Vol.2010-OS-113 No.13
2010/1/28

仮想マシンモニタを用いたVPN障害への透過的な対応手法

松橋洋平[†1]　品川高廣[†1]　加藤和彦[†1]

クラウド・SaaSが広く利用されている昨今，インターネットを介したサービスでも高いディペンダビリティを実現したいという要求が高まっている．こうした中，VPNは柔軟且つ安全にクラウドに接続する技術として注目されている一方で，VPNの可用性の向上が課題となっている．本稿では，VPN障害が生じても通信が途切れないシステムをユーザやOSから透過的に提供する手法を提案する．障害対応の基盤として仮想マシンモニタを用いることで，VPN障害の検知からVPN切り替えに至るまでを透過的に提供する．また，VPN切り替えの前後におけるパケットの不整合を解消するため，通信をネットワークレイヤで整合するパケット中継システムを提供する．これらの実装と評価を行い，VPN障害に対し透過的な障害対応を実現することを確認した．

Transparent approaches for fault recovery in VPNs using a virtual machine monitor

Yohei Matsuhashi,[†1] Takahiro Shinagawa[†1]
and Kazuhiko Kato[†1]

Cloud and SaaS are widely used in the Internet and, as a consequence, the growing demands to achieve high dependability in there services. Under these circumstances, VPN is attracting attention as a flexible and safely connect to the Cloud. However, increased availability of VPN is a challenge. In this paper, we propose transparent approaches for fault recovery in VPNs to realizing the VPN communication continue without interruption even if the result of failure. By using a virtual machine monitor for foundation failure, to provide failure detection and rangin to switch transparently. In addition, the packets to resolve the inconsistencies in the before and after switching, packet relay system to provide consistent communication at the network layer. And evaluate these implementation, VPN confirmed that failure to provide transparent support for disabilities.

1. はじめに

クラウドコンピューティング・SaaSが広く利用されている昨今，インターネットを介したサービスでも高いディペンダビリティを実現したいという要求が高まっている．こうした中，クライアントからクラウドに接続する技術としてVPNが注目されている．VPNを用いることで，ネットワークを介したノード間の通信のセキュリティを確保し，柔軟なネットワーク構成が実現される．実際に，近年ではクライアントからクラウドにVPNで接続する形態を取るサービスが登場している[1),2)]．

しかし，VPN障害が生じると，クライアントはクラウドのサービスが利用できなくなってしまう．VPNにおいて生じる障害は二つに大別される．一方はネットワークレイヤに起因する障害であり，クライアントからクラウドまではそもそもネットワークレイヤで到達できない．他方はVPNレイヤ自身に起因する障害であり，双方のノードがネットワークレイヤで到達可能であっても，クラウドのサービスを利用できない．

VPNの可用性向上を図る手法として，VPN機器の多重化がある．この手法は，VPN機器に障害が生じた際に，スタンバイ状態にある他のVPN機器に切り替えるものである．しかし，VPN機器の多重化だけではネットワークレイヤで生じる障害には対応できないほか，VPN切り替えを行うことでサーバがクライアントを一意に識別できなくなり，クライアント・サーバ間の通信が切れてしまう．また，VPN障害の検知やVPN切り替えに係る対応はクライアントのOSやアプリケーションから透過でない．

本稿では，VPN障害が生じても通信を途切らせることなく継続し，OSやアプリケーションに対して透過的な障害対応手法を提案する．まず，ネットワークレイヤで生じる障害に対応するために，クラウドに対し複数のネットワーク経路を用意する．VPN機器で生じる障害に対応するために，クライアントが自律的にVPN切り替えを行う．VPN障害をまたぐ通信を途切らすことなく継続するために，パケット中継システムを提案する．VPN障害時の対応をOSやアプリケーションから透過に提供するために，仮想マシンモニタのレイヤで導入する．

以上の提案の実装を行い，その評価を行った．以下に本稿の構成を示す．2章では本提案

†1 筑波大学 システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba

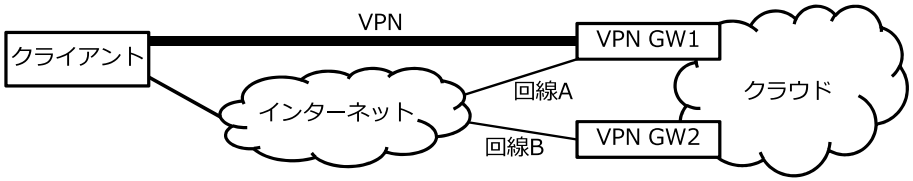

図 1 目指す環境の構成
Fig. 1 Environmental organization aimed

が前提としている環境構成を述べ，3 章では本提案におけるシステムの設計を述べ，4 章では 3 章に基づくシステムの実装を述べ，5 章では実装の評価結果を述べ，6 章では関連研究を述べ，7 章で本稿をまとめる．

2. 目指す環境の構成

本章では，VPN 障害への対応を実現するために目指す環境の構成を述べる．2.1 節では環境の全体構成を示し，2.2 節ではクラウドの構成を示し，2.3 節ではクライアントの構成を示し，2.4 節ではそれらを結ぶネットワークの構成を示す．

2.1 全体の構成

本提案が目指す環境は大まかにクライアントとクラウド，それらを繋ぐネットワークから構成される．その全体構成を図 1 に示す．クラウドはクライアントからのサービスの利用要求に応じてクラウド内のサービスを提供し，クライアントはクラウドのサービスをインターネットを介して利用する．クライアントとクラウドを結ぶネットワーク経路は，通信のセキュリティ確保とクライアントの柔軟なネットワーク構成の実現のため，VPN で仮想化する．クラウドの境界には複数の VPN ゲートウェイを置き，またそれぞれを異なる外部ネットワーク回線に繋ぐことでネットワーク経路に多様性を持たせる．またクライアントは，いずれの VPN ゲートウェイからでも同様にクラウドのサービスを利用できる．

2.2 クラウドの構成

クラウドは，クライアントからの要求に応じてクラウド内のサーバからサービスを提供する．クラウド内部にはサービスを提供するサーバ群を置き，それらをクラウド内ネットワークで結ぶ．クラウド内ネットワークは，クライアントの要求に対し，対応するサービスを提供するサーバまでルーティングを行う．クラウド内ネットワークと外部ネットワークの境界には VPN ゲートウェイを置き，クライアントとの VPN 通信を処理する．また，複数の VPN ゲートウェイをそれぞれ異なる外部ネットワーク回線に繋ぐ．クライアントはいずれの VPN ゲートウェイからでも同様にクラウドのサービスを利用できる．これらに伴うクラウド内のサーバの改変は不要である．

2.3 クライアントの構成

クライアントは，クラウドのサービスをインターネットを介して利用し，クラウドまでのネットワーク経路を VPN で仮想化する．VPN はクラウドの境界に置かれた複数の VPN ゲートウェイのいずれかとの間で確立し，他の VPN ゲートウェイに切り替えても同様にクラウドのサービスの利用を継続できる．また，これらに伴って OS やアプリケーションの改変は不要であり，クライアントで一般的に用いられている Windows 環境においても改変することなく動作する．

2.4 ネットワーク経路の構成

クライアントとクラウドを結ぶネットワーク経路は，不特定多数が利用するインターネットを想定し，インターネットを介しても，安全で柔軟なネットワーク構成を実現する通信環境を得るために，クライアント・クラウド間のネットワーク経路を VPN で仮想化する．クラウドに置く複数の VPN ゲートウェイには，ネットワーク経路を多様化させるため，それぞれ異なる外部ネットワーク回線を接続する．これにより，ネットワークレイヤで起きる障害に対して高可用な基盤を提供する．

3. VPN 障害への透過的な対応手法

本章では，VPN 障害への透過的な対応手法を述べる．透過的な障害対応を実現する手法を大きく二つに分けて述べる．一方は仮想マシンモニタを用いることで実現する透過的な障害対応手法であり，他方は本稿で提示するパケット中継システムを用いることで実現する透過的な障害対応手法である．3.1 章の概要では提案手法の全体像を示し，3.2 章では仮想マシンモニタによる透過的な障害対応手法を示し，3.3 章ではパケット中継システムを用いた透過的な障害対応手法を示す．

3.1 概　　要

クラウドのサービスを利用するクライアントは，VPN 障害によってサービスを利用できなくなってしまう．さらに，VPN 障害に対応するために VPN を切り替えたとしてもクライアント・サーバ間の通信は切れてしまう．これらの問題に対応する手法を実現する．クライアントが自律的に障害対応を取り，障害対応に係る処理を OS やアプリケーションから透過的に提供する．また障害をまたぐ通信を途切らせることなく継続する手法を実現する．

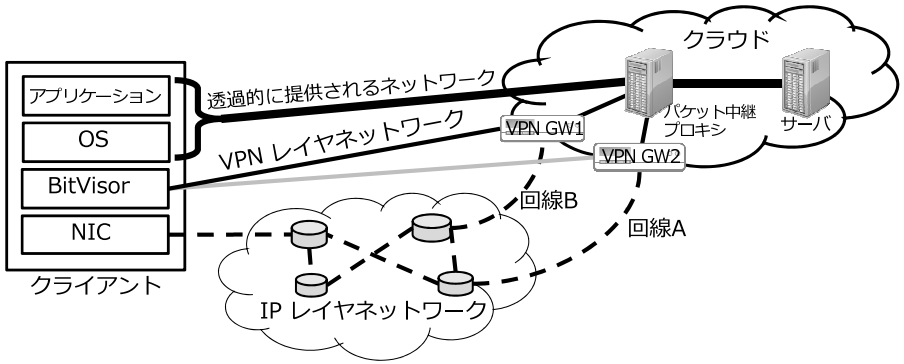

図 2 透過的障害対応システムの全体像
Fig. 2 Transparently fault recovery system details

VPN 障害の検知から VPN の切り替えに至るまでを透過的に実現するための基盤として仮想マシンモニタを用いる．また，VPN 障害をまたぐ通信を途切らすことなく透過的に継続するために，通信をネットワークレイヤで整合するパケット中継システムを構築する．これらの全体像を図 2 に示し，ネットワーク構造を図 3 に示す．

3.2 仮想マシンモニタによる障害対応

VPN 障害への対応をクライアントの OS やアプリケーションから透過的に提供するため，仮想マシンモニタのレイヤにおける障害対応を行う．仮想マシンモニタは，OS よりも低位のレイヤで動作可能なソフトウェアであり，OS から透過的に動作する．この性質を利用し，仮想マシンモニタのレイヤで OS から透過的に動作する VPN 障害の対応に係るシステムを構築する．これらを実現するために，仮想マシンモニタのレイヤでネットワークデバイスの制御が行え，また VPN クライアントが動作可能である必要がある．また，クライアントで一般的に用いられている OS である Windows が動作することが望ましい．BitVisor[8] は仮想マシンモニタとして動作し，これらの要求を全て満たしている．そこで，本提案手法の実現にあたりクライアントに据える仮想マシンモニタには BitVisor を用いる．3.2.1 節では VPN を仮想マシンモニタのレイヤで実現する手法を示し，3.2.2 節および 3.2.3 節では VPN 障害の検知及び VPN 切り替えから成る VPN 障害対応手法を示す．

3.2.1 仮想マシンモニタによる VPN 実現

仮想マシンモニタのレイヤで VPN を実現することで，VPN に関する処理を OS やアプリケーションから透過的に提供する．これを実現するために，BitVisor の VPN モジュールを用いる．BitVisor の VPN モジュールでは IPsec クライアントが動作するため，クライアント・クラウド間の通信をネットワークレイヤで送受信することが可能である．IPsec クライアントは，VPN ゲートウェイで動作する IPsec サーバとの VPN 接続を確立することで，クラウドのサービスをネットワークレイヤで利用できる．

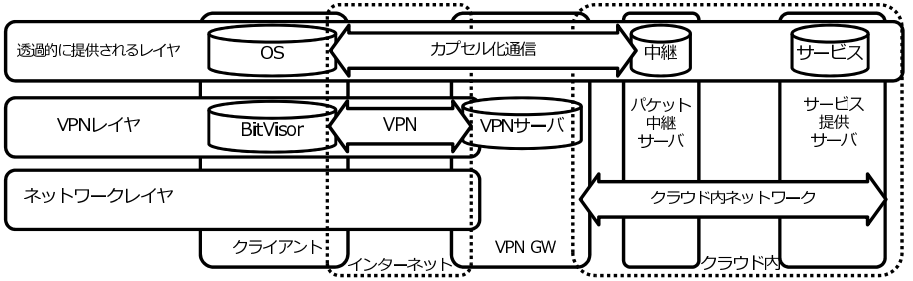

図 3 ネットワーク構造
Fig. 3 Network structure

3.2.2 VPN 障害検知

クライアントの仮想マシンモニタのレイヤにおける VPN 障害の検知手法を述べる．VPN 障害の要因をネットワークレイヤにおけるものと VPN レイヤにおけるものに大別し，それぞれにおける障害検知手法を述べる．

はじめにネットワークレイヤにおける障害検知手法を述べる．ネットワークレイヤで起きる障害は，クライアントとクラウドを結ぶ経路中のネットワークレイヤ以下で起きる障害であり，これによりクライアントからクラウドまではそもそも到達不可能である．ネットワークレイヤで双方のノードが到達可能であるかをネットワークレイヤで検知する手法がある．しかし，VPN レイヤにおける障害はネットワークレイヤでは検知できない．これは，VPN レイヤがネットワークレイヤの上位にあるためである．

次に，VPN レイヤにおける障害検知手法を述べる．VPN レイヤで起きる障害として VPN ゲートウェイにおけるインスタンスの障害が挙げられるが，これはネットワークレイヤで到達可能であっても，VPN レイヤでは到達不可能である．そのため，VPN における障害検知は VPN レイヤで行わなければならない．

そこで，VPN 障害を VPN レイヤ検知する手法を実現する．クライアントとクラウドの双方のノードが，VPN を介して到達可能であるかを検知する．まず，VPN レイヤにおける

インバウンド通信を監視する．インバウンド通信が一定期間検出されなかった際に，VPN の疎通確認を行う．実際には，VPN を介してクラウド内部のネットワーク機器との間で簡単な通信を行い，VPN が正常であればその返答を検出する．疎通確認後，さらに一定時間インバウンド通信が検出されなかった際に VPN 障害が発生したことを検知する．

3.2.3 VPN 切り替え

3.2.2 節で障害と判定された際に，VPN の切り替えを行うことで障害対応を実現する．VPN の切り替えは，ネットワークレイヤ及び VPN レイヤで生じる障害に対応するため，クラウドに対して異なるネットワーク経路を通り到達する他の VPN ゲートウェイとの間で行う．クライアントは VPN を切り替えるにあたり，あらかじめクラウドの境界に置く複数の VPN ゲートウェイとの接続情報を持つものとする．この接続情報には以下の情報を含み，これらの情報を基に新たな VPN ゲートウェイとの VPN の確立を試みる．また，VPN 切り替えに失敗した場合は，さらに他の VPN ゲートウェイとの VPN の確立を試みる．

- VPN ゲートウェイのネットワークアドレス
- 通信規約（暗号化方式，ハッシュ方式等）
- ユーザ識別情報（ユーザ識別子・パスワード・証明書）

3.3 パケット中継システムによる障害対応

VPN を切り替える際に，サーバからクライアントに向かう通信がクラウド内のネットワークレイヤで不整合となり，通信が途切れてしまう問題に対応する．そこで，パケット中継システムにおいて VPN 切り替えの前後においてクライアントを一意に識別する仕組みを持たせる．これにより，VPN 障害をまたぐ通信を途切らせることなく透過的に継続させる．パケット中継システムは，クライアントに置くパケット中継クライアントとクラウド内部に置くパケット中継サーバから成り，双方が連携して障害時の通信の整合を図る．パケット中継クライアントとパケット中継サーバとの間の通信は IP パケットの中に IP パケットを入れ込むことでカプセル化する．パケット中継システムの末端においてカプセルが解かれ，クライアントやサーバに向けてパケットが送られる．パケット中継クライアントとパケット中継サーバの構成を 3.3.1 節，3.3.2 節に示す．

3.3.1 パケット中継クライアント

パケット中継クライアントは，パケット中継プロキシと連携し，障害をまたぐ通信を途切らせることなく継続する．また，クライアントの仮想マシンモニタのレイヤで動作させることで，パケット中継システムを介した通信を OS やアプリケーションから隠蔽する．VPN 障害を検知し，VPN 切り替えを完了した直後には，クライアントのアドレスを認識させるためのパケットをパケット中継サーバに送る．

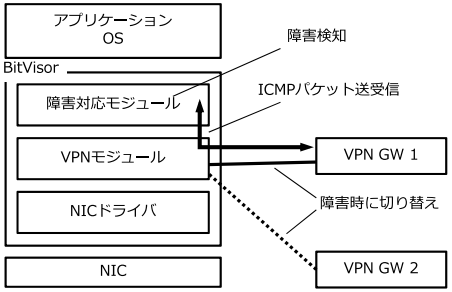

図 4 仮想マシンモニタにおける実装
Fig. 4 Implementation of the virtual machine monitor

3.3.2 パケット中継サーバ

パケット中継サーバは，クラウド内でクライアントのアドレスを把握し，クライアントとサーバとの対応付けを行う．また，クライアントが VPN 切り替え後に送るクライアント識別のためのパケットを受信すると，クライアントの対応関係を更新し，クライアントが繋がる新たな経路へと向けられる．パケット中継サーバがクライアント・サーバ間の通信を中継することで，VPN 障害をまたいでもサーバからの通信をクライアントに転送することが可能となる．

4. 実　　装

本章では，VPN 障害への透過的な対応手法の実現に向けて行った実装を述べる．4.1 節で仮想マシンモニタにおける実装を述べ，4.2 節でパケット中継システムの実装を述べる．

4.1 仮想マシンモニタにおける実装

仮想マシンモニタのレイヤでは，VPN 障害への対応に係る処理を行う．VPN 障害の対応に係る動作を OS やアプリケーションから透過的に実現するために，仮想マシンモニタのレイヤで実装を行う．実装の基盤に用いた仮想マシンモニタは BitVisor である．BitVisor が提供するコアモジュール及び VPN モジュール，NIC(Network Interface Card) ドライバを用い，図 4 に示す構成で実装を行った．4.1.1 節で VPN 障害検知における実装を述べ，4.1.2 節で VPN 切り替えの実装を述べる．

4.1.1 VPN 障害検知の実装

OS やアプリケーションを改変することなく，透過的な VPN 障害検知を実現するため，仮想マシンモニタのレイヤで VPN 障害検知を実装する．障害の検知には ICMP(Internet Control Message Protocol)[7] を用いた．クラウド内部のネットワーク機器に対して ICMP echo request を送り，VPN が正常であるときは ICMP echo reply を検出する．VPN レイヤにおける障害検知を実装するにあたり，VPN を介した通信を監視し，VPN の疎通確認を図るための ICMP パケットの送受信が行える環境が必要である．そこで，BitVisor の VPN モジュールを拡張し，仮想マシンモニタのレイヤから VPN レイヤのパケットを任意に送受信可能なネットワーク API を作成した．これにより，VPN を介してやり取りされる通信の監視が可能となり，また VPN の疎通確認を行うための VPN を介した ICMP パケットの送受信が可能となる．

VPN 障害を検知するために，ネットワーク API を用いて VPN のインバウンド通信の監視を行う．タイマーを用い，一定期間インバウンド通信が行われていないことを検知した場合に，ICMP echo request を，VPN を介してクラウドの内側のネットワーク機器に向けて送る．それから更に一定期間インバウンド通信が行われなかった場合に VPN 障害を検知する．

ICMP echo request の送信間隔を 2 秒，障害検知期間を 5 秒とした場合を例に挙げる．VPN 障害発生から障害を検知するまでに係る時間は障害検知期間と等しいため，最大でも 5 秒で障害を検知できる．また ICMP echo request の送信間隔は，ICMP echo response が届く前に障害を検知させないため，障害検知期間よりも短く設定する．

4.1.2 VPN 切り替えの実装

4.1.1 節の実装で障害を検知した後に，仮想マシンモニタのレイヤで動作する VPN 切り替えを実現する．そこで，BitVisor の VPN モジュールを拡張し，OS やアプリケーションから透過的に VPN を切り替えるモジュールを実装した．

VPN を切り替える手順は次の一連の流れで行う．VPN モジュールを停止させ，接続先情報を書き換えた後に起動する．以上の手続きで VPN 切り替えを実現する．

4.2 パケット中継システムにおける実装

VPN 障害をまたぐ通信を途切らせることなく継続する手法を，クライアントに置くパケット中継クライアントと，クラウド内に置くパケット中継サーバに分けて実装する．パケット中継サーバは，パケット中継クライアントと連携し，クライアントを識別し続ける．パケット中継クライアントは VPN 障害を検知し，VPN の切り替えが完了すると，パケット中継サーバに向けてクライアントのアドレスを通知する．パケット中継サーバはクライアントのアドレスを受け取ると，クライアントに対応するネットワークアドレスを更新する．パケット中継サーバとパケット中継クライアントの間の通信はカプセル化し，途中の経路でパケットのヘッダ情報が書き換わることを防ぐ．パケット中継クライアントは，クライアントの通信をカプセル化して VPN を介してパケット中継サーバに送り出す．VPN を介して届くカプセル化されたパケットは，その中に入るパケットを取り出して，クライアントやサーバに送る．パケットのカプセル化の手法には IP Encapsulation within IP[5] を用いる．パケット中継システムを全体像を図 5 に示す．

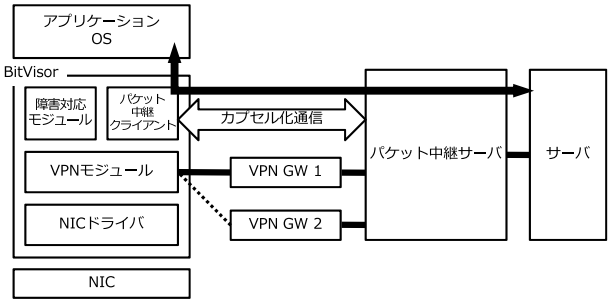

図 5 パケット中継システムにおける実装
Fig. 5 Implementation of the packet proxy system

4.2.1 パケット中継クライアントの実装

パケット中継クライアントを，BitVisor を基盤とする仮想マシンモニタのレイヤで実装する．パケット中継クライアントは二つのネットワークチャネルを持つ．一つは VPN から OS に向かう通信路，二つ目は OS から VPN に向かう通信路である．VPN から OS に向かう通信路は，パケット中継サーバを経由したパケットが流れる．このパケットは，サーバからの通信がカプセル化されているため，カプセル内のパケットを取り出して OS に向けて送信する．OS から VPN に向かう通信路は，パケット中継サーバに向かうパケットが流れ，OS から届くパケットをカプセル化して VPN を介してパケット中継サーバに送信する．また，VPN 障害が起きた後，VPN 切り替えを完了すると，パケット中継サーバにクライアントのネットワークアドレスを知らせるためのパケットを送る．

4.2.2 パケット中継サーバの実装

パケット中継サーバはクラウド内で動作し，クライアント・サーバ間の通信を中継する．

サーバはクライアントのアドレスを把握し，クライアントの識別パケットを受け取ると，アドレスとクライアントの対応関係を更新する．

これらのパケット中継サーバを，開発環境に Ubuntu 9.10 kernel 2.6.31 を用いて開発した．IP パケットをユーザ空間で受信するために，iptables の ipqueue を用い，IP パケットをユーザ空間から送信するために libnet を用いて実装した．パケット中継サーバを動作させるために，iptables でインバウンドの IP パケットを ipqueue 経由にする設定をあらかじめ行う．

5. 評　　価

本章では，VPN 障害期間の評価とパケット中継システムにおけるネットワーク性能の評価を述べる．VPN 障害機関の評価では，VPN 障害期間の内訳を示し，また VPN 障害をまたぐ通信のスループットを示す．パケット中継システムにおけるネットワーク性能の評価では，通信のバンド幅とレイテンシから評価する．これらの評価に用いたクライアント，VPN ゲートウェイ，パケット中継サーバ，サーバの環境を表 1 に示す．

5.1 VPN 障害期間の評価

VPN 障害の発生からクライアントで検知するまでにかかる期間を評価する．はじめに Aggressive Mode で動作する IPsec において，VPN を確立する際にかかる期間とその内訳を表 2 に示す．Aggressive Mode では，イニシエータ側 (クライアント) で 2 つ，レスポンダ側（VPN ゲートウェイ）で 1 つの通信パケットを送信することで IPsec における一つのフェーズを終える．IPsec では二つのフェーズによって IPsec を確立する．フェーズ 1 では ISAMP SA のパラメータを交換し，フェーズ 2 では IPsec SA のパラメータを交換する．表 2 より，フェーズ 1 で 181[ms]，フェーズ 2 では 137[ms] かかっており，IPsec を確立するためには合計 318[ms] が必要である．この数値は使用する VPN ルータ，ネットワーク構成等によって変動するが，同様の環境化においては一定である．そこで，クラウドのサービスが許容する VPN 障害期間を定め，それに合わせた障害検知期間の設定を行う．F を障害検知期間，A を許容する障害期間，C を VPN 切り替え期間とすると，障害検知期間は式 1 で求まる．

$$F = A - C \tag{1}$$

障害検知期間を 5[sec] として設定し，クライアントがサーバからファイルをコピーしている最中に VPN 障害を起こした．その時のスループットの推移を図 6 に示す．35 秒付近で VPN 障害が生じており，40 秒付近で障害を検知し，VPN 切り替えを行い，障害から回復している．障害の前後で，通信は継続している．これより，本提案手法が有効に機能していることを確認した．

5.2 パケット中継プロキシのネットワーク性能

本章では，パケット中継プロキシのネットワーク性能を述べる．ベンチマークの測定には lmbench を用いた．表 3 で TCP のバンド幅を示し，表 4 で TCP，UDP のレイテンシを示す．表中の (a) はクライアントマシン単体のネットワーク性能，(b) は VPN クライアン

表 1　評価環境
Table 1　Evaluation environment

クライアント	
CPU	Intel Core 2 Duo E6400 2.13GHz
メモリ	DDR2 667MHz 2GB
NIC	Intel Pro 1000
仮想マシンモニタ	BitVisor 1.0
メモリ	128MB
ゲスト OS	Windows XP Home Edition SP3
メモリ	1920MB
VPN ゲートウェイ	
VPN Router	NetScreen 5GT
パケット中継サーバ	
CPU	Intel Core 2 Duo E6300 1.86GHz
メモリ	DDR2 667MHz 1GB
OS	Ubuntu 9.10 kernel 2.6.31
サーバ	
CPU	Intel Core i7 965 3.2GHz
メモリ	DDR3 6GB
OS	Ubuntu 9.10 kernel 2.6.31

表 2　IPsec 切り替え時にかかる期間の内訳
Table 2　IPsec breakdown of the time required when switching

構成	経過時間期間 [ms]
フェーズ 1	181
フェーズ 2	137
合計	318

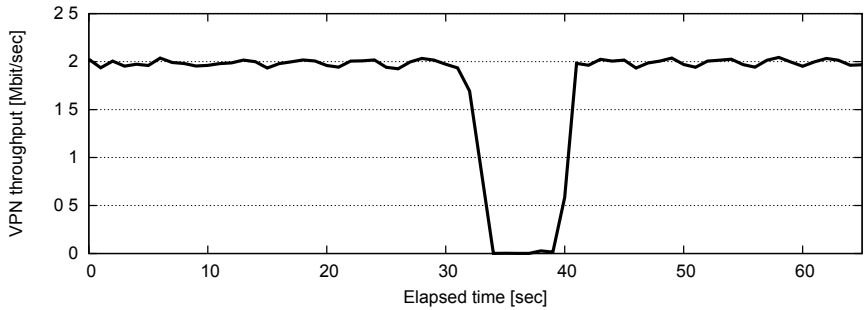

図 6 VPN 切り替え前後の VPN スループットの時間推移
Fig. 6 VPN throughput of transition period before and after failure

トを有効にした BitVisor を用いたネットワーク性能，(c) は (b) に加えてパケット中継システムを用いたネットワーク性能である．

BitVisor を用いることでバンド幅，レイテンシともに急激な性能低下がみられる．急激な性能低下の理由として，IPsec 自体の処理が重いことと，BitVisor に起因した性能低下が大きいことが挙げられる．(c) を見ると，(b) と比べバンド幅は 44%低下し，レイテンシも TCP，UDP で 240%に増加した．パケット中継システムによるパフォーマンスの低下が大きい理由として，パケット中継サーバの実装が挙げられる．IP パケットを iptables における ipqueue に介すことにより，パケットがカーネル空間からユーザ空間にコピーされ，またユーザ空間でパケットの中継処理を行っている．これにより高いオーバヘッドが生じていることが考えられる．

表 3 TCP バンド幅
Table 3 TCP bandwidth

Composition	Bandwidth [Mbit/s]
(a)	91.44
(b)	4.00
(c)	1.76

表 4 TCP/UDP レイテンシ
Table 4 TCP/UDP latency

Composition	Latency [μs]	
	TCP	UDP
(a)	154.76	153.72
(b)	4926.57	4915.49
(c)	11844.09	11808.51

6. 関連研究

Peer-to-peer やオーバレイネットワークが提供するルーティングメカニズムによって，ネットワーク経路の可用性を向上させる研究が行われている[9),12),13)]．これらで想定されている障害として，BGP(Border Gateway Protocol) における人為的な設定ミスによる障害や，AS(Autonomous System) の周辺で起きる障害があげられるが，これらを実現するにあたり，広域に分散した多数のノードが必要である．一方で，本提案はネットワークレイヤの障害に対応するために複数の VPN ゲートウェイにそれぞれ異なる外部ネットワーク回線につなぐことでネットワーク経路を多様化させる手法をとる．本稿で想定する，クラウドまでの通信経路を確保するために，最小限の対応で障害対応が可能である．

ルータの多重化を行うプロトコルである VRRP(Virtual Router Redundancy Protocol)[6)] は，ルータをマスター機とスレーブ機に多重化して配置し，マスター機に障害が生じた際には自動的にスレーブ機に切り替えることで，ルータ周辺で起きる局所的な障害への対応が可能である．しかし，ルータ周辺で起きる局所的な障害対応だけでは，途中のネットワーク経路における障害には対応できない．一方で本提案は，VPN ゲートウェイでの障害に加え，途中の経路の障害に対応可能である．

IP ノードの移動透過性を実現するプロトコルである Mobile IP[4)] は，ノードが異なる IP 体系を持つネットワークに移動した際にも通信を透過的に継続することができる．しかし，モバイル端末を前提に設計されたプロトコルであるため，端末の拠点となる HA(Home Agent) や端末の近傍のネットワークに置く FA(Foreign Agent) 等の複数のエージェントによるサポートが必要である．一方で本提案は，クライアントの仮想マシンモニタのレイヤに置くパケット中継クライアントと，クラウド内に置くパケット中継サーバの二つのシステムで構成可能である．障害発生時の対応は，クライアントの OS やアプリケーションから透過に行われ，障害をまたぐ通信を透過的に継続することが可能である．

7. おわりに

本稿では，VPN 障害への透過的な対応として，仮想マシンモニタを用いる手法とパケット中継システムを用いる手法を提案し，それらの実装を行った．仮想マシンモニタを用い

た手法では，仮想マシンモニタのレイヤで VPN 障害検知と VPN 切り替えを実現し，これを OS やアプリケーションから透過的に実現した．パケット中継システムを用いる手法では，VPN 切り替えによってクライアント・サーバ間の通信が途切れてしまう問題を解消し，VPN 障害をまたいでも通信を途切らせることなく継続した．これらの評価を行い，VPN 障害に対し透過的な障害対応を実現することを確認した．

今後の課題は，複数クライアントへの対応，パケット中継で生じるオーバヘッドの削減，BitVisor におけるネットワーク関連モジュールのオーバヘッド削減である．また，複数クライアントに対応することによって増大するパケット中継サーバに対する負荷が与えるパフォーマンスの評価も課題である．

謝　　辞

本研究の一部は総務省・戦略的情報通信研究開発推進制度 (SCOPE) の支援により行われた．

参 考 文 献

1) Amazon.com, I.: Amazon Virtual Private Cloud (2009). http://aws.amazon.com/vpc/.
2) NTT Communications, I.: Salesforce over VPN (2009). http://www.ntt.com/soas/.
3) Farkas, J., Antal, C., Westberg, L., Paradisi, A., Tronco, T.R. and Oliviera, V.G.: Fast failure handling in ethernet networks, Communications, ICC '06: IEEE International Conference, Vol.2, pp.841-846 (2006).
4) Perkins, C.: IP Mobility Support, RFC 2002 (Internet standards) (1996).
5) Perkins, C.: IP Encapsulation within IP, RFC 2003 (Internet standards) (1996).
6) Hinden, R.: Virtual Router Redundancy Protocol, RFC 3768 (Internet standards) (2004).
7) Postel, J.: Internet Control Protocol, RFC 792 (1981).
8) Shinagawa, T., Eiraku, H., Tanimoto, K., Omote, K., Hasegawa, S., Horie, T., Hirano, M., Kourai, K., Oyama, Y., Kawai, E., Kono, K., Chiba, S., Shinjo, Y. and Kato, K.: BitVisor: A Thin Hypervisor for Enforcing I/O Device Security, In Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual Execution Environments, pp.121-130 (2009).
9) Harvey, N.J.A., Jones, M.B., Saroiu, S., Theimer, M. and Wolman A.: SkipNet: A Scalable Overlay Network with Practical Locality Properties, Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems, Vol.4, pp.113-126 (2003).
10) Kim, C.: Scalable and Efficient Self-configuring networks, Princeton university, technical report, TR-860-09 (2009).
11) Agosta, J.M., Chandrashekar, J., Dash, D.H., Dave, M., Durham, D., Khosravi, H., Li, H., Purcell, S., Rungta, S., Sahita, R., Savagaonkar, U. and Schooler, E.M.: Towards autonomic enterprise security: self-defending platforms, distributed detection, and adaptive feedback, Intel technology journal, Vol.10, Issue.4 (2006).
12) Andersen, D., Balakrishnan, H., Kaashoek, F. and Morris R.: Resilient overlay network, Proceedings of the eighteenth ACM symposium on Operating system principles, pp.131-145 (2001).
13) Aspnes, James., Diamadi, Z. and Shah, G.: Fault-tolerant routing in peer-to-peer systems, Proceedings of the twenty-first annual symposium on Principles of distributed computing, pp.223-232 (2002).
14) Oppenheimer, D., Ganapathi, A. and Patterson, D.A.: Why do Internet services fail, and what can be done about it?, Proceedings of Fourth USENIX Symposium on Internet Technologies and Systems (USITS'03) (2003).
15) Labovitz, C. and Ahuja, A.: Experimental Study of Internet Stability and Wide-Area Backbone Failures, In Fault-Tolerant Computing Symposium (FTCS) (1999).
16) Rowstron, A. and Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer system, International Conference on Distributed Systems Platforms (Middleware), pp.329-350 (2001).
17) Stoica, I., Morris, R., Karger, D., Kaashoek, M.F. and Balakrishnan, H.: Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications, Proceedings of the ACM SIGCOMM '01 Conference, pp.149-160 (2001).