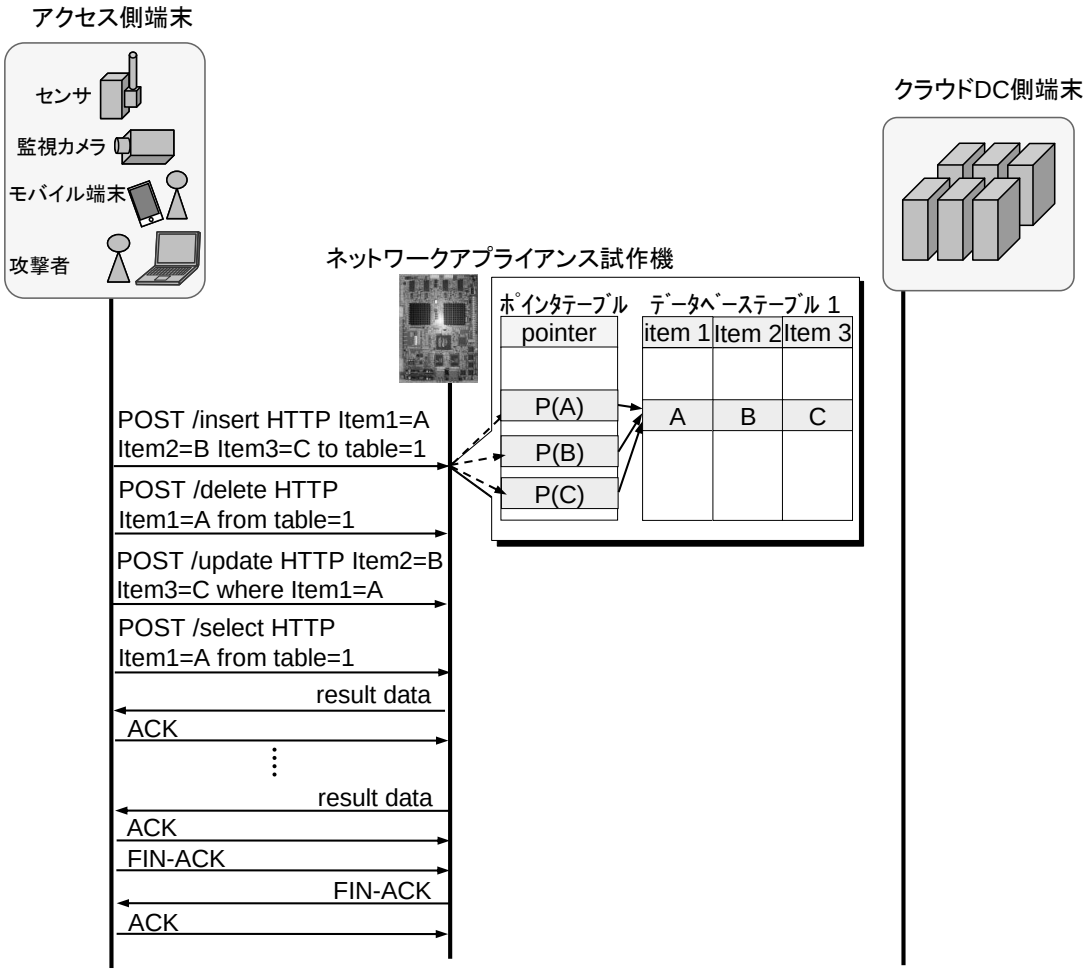


図 110　DB へのデータ挿入/削除/更新/選択

ファイルデータテーブルが GET コマンドの要求ファイル(ファイル N)をキャッシュしていない場合、ネットワークアプライアンスは、クラウド DC 側のサーバ端末に接続して新規 TCP コネクションを生成して、ファイル N を TCP コネクションを用いてダウンロードして、ファイルデータテーブルに記録する。更に、ファイルポインタテーブルに、ファイル N の名前と、ファイル N をキャッシュしているアドレスポインタを記録し、蓄積完了とする(図 109)。

ファイルデータテーブルが GET コマンドの要求ファイル(ファイル N)をキャッシュしている場合は、ファイルテーブルにキャッシュ済みのファイル N を、コマンドを送信してきたアクセス側端末に返信する。

<u>データベースに対するデータ選択/登録/削除/更新</u>

ネットワークアプライアンスの試作機は、クラウド DC 側のサーバ端末のデータベースのエントリをキャッシュするためのデータベーステーブル(図 110)と、エントリが含むアイテムへのアドレスポインタを記録するポインタテーブル(図 110)を、外付けメモリ(図 107 の

2)に備える。

HTTP プロトコル上で、最大 3 つのアイテムデータからなるエントリの登録/削除/更新/選択を要求する POST コマンド付パケットがネットワークアプライアンスに到着する(図 110)。ネットワークアプライアンスは、ポインタテーブルがアイテムに対するポインタをキャッシュしているかどうかに基づいて、データベーステーブルのエントリとポインタテーブルのポインタに対して、アイテムの登録/削除/更新/選択を行う。更に、SQL トランザクションの結果を含むパケットをアクセス側の端末へ返信する。

HTTP プロトコル上で、最大 3 つのアイテムデータからなるエントリの登録を要求する POST コマンド付パケットがネットワークアプライアンスに到着すると(図 110)、ネットワークアプライアンスの動的再構成 LSI はアイテムデータを用いてポインタテーブルを検索して、アイテムデータを持つエントリがデータベーステーブルに登録済みかを判定する。未登録の場合は、3 つのアイテムデータそれぞれに対して同一のポインタを生成して、ポインタテーブルに登録する。更に、データベーステーブル 1 に登録要求のあったエントリを登録する(図 110)。

HTTP プロトコル上で、特定のアイテムデータを持つエントリの削除を要求する POST コマンド付パケットがネットワークアプライアンスに到着すると、削除要求のあったアイテムデータを含むエントリを用いて生成したポインタと、削除要求のあったアイテムデータを持つエントリを、ポインタテーブルおよびデータベーステーブルから削除する。

HTTP プロトコル上で、特定のアイテムデータを持つエントリの更新を要求する POST コマンド付パケットがアプライアンスに到着すると、更新要求のあったアイテムデータを含むエントリを用いて生成したポインタと、更新要求のあったアイテムデータを持つエントリを、ポインタテーブルおよびデータベーステーブルから削除する。更に、新たな 3 つのアイテムデータを用いて生成したポインタと、新たな 3 つのアイテムデータからなるエントリを、ポインタテーブルおよびデータベーステーブルに挿入する。

HTTP プロトコル上で、特定のアイテムデータを持つエントリの選択を要求する POST コマンド付パケットがアプライアンスに到着すると、選択要求のあったアイテムデータを持つエントリを、ポインタテーブルおよびデータベーステーブルから選択する。

キャッシュした DB 内容をサーバへアップロード

ネットワークアプライアンスの試作機は、キャッシュしたデータベースの全エントリ内容を、クラウド DC 側サーバ端末へアップロードするよう要求する HTTP GET コマンド付パケットを受信すると、クラウド DC 側サーバ向けに SYN パケットと、クライアント向けにアップロードを開始したことを通知するパケットを送信する。その後、クラウド DC 側サーバから SYN-ACK パケットが到着すると、ネットワークアプライアンス内の外付けメモリにキャッシュしたデータベーステーブルの全エントリ内容をサーバへアップロードする。サーバは、アップロードしたデータを用いて、データベースの内容を更新する(図 111)。

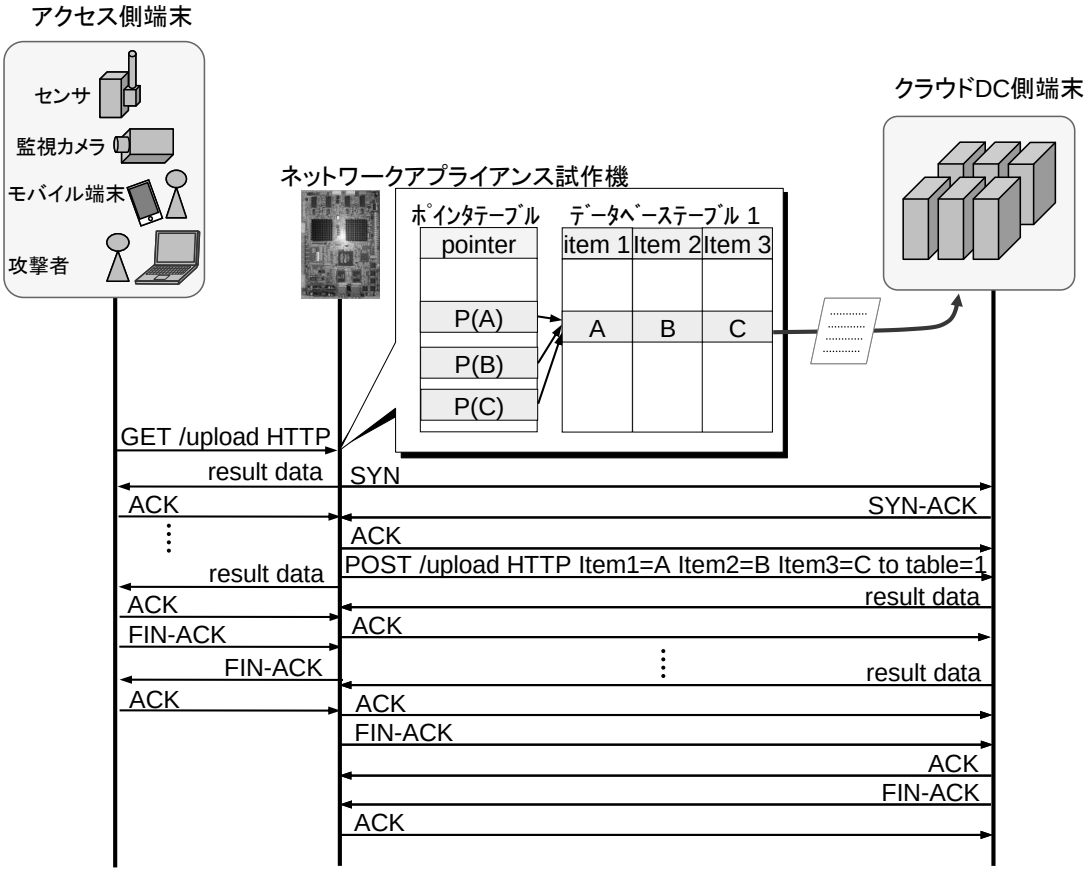


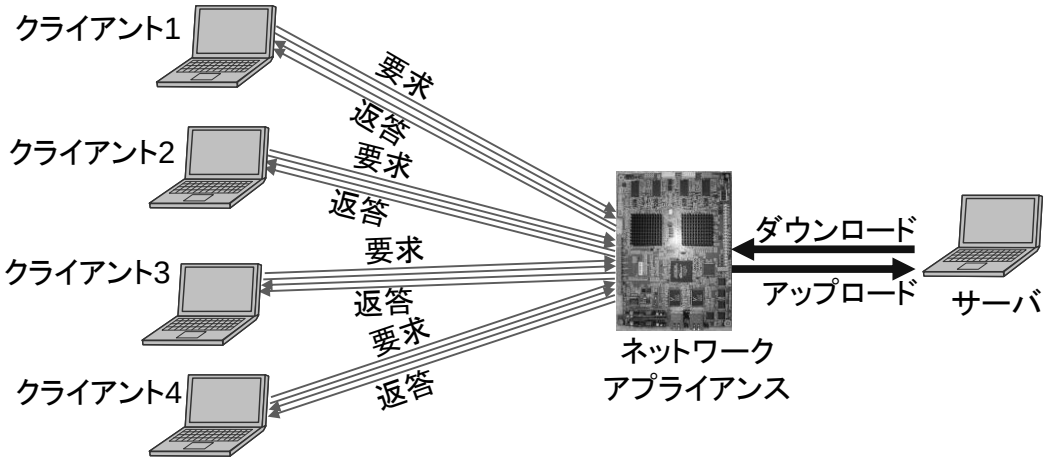
図 111　DB データのサーバへのアップロード

5.5.3　実験評価

クラウドサービス高速化を行うネットワークアプライアンスの試作機に対して、DDoS 攻撃を行う実験や、HTTP コマンド上に記載した SQL コマンドを継続的に送信して、データベースのトランザクションを実行させる実験を行った。1 秒間あたりに実行可能なトランザクション数や機能切替頻度、周波数から見積もった演算性能などを評価した。実験評価の構成を図 112 に示す。

最初に、セキュリティ機能の試験を実施した。クライアント 2-4 からサーバに向けて 100k パケット/秒にて SYN フラッド DDoS 攻撃を行っている状態で、クライアント 1 がサーバファイルを円滑にダウンロードできることや、データベースに対するデータ登録/削除/更新/選択を正常に実行できることを確認した。

次に、クラウドサービス高速化機能の試験を実施した。クライアント 1～4 からサーバに向けて、最大 8 つの TCP コネクションを確立して、データベースに対してアイテムデータの登録，削除，更新，選択のトランザクション(5.5.2 節③記載)をこの順番で、各コネクション上で実行し続けた時の 1 秒間あたりのトランザクション実行回数(TPS: Transaction Per Second)を評価した。ネットワークアプライアンスの試作機の機能を無効にして、サーバが



サーバ	HTTPD とDBを実行 (汎用プロセッサ 2.0 GHz 21W, 1-Gbps LAN)
クライアント1-4	DDoS攻撃や、各種トランザクション要求の送信を実行 (汎用プロセッサ 2.0 GHz 21W, 1-Gbps LAN)
ネットワーク アプライアンス	L4-7 層の機能を実行 (5.2節の8つの機能) (動的再構成LSI 166 MHz 7W, 1-Gbps LAN x 2)

図 112　実験構成

クライアント要求を直接処理した時と、ネットワークアプライアンスの試作機の機能を有効にして、ネットワークアプライアンスがクライアント要求を処理したときのトランザクションの実行性能の比較結果を表 11 に示す。

ネットワークアプライアンスの試作機は、各種トランザクションを実行する全ての実験において、サーバの 10 倍以上の演算性能を、サーバの 1/3 倍の消費電力を追加することで実現することを確認した。

表 12 には、ネットワークアプライアンスが各トランザクションを実行する際の毎秒の再構成回数を、再構成の種別毎に示す。

データベースに対してデータの登録を行うトランザクションのみ、あるいは削除を行うトランザクションのみを実行し続ける評価では、構成キャッシュサイズと同じ 3 つの構成データを用いるため、キャッシュヒット率は 100%となる。トランザクション性能は 22500TPS, 24600TPS となりサーバ単独のケースと比較して 73～332 倍へと大幅に向上した。

データベースに対してデータの挿入を行うトランザクションのみ、あるいは更新を行うトランザクションのみを実行し続ける評価では、構成キャッシュサイズを上回る 4 つの構成データを用いる。構成データのうち 1 つはパケット受信時に常にヒットして自律再構成により用いられ、1 つは TCP コネクションの確立時以外は用いられず、2 つは常にミスヒットとなりプログラム再構成する必要がある。ヒットする構成データ 1 つと、ミスヒットする構成データ 2 つを繰り返し利用するため、キャッシュヒット率は 33%と低くなる。トラ

表 11　各トランザクションの実行性能

	トランザクション					
	100 kpps DDoS	選択	削除	挿入	更新	選択/削除 挿入/更新
サーバ(21 W)	機能停止	310 TPS	74 TPS	117 TPS	72 TPS	128 TPS
ﾈｯﾄﾜｰｸｱﾌﾟﾗｲｱﾝｽの試作機追加(7 W)	防御	22500 TPS	24600 TPS	1556 TPS	1559 TPS	2273 TPS
改善率	+防御	X73	X332	X13	X22	X18
改善率/ 1 W	+防御	X219	X996	X39	X66	X54

表 12　各トランザクションの毎秒の再構成回数

	トランザクション					
	100 kpps DDoS	選択	削除	挿入	更新	選択/削除 挿入/更新
自律再構成	200000	22500	24600	1556	1559	2273
プログラム再構成 キャッシュヒット有	0	22500	24600	0	0	284
プログラム再構成 キャッシュヒット無	0	0	0	3112	3118	3125
全再構成回数	200000	45000	49200	4668	4677	5682
キャッシュヒット率	100%	100%	100%	33%	33%	40%

ンザクション性能は1556TPS, 1559TPS となりサーバ単独のケースと比較して13～22倍へと向上したが、登録や削除を行うトランザクションと比較して高速化効果は一桁小さい。

データベースに対してデータの登録・削除・更新・選択を順番に行うトランザクション処理では、構成キャッシュサイズを大幅に上回る 8 つの構成データを用いる。構成データのうち 1 つはパケット受信時に常にヒットして自律再構成により用いられ、1 つは TCP コネクション生成時以外は用いられず、6 つは常にミスヒットとなりプログラム再構成する必要がある。ヒットする構成データ 1 つと、ミスヒットする構成データ 1～2 つを繰り返し利用するため、キャッシュヒット率は 40%と低くなる。1 秒間に 5682 回、自律再構成の場合で最短 6ns、キャッシュヒット無のプログラム再構成の場合で平均 300μs の応答速度で 1ms 以内での再構成を実現することで、トランザクション性能は 2273TPS となりサーバ単独のケースと比較して 18 倍へと向上した。但し、登録や削除を行うトランザクションと比較して高速化効果は一桁小さい。

各トランザクションの理論性能を、以下の式 5-1 に示す。式 5-1 を用いて計算した理論性能と、実際の実験評価で得られた実験性能の比較を行った。比較結果を図 113 に示す。

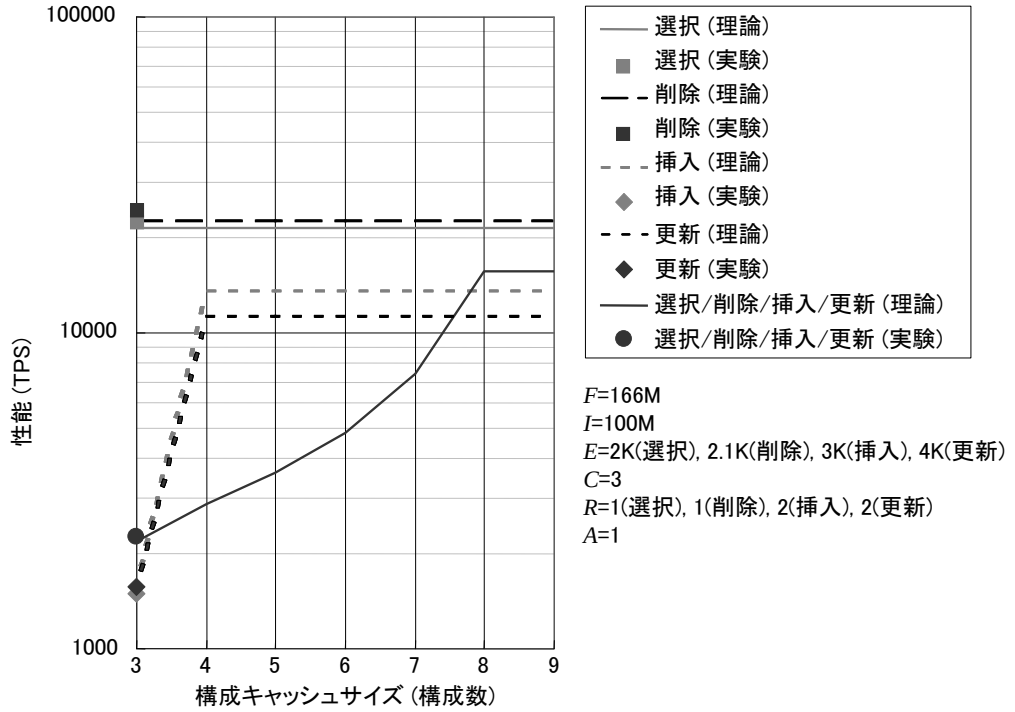


図 113　理論性能と実験性能の比較

$$P=(F-I)/(E+(1000*H(C)+20000*(1-H(C)))*R+A) \quad (5\text{-}1)$$

P: 理論性能
F: プロセッサの毎秒のクロックサイクル
I: OS において再構成関数呼出に用いる毎秒のクロックサイクル
E: トランザクション実行に用いるクロックサイクル
C: 構成データキャッシュのサイズ
H(*C*): プログラム再構成の時のキャッシュヒット率
(*C* の値とキャッシュアルゴリズムに依存)
R: トランザクション毎のプログラム再構成数
A: トランザクション毎の自律再構成数

実験性能は、理論性能とよく一致した。構成キャッシュサイズが 3 のときについて、式 5-1 は正しいと推測される。理論性能からは、使用する構成データ数が構成データキャッシュサイズ以下であり、キャッシュヒット率が 100%となるときに、性能が最大化すると予測される。従って、データの選択，削除，挿入，更新をこの順番で繰り返し行うトランザクションの性能は、構成データキャッシュサイズが 8(使用する構成データ数と同じサイズ)となれば、性能を最大化できると予想される。

5.6 結言

本研究では、遠隔地のセンサデバイスや端末からクラウドにアクセスする際のコマンド往復時間が長い場合に、クラウドサービスの応答時間が遅くなるという課題を解決するため、図 93 に示すように、クラウドサービスを肩代わりするネットワークアプライアンスを、ネットワークエッジに分散配置して、クラウドサービスの応答時間を高速化することを提案した。

提案したネットワークアプライアンスは、遠隔地のセンサデバイス近くのネットワークエッジに設置するユースケースが想定されるため、設置場所を選ばないよう小型省電力な装置として実現する必要がある。また、多数センサデバイスからの大量の通信データの収集・解析を可能とする高い演算性能が求められる。更に、クラウドサービスをアプリレベルで代替する性格上、クラウドサービスを代行して応答を高速化するクラウドサービス高速化系の機能(2.5 節)だけでなく、公衆網からの攻撃トラフィックを排除するセキュリティ系の機能(2.3 節)も備える必要がある。

上記の要件を満たすネットワークアプライアンスには、小型省電力にてパケットを高速演算する性能と共に、ユーザやセンサデバイス毎に提供するクラウドサービス高速化系やセキュリティ系の機能を高速応答にて切替える柔軟性が要求される。

本研究では、提案するネットワークアプライアンス向けプロセッサとして、省電力にてパケットを高速演算する性能と、多様なネットワークアプライアンスの機能を高速応答にて切替える柔軟性の両方を備える動的再構成 LSI を用いた。更に、以下 3 つの提案方式を用いてクラウドサービス高速化ネットワークアプライアンスを試作した。

- メモリ非経由のダイレクトパケット I/O
- プロセッサの階層接続
- 通信状態に基づくパケット毎自己再構成

ネットワークアプライアンスの試作機は、クラウドサービス高速化系の機能とセキュリティ系の機能の一部の機能 (5.5.2 節記載の機能①～④)を実装した。更に、提案するネットワークアプライアンスをアクセス側端末の近いところに設定して、4 種類の SQL を用いた DB 向けクエリトランザクションを順に継続的に実行する実験評価を実施した。提案するネットワークアプライアンスを追加利用することで、消費電力の増加をサーバ消費電力の 3 分の 1 に抑えつつ、2GHz の汎用 CPU を用いるサーバと比較して 18 倍の演算性能を達成することを確認した。また、提案するネットワークアプライアンスは、1ms 以内(6ns～300μs)の高速応答な機能切り替えが可能であった。

以上により、クラウドサービスの一部をネットワークエッジのネットワークアプライアンスで代行する提案方式を用いることで、遠隔地からクラウドサービスを利用する際の応答時間を高速化できることを示した。

第 6 章　結論

クラウドのリソースは、遠距離にある拠点やモバイル端末から、インターネットや仮想専用線や無線網などのネットワークを経由して、主に TCP を用いて通信することで利用する。そのため、クラウドへのネットワーク経由での安全なアクセスをサポートするセキュリティ系のネットワークアプライアンスと、クラウドサービスを利用する際の速度を向上させる通信高速化系やクラウドサービス高速化系のネットワークアプライアンスの重要性が増加している。

セキュリティ系のネットワークアプライアンスは、多数ユーザを収容するクラウドや、クラウドが接続するネットワークを防御するための高スループットを実現しつつ、日々進化する多様な攻撃手法に対応しなければならないという課題があった。通信高速化系のネットワークアプライアンスは、多様なアプリケーションの通信に対応する高い汎用性を実現しつつ、通常の固定式の PC 端末だけではなく、スマートフォンやタブレット PC などのモバイル端末や、設備監視装置や車載端末などのセンサデバイスなど、多様な端末に対する通信高速化をサポートしなければならないという課題があった。特に、モバイル端末やセンサデバイス向けの通信高速化をサポートするためには、ビットエラーによるパケット廃棄により通信速度が低下しやすいラストワンマイルの無線回線区間を高速化せねばならないという課題があった。クラウドサービス高速化系のネットワークアプライアンス③は、遠隔地からクラウドにアクセスする際の往復通信時間が長く、遠隔地の端末からコマンド要求を送信してクラウドからリプライ応答が帰ってくるまでのコマンド往復時間が長い場合に、サービスの応答時間を高速化しなければならないという課題があった。

本研究では、これらの問題を解決するため、以下(1)〜(3)の特徴を持つネットワークアプライアンスを提案および開発し、クラウドデータセンタやネットワークエッジに設置することを提案した。

(1)高スループットを実現しつつ、様々なセキュリティ機能を無瞬断で更新
(2)多様なアプリケーションと端末に対応する TCP 通信のままでの汎用的な通信高速化
(3)ネットワークエッジにおいて様々なクラウドサービスを代行して応答を高速化

本研究で提案および開発した上記(1)〜(3)の特徴を持つネットワークアプライアンスを適宜選択して適用することで、クラウドサービスへの安全なアクセスをサポートしつつ、クラウドサービスを利用する際の速度を向上させることが可能となる。本研究で提案および開発したネットワークアプライアンスの特徴は以下の通りである。

(1)高スループットを実現しつつ、様々なセキュリティ機能を無瞬断で更新

第 3 章に記載の研究では、以下の 3 つの手法を用いることで、様々な異常通信の高速探

知と、高スループットを維持しつつ無瞬断でのアルゴリズム更新を可能とする対異常通信防御機能を提案した。

- 高速演算性能(83Mpps)を持つ動的再構成 LSI を用いたパケット全数解析方式
- 3 種類の異常通信(DDoS，ワーム，P2P)を少ない演算リソースで防止する一括防御方式
- パケット廃棄やスループット劣化の無いアルゴリズム無瞬断更新方式

提案するパケット全数解析方式と一括防御方式とアルゴリズム無瞬断更新方式を用いて、3 種類の異常通信(DDoS、ワーム、P2P)を探知して制御する対異常通信防御装置を設計してシミュレーションを行った。その結果、以下を実現する見通しを得た。

① 83Mpps のスループットにおける 1 秒以内の対異常通信防御
② 従来の特化防御方式と比較して回路量を 24%削減
③ 10Gbps×4 本の帯域利用率 100%でも 1 時間に 24M 回のアップデート可能

対異常通信防御部をデータが通過する時間とアルゴリズム更新時間を合わせた時間が 954ns であるのに対して、フルサイズのパケット到着間隔は 1230ns なので、アルゴリズムを無瞬断で更新可能である。

シミュレーション評価の次に、提案する対異常通信防御機能を、2 つの動的再構成 LSI に搭載した試作機を用いて実験評価を行った。その結果、以下を実現する見通しを得た。

① 対異常通信防御機能を有効にした後、直ちに異常通信を検出して制御
② 標本解析による従来方式よりも 3000 倍高速(0.01 秒)に異常通信を防止
③ 異常通信の帯域占有率が 10^{-3} の時の見逃し率は、従来の標本解析と同等

パケット全数解析による提案方式は、パケット標本解析による従来方式よりも 3000 倍高速(0.01 秒)に異常通信を防止できることを確認した。

以上の研究により、ネットワークアプライアンスの様々な機能をユーザ要求に応じて無瞬断再構成する提案方式を用いることで、高スループットを維持したまま、日々進化する多様な攻撃手法に対応できることを確認した。

(2)多様なアプリケーションと端末に対応する TCP 通信のままでの汎用的な通信高速化

第 4 章に記載の研究では、通信遅延が大きく、廃棄率の大きいネットワークにおいて、TCP のスループットが劣化する問題を解決するために、以下 3 つの特徴を持つ TCP 通信高速化技術 RADIC-TCP を提案した。

① パケット廃棄率の変化率に基づく輻輳制御
② トークンサイズを用いた送信量制御
③ NACK を用いた再送制御

RADIC-TCP は、パケット廃棄率の変化率を用いてネットワークの輻輳を検出して、送信帯域を残存帯域に追従するよう動的に制御することで、一時的なパケット廃棄の影響を抑えて WAN やインターネットの帯域を有効活用する。

更に、パブリッククラウドを利用する際の、クラウド利用ユーザの端末向けの TCP 通信を高速化するために、RADIC-TCP と SACK (Selective ACKnowledgement)[130]を併用することで、データセンタ側に設置するだけで、モバイル端末をはじめとする多様な端末向けの通信を高速化可能な非対向設置による SACK 併用 RADIC-TCP を提案した。

非対向設置による SACK 併用 RADIC-TCP は、送信側に設置するだけで、受信側拠点への装置設置や、受信端末へのソフトウェアのインストールを必要とせずに、不特定多数の端末への送信方向の TCP 通信を高速化可能である。

実験評価では、標準 TCP を RADIC-TCP へ変換するプロトコルコンバータを用いて、RADIC-TCP のスループットを評価した。その結果、以下を確認した。

・残存帯域が 1 秒毎に変化する環境において残存帯域に高速に追従
・ボトルネックのキュー長に依存しないスループットを実現
・RTT 200ms, パケット廃棄率 0.1%以上で、従来 TCP 比 10 倍のスループットを実現
・パケット廃棄発生環境でウィンドウサイズに応じて RADIC-TCP のスループット向上
・提案 TCP は、FTP だけでなく CIFS のスループットも向上
・上限帯域が 10Mbps の時に、複数のコネクション間で公平に帯域を分割して利用

更に、実際の無線ネットワーク回線を用いて、実ネットワーク環境での高速化効果を評価した。非対向設置による SACK 併用 RADIC-TCP を用いた TCP 通信高速化ネットワークアプライアンスの試作機を用いて、実際の無線ネットワークを経由して、性能に制約があるスマートフォン向けデータ転送実験を行い、高速化効果を実証した。その結果、以下を確認した。

・RTT 40ms の LTE 回線にて、実験環境同様の約 3 倍の高速化
・3G 回線や WiFi 回線、SINET 環境、混雑環境においても、1.2～3.0 倍の高速化

以上により、クラウドデータセンタへの非対向設置形態も可能な、多様な端末に対応する汎用性の高い TCP 通信高速化方式を用いることで、ユーザが利用する多様なアプリケーションと端末に対応する汎用性の高い通信高速化を実現できることを示した。

(3)ネットワークエッジにおいて様々なクラウドサービスを代行して応答を高速化

第 5 章に記載の研究では、遠隔地のセンサデバイスや端末からクラウドにアクセスする際のクラウドサービスの応答時間が遅くなる課題を解決するため、クラウドサービスを肩代わりするネットワークアプライアンスをネットワークエッジに分散配置して、クラウドサービスの応答時間を高速化することを提案した。

提案したネットワークアプライアンスは、遠隔地のセンサデバイス近くのネットワークエッジに設置するユースケースが想定されるため、設置場所を選ばないよう小型省電力な装置として実現する必要がある。また、多数センサデバイスからの大量の通信データの収集・解析を可能とする高い演算性能が求められる。更に、クラウドサービスをアプリレベルで代替する性格上、クラウドサービスを代行して応答を高速化するクラウドサービス高速化系の機能(2.5 節)だけでなく、公衆網からの攻撃トラフィックを排除するセキュリティ系の機能(2.3 節)も備える必要もある。

上記の要件を満たすネットワークアプライアンスには、小型省電力にてパケットを高速演算する性能と共に、ユーザやセンサデバイス毎に提供するクラウドサービス高速化系やセキュリティ系の機能を高速応答にて切替える柔軟性が要求される。

本研究では、提案するネットワークアプライアンス向けプロセッサとして、省電力にてパケットを高速演算する性能と、多様なネットワークアプライアンスの機能を高速応答にて切替える柔軟性の両方を備える動的再構成 LSI を用いた。更に、以下の 3 つの提案方式を用いてクラウドサービス高速化用のネットワークアプライアンスを試作した。

- メモリ非経由のダイレクトパケット I/O
- プロセッサの階層接続
- 通信状態に基づくパケット毎自己再構成

ネットワークアプライアンスの試作機は、クラウドサービス高速化系の機能とセキュリティ系の機能の一部の機能 (5.5.2 節記載の機能①～④)を実装した。更に、提案するネットワークアプライアンスをアクセス側端末の近いところに設定して、4 種類の SQL を用いた DB 向けクエリトランザクションを順に継続的に実行する実験評価を実施した。提案するネットワークアプライアンスを追加利用することで、消費電力の増加をサーバ消費電力の 3 分の 1 に抑えつつ、2GHz の汎用 CPU を用いるサーバと比較して 18 倍の演算性能を達成することを確認した。また、提案するネットワークアプライアンスは、1ms 以内(6ns～300μs)の高速応答な機能切り替えが可能であった。

以上により、クラウドサービスの一部をネットワークエッジのネットワークアプライアンスで代行する提案方式を用いることで、遠隔地からクラウドサービスを利用する際の応答時間を高速化できることを示した。

これらの(1)〜(3)の研究成果により、クラウドサービスの通信品質を向上させるネットワークアプライアンスの要素技術を確立することができたと考える。ネットワーク経由で何時でも何処でも利用可能なクラウドサービスの重要度は増しており、クラウドサービスを利用する多様な ICT 機器や多数のセンサデバイスを収容しつつあるネットワークは、社会インフラの一部として高度な機能をサポートしていく必要があると考えている。本研究の成果が、ネットワークの高度化に少しでも役に立てば幸いである。

謝辞

本論文は筆者が筑波大学大学院システム情報工学研究科リスク工学専攻博士後期課程に在籍中に研究成果をまとめたものである。本研究を進めるにあたり同専攻教授 吉田健一先生には指導教員として、終始有益なご指導とご助言を頂いた。ここに感謝の意を表する。

本研究を遂行するにあたり、同専攻教授 津田和彦先生、並びに、同専攻准教授 倉橋節也先生にはご助言、ご指導を頂いた。ここに感謝の意を表する。また、発表会の場などで、有益なアドバイスやコメントをくださった同専攻教員の方々に感謝の意を表する。吉田研究室の各位には日頃より有益なご意見をいただいた。ここに感謝の意を表する。

本学の博士課程入学をサポートいただき、入学後も研究と仕事の両面からご指導いただいた(株)日立製作所 中央研究所の西村信治氏、坂本健一氏、矢崎武己氏、對馬雄次氏に心からお礼申し上げる。また、SINET における実験評価を進めるにあたり、(株)日立製作所 情報通信ネットワーク事業部の各位にご協力を賜った。ここに感謝の意を表す。

参考文献

[1] 平岩賢志, 桝川博史, 西村信治, “クラウドコンピューティングを支えるネットワークへの取り組み”, 日立評論, vol. 92, no. 05, pp. 352 - 357, 2010 年 5 月

[2] 田中智佳子, 江崎尚, 内山靖弘, 湯本一磨, 矢崎武己, “クラウドコンピューティングを支えるネットワークの取り組み”, 日立評論, vol. 93, no. 07, pp. 475 - 483, 2011 年 07 月

[3] 飯塚孝好, 佐瀬信之, 千葉望, 加瀬奈月, 廣喜充, 今村祐一, “企業基幹情報システムの進化を支える Harmonious Cloud”, 日立評論, vol. 93, no. 07, pp. 462 - 467, 2011 年 07 月

[4] 矢崎武己, 磯部隆史, “クラウドコンピューティング向けネットワーク高速化技術” ,日立評論, vol. 95, no. 6 - 7, pp. 424 - 431, 2013 年 07 月

[5] 西村信治, 柴田治朗, 鈴木政朗, 矢崎武己, “スマートな社会, ビジネスを支えるネットワーク”, 日立評論, vol. 94, no. 10, pp. 724 - 729, 2012 年 10 月

[6] G. Vigna and R. A. Kemmerer, "Netstat: a network-based intrusion detection approach," Proc. 14th An. Comp. Sec. App. Conf., 1998, pp. 25 - 34

[7] C. Manikopoulos and S. Papavassiliou, "Network intrusion and fault detection: a statistical anomaly approach," IEEE Commun. Magn., Oct. 2002, vol. 40, no. 10, pp. 76 - 82

[8] J. Li and C. Manikopoulos, "Early statistical anomaly intrusion detection of dos attacks using MIB traffic parameters," In Proceedings of the 2003 IEEE Systems, Information Assurance Workshop, Man and Cybernetics Society, June 2003, pp. 53 - 59

[9] Y. Ohsita, S. Ata, M. Murata, T. Murase, "Detecting distributed denial-of-service attacks by analyzing TCP SYN packets statistically," Proceedings of IEEE Globecom 2004, Nov. 2004

[10] J. Mirkovic, G. Prier; P. Reiher, "Attacking DDoS at the source," Network Protocols, 2002. Proceedings. 10th IEEE International Conference, pp. 312 - 321, Nov. 2002

[11] X. Zhang, C. Li, W. Zheng, "Intrusion prevention system design," IEEE, Computer and Information Technology (CIT) 2004, The Fourth International Conference, pp. 386 - 390, Sept. 2004

[12] Haining Wang; Danlu Zhang; Shin, K.G., "Detecting SYN flooding attacks," INFOCOM 2002, Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , vol. 3, pp. 1530 - 1539, 23 - 27 June 2002

[13] 原田薫明 他, “インターネットトラヒックにおける平均ボリュームと周期性の分析”, 信学技報, vol. 108, no. 458, IN2008-168, pp. 213 - 218, 2009 年 3 月

[14] 原田薫明 他, “周期性を考慮した異常トラヒック検知手法”, 信学技報, vol. 107, no. 222, IN2007-59, pp. 93 - 98, 2007 年 9 月

[15] Jackson, A.W.; Sterbenz, J.P.G.; Condell, M.N.; Hain, Regina Rosales, "Active network monitoring and control: the SENCOMM architecture and implementation," DARPA Active NEtworks Conference and Exposition 2002 Proceedings, pp. 379 - 393, 2002

[16] Stanic, S.; Subramaniam, S.S.; Sahin, G.; Choi, H.; Choi, H.-A., "Active monitoring and alarm management for fault localization in transparent all-optical networks," Network and Service Management, IEEE Transactions on , vol. 7, no. 2, pp. 118 - 131, June 2010

[17] Schultz, M.J.; Ben Wun; Crowley, P., "A Passive Network Appliance for Real-Time Network Monitoring," Architectures for Networking and Communications Systems (ANCS), 2011 Seventh ACM/IEEE Symposium, pp. 239 - 249, 3 - 4 Oct. 2011

[18] P. Phaal, S. Panchen, N. McKee, "InMon corporation's sFlow: a method for monitoring traffic in switched and routed networks," RFC3176, Sep. 2001

[19] B. Claise, Ed., "Cisco Systems NetFlow Services Export Version 9," IETF RFC3954, October 2004

[20] 縄田秀一, 小頭秀行, 福元徳広, 横田英俊, “パワースペクトルを用いた通信トラヒックの特徴抽出手法の検討”, 電子情報通信学会 ソサイエティ大会, B-6-76, 2014 年 9 月

[21] Bujlow, T.; Hald, S.L.N.; Riaz, T.; Pedersen, J.M., "A method for evaluation of quality of service in computer networks," Advanced Communication Technology (ICACT), 2013 15th International Conference on , pp. 17 - 25, 27 - 30 Jan. 2013

[22] Bujlow, T.; Riaz, T.; Pedersen, J.M., "A method for assessing quality of service in broadband networks," Advanced Communication Technology (ICACT), 2012 14th International Conference, pp. 826 - 831, 19 - 22 Feb. 2012

[23] Smallwood, D.; Vance, A., "Intrusion analysis with deep packet inspection: Increasing efficiency of packet based investigations," Cloud and Service Computing (CSC), 2011 International Conference, pp. 342 - 347, 12 - 14 Dec. 2011

[24] Zoican, S.; Zoican, R., "Intrusive detection system implementation using deep packet inspection," Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS), 2013 11th International Conference, vol. 2, pp. 413 - 416, 16 - 19 Oct. 2013

[25] Tran Ngoc Thinh; Tran Trung Hieu; Van Quoc Dung; Kittitornkun, S., "A FPGA-based deep packet inspection engine for Network Intrusion Detection System," Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2012 9th International Conference, pp. 1 - 4, 16 - 18 May 2012

[26] 河野伸也, 久保庭章子, 宍戸　豪, “アプリケーション識別制御基盤におけるフロー情報同期方式の提案”, 電子情報通信学会 ソサイエティ大会, B-6-36, 2014 年 9 月

[27] 宗宮 利夫, 西永 望, “DPI を用いたネットワークからの情報抽出手法の一検討とその試作”, 電子情報通信学会 NS 研究会, 信学技報, vol. 111, no. 196, pp. 17 - 22, 2011 年 8 月

[28] Weirong Jiang; Gokhale, M., "Real-Time Classification of Multimedia Traffic Using FPGA," Field Programmable Logic and Applications (FPL) 2010 International Conference, pp. 56 - 63, Aug. 31 2010 - Sept. 2 2010

[29] Xin-Yu Sun; Shou-Huai Wang, "Application of deep packet inspection in peer-to-peer traffic identification," Consumer Electronics, Communications and Networks (CECNet) 2011 International Conference, pp. 2963 - 2965, 16 - 18 April 2011

[30] Piyachon, P.; Yan Luo, "Efficient memory utilization on network processors for deep packet inspection," Architecture for Networking and Communications Systems (ANCS) 2006 ACM/IEEE Symposium, pp. 71 - 80, 3 - 5 Dec. 2006

[31] Sarang Dharmapurikar; Praveen Krishnamurthy; Sproull, T.S.; Lockwood, J.W., "Deep packet inspection using parallel bloom filters," Micro, IEEE , vol. 24, no. 1, pp. 52 - 61, Jan. - Feb. 2004

[32] Bellovin, S.M.; Cheswick, W.R., "Network firewalls," IEEE Communications Magazine, vol. 32, no. 9, pp.50 - 57, Sept. 1994

[33] Al-Shaer, E.S.; Hamed, H.H., "Discovery of policy anomalies in distributed firewalls," INFOCOM 2004, Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, vol. 4, pp. 2605 - 2616, 7 - 11 March 2004

[34] Moscola, J.; Lockwood, J.; Loui, R.P.; Pachos, M., "Implementation of a content-scanning module for an Internet firewall," Field-Programmable Custom Computing Machines (FCCM) 2003, 11th Annual IEEE Symposium , pp. 31 - 38, 9 - 11 April 2003

[35] Hamed, H.; El-Atawy, A.; Al-Shaer, E., "Adaptive Statistical Optimization Techniques for Firewall Packet Filtering," INFOCOM 2006, 25th IEEE International Conference on Computer Communications. Proceedings, pp. 1 - 12, April 2006

[36] M. Katayama, H. Kai, J. Yoshida, M. Inami, H. Yamada, K. Shiomoto, and N. Yamanaka, "A 10Gb/s firewall system for network security in photonic era," IEICE Trans. Commun., vol. E88-B, no. 5, May 2005, pp. 1914 - 1920

[37] Smith, M.; Hunt, R., "Network security using NAT and NAPT," Networks, ICON 2002 10th IEEE International Conference, pp.355 - 360, 2002

[38] Zeng Chuanhuang; Hu Haonan, "Analysis of the NAT-PT Gateway," Computer Science & Service System (CSSS) 2012 International Conference, pp. 46 - 49, 11-13 Aug. 2012

[39] 磯部隆史, 伊藤大輔, 明石　大, 堤　聡, “広帯域通信プロトコル RADIC-TCP の WAN 仮想専用回線への適用”, 電子情報通信学会 IN 研究会, 信学技報, vol. 110, no. 341, IN2010-116, pp. 111 - 116, 2010 年 12 月

[40] M. Allman and et al., "TCP Congestion Control," IETF RFC2581, Apr. 1999

[41] S. Floyd and et al., "The NewReno Modification to TCP's Fast Recovery Algorithm," IETF RFC2582, Apr. 1999

[42] L. Xu and I. Rhee., "CUBIC: A New TCP-Friendly High-Speed TCP Variant," Workshop on Protocols for Fast Long Distance Networks, Feb. 2005

[43] Kun Tan and et al., "A Compound TCP Approach for High-speed and Long Distance Networks," IEEE INFOCOM, April 2006

[44] S. Floyd, "HighSpeed TCP for Large Congestion Windows," IETF RFC3649, Dec. 2003

[45] Jin, C. and et al., "FAST TCP: Motivation, Architecture, Algorithms, Performance," IEEE INFOCOM, March 2004

[46] Rong Xu; Zhiyuan Li; Cheng Wang; Peifeng Ni, "Impact of data compression on energy consumption of wireless-networked handheld devices," Distributed Computing Systems 2003, Proceedings 23rd International Conference, pp. 302 - 311, 19 - 22 May 2003

[47] Rigler, S.; Bishop, W.; Kennings, A., "FPGA-Based Lossless Data Compression using Huffman and LZ77 Algorithms," Electrical and Computer Engineering (CCECE) 2007 Canadian Conference, pp. 1235 - 1238, 22 - 26 April 2007

[48] Qinlu He; Zhanhuai Li; Xiao Zhang, "Data deduplication techniques," Future Information Technology and Management Engineering (FITME), 2010 International Conference on , vol. 1, pp. 430 - 433, 9 - 10 Oct. 2010

[49] Austin Clements, Irfan Ahmad, Murali Vilayannur, and Jinyuan Li., "Decentralized deduplication in san cluster file systems," In Proc. of the USENIX Annual Technical Conference, June 2009

[50] N. T. Spring and D. Wetherall, "A protocol-independent technique for eliminating redundant network traffic," SIGCOMM 2000, pp. 87 - 95, May 2000

[51] Jiansheng Wei; Hong Jiang; Ke Zhou; Dan Feng, "MAD2: A scalable high-throughput exact deduplication approach for network backup services," Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium, pp. 1 - 14, 3 - 7 May 2010

[52] Xiang Zhang; Zhigang Huo; Jie Ma; Dan Meng, "Exploiting Data Deduplication to Accelerate Live Virtual Machine Migration," Cluster Computing (CLUSTER), 2010 IEEE International Conference on , pp. 88 - 96, 20 - 24 Sept. 2010

[53] Bing Zhou; Jiangtao Wen, "Efficient File Communication via Deduplication over Networks with Manifest Feedback," Communications Letters, IEEE , vol. 18, no. 1, pp. 94 - 97, January 2014

[54] Zhang, Yan; Ansari, Nirwan; Wu, Mingquan; Yu, Heather, "On Wide Area Network Optimization," Communications Surveys & Tutorials, IEEE , vol. 14, no. 4, pp. 1090 - 1113, Fourth Quarter 2012

[55] Singh; Amit P., "System and method for incremental and continuous data compression," US2002/0037035, 2001 年 05 月 31 日

[56] McCanne; Steven, "Content-based segmentation scheme for data compression in storage and transmission including hierarchical segment representation," US6667700, 2002 年 10 月 30 日

[57] "Cisco WAAS," http://www.cisco.com/web/JP/product/hs/ contnetw/waassw/index.html, June 2014 access

[58] 磯部隆史, 谷田直輝, “TCP 高速化と併用する高スループット差分転送技術”, 電子情報通信学会 ソサイエティ大会, BS-4, 2014 年 9 月

[59] Sukhwani, B.; Hong Min; Thoennes, M.; Dube, P.; Brezzo, B.; Asaad, S.; Dillenberger, D.E., "Database Analytics: A Reconfigurable-Computing Approach," Micro, IEEE , vol. 34, no. 1, pp. 19 - 29, Jan. - Feb. 2014

[60] B. Sukhwani, et al., "Database Analytics Acceleration Using FPGAs," Proc. 21st International Conference Parallel Architectures and Compilation Techniques, ACM, 2012, pp. 411 - 420

[61] Tao Fu; Mengchi Liu, "A gateway from HTML to XML," International Database Engineering and Applications Symposium (IDEAS) 2004, pp. 205 - 214, 7 - 9 July 2004

[62] Kim, Jangwon; Jeong, Dongwon; Kim, Jinhyung; Baik, Doo-Kwon, "An Algorithm for Extracting Referential Integrity Relations Using Similarity during RDB-to-XML Translation," 2007 International Conference on Computational Intelligence and Security, pp. 157 - 161, 15 - 19 Dec. 2007

[63] Krishnamurthy, R.; Chakaravarthy, V.T.; Kaushik, R.; Naughton, J.F., "Recursive XML schemas, recursive XML queries, and relational storage: XML-to-SQL query translation," 20th International Conference on Data Engineering, pp. 42 - 53, 30 March - 2 April 2004

[64] Sheng-De Wang; Chun-Wei Chen; Pan, M.; Chih-Hao Hsu, "Hardware accelerated XML parsers with well form checkers and abstract classification tables," International Computer Symposium (ICS) 2010, pp. 467 - 473, 16 - 18 Dec. 2010

[65] 高田 智規, 山本 隆二, 西岡 秀一, 阿部 剛仁, 川村 春美, “XML データの包含関係比較における高速化手法の提案”, 情報処理学会研究報告マルチメディア通信と分散処理（DPS）, 2003 巻, 87 号(2003-DPS-114), pp. 1 – 8, 発行年 2003-08-28

[66] Crovella, M.E.; Bestavros, A., "Self-similarity in World Wide Web traffic: evidence and possible causes," IEEE/ACM Transactions on Networking, vol. 5, no. 6, pp. 835 - 846, Dec 1997

[67] Li Fan; Pei Cao; Almeida, J.; Broder, A.Z., "Summary cache: a scalable wide-area Web cache sharing protocol," Networking, IEEE/ACM Transactions, vol. 8, no. 3, pp.281 - 293, Jun 2000

[68] M. Y. M. Chiu and K.-H.A. Yeung, "Partial video sequence caching scheme for VOD systems with heterogeneous clients," IEEE Transactions on Industrial Electronics, vol. 45, no. 1, pp. 44 - 51, Feb 1998

[69] Y. M. Chiu, and K. H. Yeung, "Partial video sequence caching scheme for VOD systems with heterogeneous clients," 13th International Conference on Data Engineering, pp.323 - 332, Apr 1997

[70] Zhiwen Xu, Xiaoxin Guo, Yunjie Pang and Zhengxuan Wang, "The patched algorithm of dynamic cache for streaming media," International Conference on Communications, Circuits and Systems (ICCCAS), vol. 1, pp. 558 - 562, June 2004

[71] T. Isobe and et al., "10 Gbps implementation of TLS/SSL accelerator on FPGA," IEEE 18th International Workshop on Quality of Service (IWQOS), June 2010

[72] Y. Hasegawa, S. Abe, H. Matsutani, H. Amano, K. Anjo, and T. Awashima, "An adaptive cryptographic accelerator for IPsec on dynamically reconfigurable processor," Field-Programmable Technology, Proceedings of IEEE International Conference, Dec. 2005, pp 163 - 170

[73] Richa Garg and Renu Vig, "An Efficient Montgomery Multiplication Algorithm and RSA Cryptographic Processor," IEEE International Conference on Computational Intelligence and Multimedia Applications, vol. 2, pp. 188 - 195, Dec. 2007

[74] S.E. Eldridge and C.D. Walter, "Hardware Implementation of Montgomery's Modular Multiplication Algorithm," IEEE Trans. Computers, vol. 42, no. 6, pp. 693 - 699, June 1993

[75] Hua Li and Jianzhou Li, "A High Performance Sub-Pipelined Architecture for AES," IEEE International Conference on Computer Design: VLSI in Computers and Processors 2005 (ICCD 2005), pp. 491 - 496, Oct. 2005

[76] Panu Hämäläinen, Marko Hännikäinen, Timo Hämäläinen, and Jukka Saarinen, "Hardware Implementation of the RC4 Stream Cipher," IEEE International Symposium on Micro - Nano Mechatronics and Human Science 2003, vol. 3, pp. 1363 - 1366, Dec. 2003

[77] Hoang Anh Tuan, Katsuhiro Yamazaki and Shigeru Oyanagi: "Multistage Pipelining MD5 Implementations on FPGA with Data Forwarding," IPSJ Online Transactions, vol. 2, pp. 15 - 26, 2009

[78] A.P. Kakarountas, G. Theodoridis, T. Laopoulos, and C.E. Goutis, "High-Speed FPGA Implementation of the SHA-1 Hash Function," IEEE Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), pp. 211 - 215, Sept. 2005

[79] M. C. Murphy and F. Golshani, "Supporting video on demand," IEEE 15th Annual International Phoenix Conference on Computers and Communications, pp. 29 - 36, Mar 1996

[80] Lian Shen, Wei Tu and E. Steinbach, "A Flexible Starting Point Based Partial Caching Algorithm for Video on Demand," IEEE International Conference on Multimedia and Expo, pp.76 - 79, July 2007

[81] Wei Tu, E. Steinbach, M. Muhammad, and Xiaoling Li, "Proxy Caching for Video-on-Demand Using Flexible Starting Point Selection," IEEE Transactions on Multimedia, vol. 11, no. 4, pp. 716 - 729, June 2009

[82] Xiaoling Li, Wei Tu and E. Steinbach, "Dynamic segment based proxy caching for Video on Demand," IEEE International Conference on Multimedia and Expo, pp. 1181 - 1184, June - April 2008

[83] M. Muhammad, Wei Tu and E. Steinbach, "Evaluation of segmentbased proxy caching for video on demand," IEEE International Conference on Multimedia and Expo, pp. 1093 - 1096, June-April 2008

[84] T. Isobe, S. Tsutsumi, "Load balancer for high efficient VOD system achieving quick seek," IEEE Access Spaces (ISAS), 2011 1st International Symposium on , pp. 227 - 232, 17 - 19 June 2011

[85] F. Thouin, M. Coates and D. Goodwill, "Video-on-Demand Equipment Allocation," 5th IEEE International Symposium on Network Computing and Applications (NCA) , pp. 103 - 110, July 2006

[86] W.-F. Poon, K.-T. Lo and J. Feng, "Hierarchical network architecture for layered video streaming," 24th International Conference on Distributed Computing Systems Workshops, pp. 164 - 169, March 2004

[87] S.-H.G. Chan, "Operation and cost optimization of a distributed servers architecture for on-demand video services," IEEE Communications Letters, vol. 5, no. 9, pp. 384 - 386, Sep 2001

[88] Chen-Lung Chan, Shih-Yu Huang and Jia-Shung Wang, "Performance Analysis of Proxy Caching for VOD Services With Heterogeneous Clients," IEEE Transactions on Communications, vol. 55, no. 11, pp. 2142 - 2151, Nov. 2007

[89] Y. Guo, S. Sen and D. Towsley, "Prefix caching assisted periodic broadcast for streaming popular videos," IEEE International Conference on Communication (ICC), vol. 4, pp. 2607 - 2612, May 2002

[90] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," IEEE INFOCOM , vol. 3, pp. 1310 - 1319, Mar 1999

[91] Lixin Gao, Zhi-Li Zhang and D. Towsley, "Proxy-assisted techniques for delivering continuous multimedia streams," IEEE/ACM Transactions on Networking, vol. 11, no. 6, pp. 884 - 894, Dec. 2003

[92] K.-M. Ho, W.-F. Poon and K.-T. Lo, "Performance Study of Large-Scale Video Streaming Services in Highly Heterogeneous Environment," IEEE Transactions on Broadcasting, vol. 53, no. 4, pp. 763 - 773, Dec. 2007

[93] Sun-Euy Kim and C. R. Das, "A reliable statistical admission control strategy for interactive video-on-demand servers with interval caching," International Conference on Parallel Processing, pp. 135 - 142, Aug. 2000

[94] Wanneng Shu, Shijue Zheng, Li Gao and Xiong Wang, "A Self-Adjusted Genetic Algorithm Applied to VOD Cluster," International Conference on Communications, Circuits and Systems, vol. 1, pp. 66 - 69, June 2006

[95] Jianhua Sun, Hai Jin, Hao Chen and Zongfen Han, "Server scheduling scheme for asynchronous cluster video server," 17th International Conference on Advanced Information Networking and Applications (AINA), pp. 509 - 512, March 2003

[96] Suneuy Kim and Kim Hang Thi Tran, "Supporting Scalable and Cooperative Interval Caching in a Clustered Video Server," 6th IEEE International Symposium on Network Computing and Applications (NCA), pp. 283 - 286, July 2007

[97] Lin Wujuan, Law Sie Yong and Yong Khai Leong, "A Novel Interval Caching Strategy for Video-on-Demand Systems," 14th IEEE International Conference on Networks (ICON), vol. 2, pp. 1 - 5, Sept. 2006

[98] D. K. Madathil, R. B. Thota, P. Paul and Tao Xie, "A static data placement strategy towards perfect load-balancing for distributed storage clusters," IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 1 - 8, April 2008

[99] Ohhoon Kwon, Yunjung Yoo, Hyokyung Bahn and Kern Koh, "Caching and Data Allocation for Streaming Service in MEMS-Based Storage," International Conference on Computational Sciences and Its Applications (ICCSA), pp. 132 - 138, June-July 2008

[100] Chentao Wu, Xubin He, Shenggang Wan, Qiang Cao and Changsheng Xie, "Hotspot Prediction and cache in distributed stream-processing storage systems," IEEE 28th International Performance Computing and Communications Conference (IPCCC), pp. 331 - 340, Dec. 2009

[101] E. Kim, and J.C.L. Liu, "Optimizing Prefetch in a Residential Gateway with Networked Storage Systems," IEEE International Conference Multimedia and Expo (ICME), pp. 1050 - 1053, July 2005

[102] P. Sumari, M. Merabti and R. Pereira, "Video-on-demand server: strategies for improving performance," IEEE Proceedings on Software, vol. 146, no. 1, pp. 33 - 37, Feb 1999

[103] P. Jayarekha and A. S. Manjunatha, "A fast start-up and scalable algorithm for continuous media servers," IET-UK International Conference on Information and Communication Technology in Electrical Sciences (ICTES), pp.542 - 545, Dec. 2007

[104] Rahman, M.; Iqbal, S.; Gao, J., "Load Balancer as a Service in Cloud Computing," 2014 IEEE 8th International Symposium on Service Oriented System Engineering (SOSE), pp. 204 - 211, 7 - 11 April 2014

[105] Rahmawan, H.; Gondokaryono, Y.S., "The simulation of static load balancing algorithms," Electrical Engineering and Informatics, 2009. ICEEI '09. International Conference on , vol. 2, pp. 640 - 645, 5 - 7 Aug. 2009

[106] Cardellini, V.; Colajanni, M.; Yu, P.S., "Dynamic load balancing on Web-server systems," Internet Computing, IEEE , vol. 3, no. 3, pp. 28 - 39, May/Jun 1999

[107] Randles, M.; Lamb, D.; Taleb-Bendiab, A., "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing," 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 551 - 556, 20-23 April 2010

[108] A. Bhadani and S. Chaudhary, "Performance Evaluation of Web Servers using Central Load Balancing Policy for Virtual Machines on Cloud," COMPUTE 2010 Proceedings of the Third Annual ACM Bangalore Conference, Article No 16, January 2010

[109] H. Mehta, P. Kanungo, M. Chandwani, "Decentralized content aware load balancing algorithm for distributed computing environment," International. Conference and Workshop on Emerging Trends and Technology, Feb. 2011

[110] A. Lakhina, M. Crovella, C. Diot, "Diagnosing network-wide traffic anomalies," ACM SIGCOMM 2004, Aug.-Sep. 2004

[111] T. H. Ptacek, T. N. Newsham, "Insertion, evasion, and denial of service: eluding network intrusion detection," Secure Networks, Inc., Jan. 1998, http://www.insecure.org/stf/secnet_ids/secnet_ids.html, Oct. 2014 access

[112] “ダイナミック・リコンフィギュラブル・プロセッサ DAPDNA-2”, http://www.tokyo-keiki.co.jp/hyd/j/products/pdf/cat_dd2_02.pdf, Oct 2014 access

[113] "Rapidly Reconfigurable Chip," http://www.tabula.com/news/M11_Tabula_Reprint.pdf, Oct. 2014 access

[114] T. Sato, H. Watanabe, and K. Shiba, "Implementation of dynamically reconfigurable processor DAPDNA-2," IEEE VLSI-TSA International Symposium, pp 323 - 324, Apr. 2005

[115] T. Sato, "DAPDNA-2: A Dynamically Reconfigurable Processor with 376 32-bit Processing Elements," Hotchips17, Aug. 2005

[116] 磯部隆史, 渡辺義則, 樋口秀光, 相本　毅, 吉田健一, “広域ネットワーク網向け異常通信の探知機能の検討”, 電子情報通信学会 IN 研究会, 信学技報, vol. 105, no. 178, IN2005-47, pp. 109 - 114, 2005 年 7 月

[117] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, D. Zamboni, "Analysis of a denial of service attack on TCP," Security and Privacy, Proceedings of IEEE Symposium, pp. 208 - 223, May 1997

[118] B. P. Lim, M. S. Uddin, "Statistical-based SYN-flooding detection using programmable network processor," Information Technology and Applications, ICITA 2005. 3rd International Conference, vol. 2, Jul. 2005, pp. 465 - 470

[119] Rocky K. C. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," Communications Magazine, IEEE, vol. 40, issue 10, Oct. 2002, pp. 42 - 51

[120] "Stateful Inspection," http://www.checkpoint.co.jp/data/Stateful_Inspection_WP_200805.pdf, Oct 2014 access

[121] Jon Postel, "Transmission Control Protocol," IETF RFC793, Sep. 1981

[122] "SYN cookie," http://cr.yp.to/syncookies.html, Novenber 2014 access

[123] H. Kai, H. Yamada, "A study of loss-less reconfiguration on packet processing system," Proceeding of the 2003 IEICE Society Conference (in Japanese), B-6-150, Sept. 2003

[124] H. Yamada, H. Kai and J. Yoshida, "Using re-configurable technology in high-speed packet processing," Proc. ICN 2004, Feb. 2004, pp. 628 - 634

[125] "DAPDNA-FW II," http://www.tokyo-keiki.co.jp/hyd/j/products/pdf/cat_fw2_02.pdf, Oct. 2014 access

[126] "MAWI Working Group Traffic Archive," http://mawi.wide.ad.jp/mawi/samplepoint-B/2006/, Nov. 2014 access

[127] "Intel® Pentium® M Processor 1.50 GHz," http://ark.intel.com/products/27576/Intel-Pentium-M-Processor-1_50-GHz-1M-Cache-400-MHz-FSB, Novenber 2014 access

[128] 磯部隆史, “動的再構成プロセッサを用いた対異常通信防御システム”, 電子情報通信学会 IN 研究会, 信学技報, vol. 105, no. 628, IN2005-227, pp. 419 - 424, 2006 年 3 月

[129] T. Isobe et al., "RADIC-TCP: High-Speed Protocol Applied for Virtual Private WAN," IEEE ICT, pp. 544 - 549, May. 2011

[130] M. Mathis and et al., "TCP Selective Acknowledgment Options," IETF RFC2018, Oct. 1996

[131] H. Chaoxin et al., "WiTracer: A novel solution to improve TCP performance over Wireless network," IEEE IWCMC, pp. 450 - 455, July 2013

[132] J. Kellokoski, "Real-life multipath TCP based make-before-break vertical handover," IEEE ISCC, pp. 252 - 256, July 2013

[133] Kelly, T., "Scalable TCP: Improving Performance in High Speed Wide Area Networks," ACM SIGCOMM, April 2003

[134] Xu, L., Harfoush, K., and Rhee, I., "Binary Increase Congestion Control for Fast Long-Distance Networks," IEEE INFOCOM, March 2004

[135] Y. Gu, X. Hong and et al., "SABUL: A High Performance Data Transport Protocol," IEEE Communications Letters, 2003

[136] Lawrence S. Brakmo, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," IEEE Journal on Selected Areas in Communications, vol. 13, No. 8, Oct. 1995

[137] C. Caini and R. Firrincieli, "TCP Hybla: A TCP Enhancement for Heterogeneous Networks," International Journal of Satellite Communications and Networking, vol. 22, pp. 547 - 566, September 2004

[138] S. Mascolo and et al., "TCP westwood: Bandwidth estimation for enhanced transport over wireless links," MobiCom 2001 Proceedings of the 7th annual international conference on Mobile computing and networking, pp. 287 - 297, July 2001

[139] H. Obata and et al., "A new TCP congestion control method considering adaptability over satellite Internet," Distributed Computing Systems Workshops, 25th IEEE International Conference on , pp. 75 - 81, June 2005

[140] 石川有一 他, “省電力ルータ評価向けモデルトラフィック発生方式の提案”, 電子情報通信学会 総合大会, BS-3-3, 2013 年 3 月

[141] 日高稔 他, “ルータの省電力化に向けた転送性能予測技術の検討”, 信学技報, vol. 110, no. 69, pp. 51 - 56, 2010 年 6 月

[142] V. Jacobson and M. Karels, "Congestion avoidance and control," ACM Comput. Commun. Rev., vol. 18, no. 4, pp. 314 - 329, Aug. 1990

[143] "HTB," http://luxik.cdi.cz/~devik/qos/htb/, Oct 2014 access

[144] "PSPacer," http://www.gridmpi.org/index.jsp, Oct 2014 access

[145] E. Blanton and et al., "A Conservative Selective Acknowledgment (SACK)-based Loss Recovery Algorithm for TCP," IETF RFC3517, April 2003

[146] V. Jacobson and et al, "TCP Extensions for High Performance," IETF RFC1323, May 1992

[147] Dina Katabi and et al., "Internet Congestion Control for High Bandwidth-Delay Product Networks," ACM SIGCOMM, Aug. 2002

[148] "GridFTP," http://globus.org/toolkit/docs/3.2/gridftp/, Oct 2014 access

[149] W. Allcock and et al, "The Globus Striped GridFTP Framework and Server," ACM/IEEE SC 2005 Conference, pp. 54, Nov. 2005

[150] 下見淳一郎, 他, “長距離 TCP 高速化機構の開発”, インターネットコンファレンス 2004, http://www.internetconference.org/ic2004/program.html, Oct. 2014 access

[151] 磯部隆史, 伊藤大輔, 明石 大, 堤 聡, “WAN 仮想専用回線向け広帯域通信 TCP”, 電子情報通信学会 IN 研究会, 信学技報, vol. 112, no. 230, IN2012-100, pp. 139-144, 2012 年 10 月

[152] Takasih ISOBE, Naoki TANIDA, Yuji OISHI, Kenichi YOSHIDA, "TCP Acceleration Technology for Cloud Computing:Algorithm, Performance Evaluation in Real Network," IEEE The International Conference on Advanced Technologies for Communications (ATC), pp. 714 - 719, Oct 2014

[153] Mingzhi Feng; Hua Zhang; Dong Wang; Zhigang Sun, "Design and implementation of high-speed token bucket based on FPGA," Information Science and Engineering (ICISE), 2010 2nd International Conference on , pp. 4474 - 4476, 4 - 6 Dec. 2010

[154] R. Fox, "TCP Big Window and Nak Options," IETF RFC1106, June 1989

[155] Macker J.P. and Adamson R.B.; "A TCP Friendly, Rate-Based Mechanism for NACK-Oriented Reliable Multicast Congestion Control," IEEE GLOBECOM, vol. 3, pp. 1620 - 1625, Dec. 2001

[156] Rung-Shiang Cheng and Hui-Tang Lin, "TCP Selective Negative Acknowledgment over IEEE 802.11 Wireless Networks," International Conference on Networking and Services (ICNS), pp. 98 - 98, July 2006

[157] "Docomo LTE/3G," https://www.nttdocomo.co.jp/, March 2014 access

[158] "Intel Core i7 processor," http://www.intel.co.jp/content/www/jp/ja/processors/core/core-i7-processor.html, March 2014 access

[159] "Microsoft OS Windows 7," http://windows.microsoft.com/ja-jp/windows/windows-help#windows=windows-8, March 2014 access

[160] "CentOS," http://www.centos.org/, March 2014 access

[161] "Google Android OS," https://www.google.co.jp/mobile/android/, March 2014 access

[162] "Apple iOS," http://www.apple.com/jp/ios/, March 2014 access

[163] "NTT B flets," https://flets.com/bflets/, March 2014 access

[164] "Plara Internet Access Service," http://www.plala.or.jp/, March 2014 access

[165] "KDDI AU LTE," http://www.au.kddi.com/, March 2014 access

[166] "SINET," http://www.sinet.ad.jp/case/tsukuba2, March 2014 access

[167] J. Yu, W. Wu, X. Chen, H. Hsieh, J. Yang, and F. Balarin, "Assertion-based power/performance analysis of network processor architectures," High-level Design Validation and Test Workshop 9th IEEE International, Nov. 2004, pp. 155 - 160

[168] "Intel Network Processor," http://datasheet.octopart.com/BIXQDR2416AA-Intel-datasheet-11852570.pdf, Novenber 2014 access

[169] "Cavium Octeon Processor," http://www.cavium.com/OCTEON_MIPS64.html, Novenber 2014 access

[170] I. Papaefstathiou, G. Kornaros, and N. Zervos, "Software processing performance in network processors," Design, Automation and Test in Europe Conference and Exhibition, Proceedings vol. 3, Feb. 2004, pp. 186 - 191

[171] R. Hecht, S. Kubisch, A. Herrholtz, and D. Timmermann, "Dynamic reconfiguration with hardwired networks-on-chip on future FPGAs," Field Programmable Logic and Applications, 2005. International Conference, Aug. 2005, pp. 527 - 530

[172] T. Isobe, "Large-throughput anomaly prevention mechanism implemented in dynamic reconfigurable processor," IEICE Trans. Commun., vol. E89-B, no. 9, Sep. 2006, pp. 2440 - 2447

[173] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," IETF RFC2001, Jan. 1997

関連業績リスト

査読付き学術論文

1. Takashi ISOBE, "Large-Throughput Anomaly Prevention Mechanism Implemented in Dynamic Reconfigurable Processor," IEICE TRANSACTIONS on Communications, Special Section on Networking Technologies for Overlay Networks, vol. E89 - B, No. 9, pp. 2440 - 2447, 1 September 2006
2. Takashi ISOBE, "Query-Transaction Acceleration Using a DRP Enabling High-Speed Stateful Packet-by-Packet Self-Reconfiguration," IEICE TRANSACTIONS on Information and Systems, Special Section on Reconfigurable Systems, vol. E90 - D, No. 12, pp. 1905 - 1913, 1 December 2007

査読付き国際会議論文

1. Takasih ISOBE, Naoki TANIDA, Yuji OISHI, Kenichi YOSHIDA, "TCP Acceleration Technology for Cloud Computing:Algorithm, Performance Evaluation in Real Network," IEEE The International Conference on Advanced Technologies for Communications (ATC), pp. 714 - 719, Oct 2014
2. Takashi ISOBE, Daisuke ITO, Dai AKASHI, Satoshi TSUTSUMI, "RADIC-TCP: High-speed protocol applied for virtual private WAN," IEEE Telecommunications (ICT), 2011 18th International Conference on, pp. 505 - 510, 8 - 11 May 2011
3. Takashi ISOBE, "Query-transaction Acceleration based on Stateful Packet-by-packet Self-reconfiguration using a Dynamic Reconfigurable Processor," Information and Computer Elements (ISICE), 1st International Symposium on, pp. 145 - 150, September 2007

学会発表

1. 磯部隆史, 伊藤大輔, 明石　大, 堤　聡, “WAN 仮想専用回線向け広帯域通信 TCP”, 電子情報通信学会 IN 研究会, 信学技報, vol. 112, no. 230, IN2012-100, pp. 139 - 144, 2012 年 10 月
2. 磯部隆史, 伊藤大輔, 明石　大, 堤　聡, “広帯域通信プロトコル RADIC-TCP の WAN 仮想専用回線への適用”, 電子情報通信学会 IN 研究会, 信学技報, vol. 110, no. 341, IN2010-116, pp. 111 - 116, 2010 年 12 月
3. 磯部隆史, “通信状態に基づくパケット毎自己再構成を用いた動的再構成プロセッサ搭

載クエリトランザクション高速化装置”, 電子情報通信学会 RECONF研究会, 信学技報, vol. 107, no. 41, RECONF2007-1, pp. 1 - 6, 2007 年 5 月
4. 磯部隆史, “動的再構成プロセッサを用いた分散ネットワークトランザクション装置の開発”, 電子情報通信学会 CPSY 研究会, 信学技報, vol. 106, no. 386, CPSY2006-37, pp. 19 - 24, 2006 年 11 月
5. 磯部隆史, “動的再構成プロセッサを用いた対異常通信防御システム”, 電子情報通信学会 IN 研究会, 信学技報, vol. 105, no. 628, IN2005-227, pp. 419 - 424, 2006 年 3 月
6. 磯部隆史, 西村信治, “動的再構成プロセッサに搭載した異常通信の探知機能”, 電子情報通信学会 RECONF 研究会, 信学技報, vol. 105, no. 451, RECONF2005-62, pp. 19 - 24, 2005 年 11 月

表彰

1. 主筆特許「端末間の通信を高速化する通信装置および通信システム（特許第 5175982 号)」が、平成 25 年 関東地方発明表彰を受賞
2. 主筆特許をメイン機能として搭載する製品「日立 WAN アクセラレータ GX1000」が、日刊工業新聞社 2012 年「十大新製品賞 日本力(にっぽんぶらんど)賞」を受賞