社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－DPS－126（31）
2006－CSEC－32（31）
2006／3／17

ネットワーク仮想化技術を用いたポリシーベースVPNの提案

岩田 浩真† 中沢 実‡ 千石 靖† 服部 進実‡

† 金沢工業大学大学院工学研究科情報工学専攻
〒 921–8501 石川県石川郡野々市町扇が丘 7–1
‡ 金沢工業大学大学院工学研究科知的創造システム専攻
〒 105–0002 東京都港区愛宕 1–3–4 愛宕東洋ビル 12F
E-mail: {iwata,nakazawa,sengoku,hattori}@infor.kanazawa-it.ac.jp

あらまし 2つのホスト間を仮想的な専用線で接続するVPN（Virtual Private Network）という技術がある．VPNは，自宅や外出先からインターネットを経由して，企業や大学の内部ネットワークへ，安全にアクセスできる．特に，仮想的なネットワークインターフェイスをマシン上に構築するVPNは，既存のプログラムを修正することなく，プログラムがリモートネットワークと通信できる．しかし，このVPNは，ネットワークの経路表を動的に書き換え，通信中のプログラムに影響を及ぼす可能性がある．本稿では，プログラムごとに仮想的なネットワーク環境を構築し，通信中の他のプログラムに影響を与えない，ポリシーベースVPNの提案を行う．

A Proposal of Policy based VPN using Virtualized Network method

Kooshin IWATA†, Minoru NAKAZAWA‡, Yasushi SENGOKU†, and Shimmi HATTORI‡

†Graduate School of Information and Computer Engineering, Kanazawa Institute of Technology, 7-1, Ogigaoka, Nonoichi-machi, Ishikawagun, Ishikawa, 921–8501 Japan
‡Graduate Program in Systems for Intellectual Creation, Kanazawa Institute of Technology, Atago Toyo Bldg.12F Atago 1–3–4, Minato-ku, Tokyo, 105–0002 Japan
E-mail: {iwata,nakazawa,sengoku,hattori}@infor.kanazawa-it.ac.jp

Abstract A VPN is a technology which connects two hosts by a virtual private line. It enables us to safely access Intranet of a company or a university via Internet from our house or out-of-doors. The VPN which constructs a virtual network interface on the machine communicates with remote networks without needing to correct the program. However, this particular VPN automatically rewrites the routing table of the network, which might influence other programs during communication. In this paper, we propose a policy-based-VPN which constructs a virtual network environment at each program, which does not influence other programs during communication.

1 はじめに

現在，ホスト間を仮想的な専用線で接続するVPN（Virtual Private Network）は，企業や大学の拠点間を結ぶためや，モバイル端末がリモートアクセスするための技術として普及している．その中でも，アクセス回線に安価なインターネットを用いてVPN通信を行う，インターネットVPN（以下，VPN）が主流になりつつある．

VPNには，IPsec[1] やPPTP（Point-to-Point Tunneling Protocol）[2] などの標準化されたプロトコルがある．また，SSL-VPNやOpenVPN[3] など，独自のプロトコルを用いるVPNもある．

主なVPNは，仮想的なネットワークインターフェイスを構築し，新たな通信経路を確保するため，既存のプログラムを修正することなく，プログラムがリモートネットワークと通信できる．しかし，このVPNは，ネットワークの経路表を動的に書き換え

るため，通信中のプログラムに影響を及ぼす可能性がある．

本稿では，この問題を解決するために，仮想的に独立したネットワーク環境を構築し，通信中の他のプログラムに影響を与えない，ポリシーベース VPN の提案について述べる．

文献 [4] では，TCP/IP プロトコルスタックを半仮想化し，Web サーバプロセスを独立したネットワーク環境で動作させる手法が述べられている．この手法は，プロセスに対する通信をアクセス制御するため，独立したネットワーク環境を構築している．本稿では，これらの手法を応用し，独立ネットワーク環境を構築するポリシーベース VPN について述べる．

本稿では，第 2 章で VPN 利用時に発生しうる問題点について，第 3 章で本提案であるポリシーベース VPN について，第 4 章でその実装方針について，第 5 章でまとめと今後の予定について述べる．

2　従来の VPN の問題

本章では，仮想ネットワークインターフェイスを構築し，VPN クライアントに IP アドレスを割り当てる，VPN に関する問題点と既存の解決策について述べる．

2.1　プライベート IP アドレス

IP アドレスは，世界中で一意に識別できるグローバル IP アドレスと，企業や大学など組織の中で一意に識別できるプライベート IP アドレスがある．大学や企業の内部ネットワークは，グローバル IP アドレスの不足から，プライベート IP アドレスで運用されていることが多い．

2.1.1　アドレスの衝突

VPN クライアントは，別の組織が管理運用するリモートネットワークに接続する．このとき，リモートネットワークとローカルネットワークがプライベート IP アドレスで運用され，各ホストに割り当てられた IP アドレスの体系が似ていると，両ネットワークで同じ IP アドレスを割り当てられたホストが存在し，アドレス衝突が発生する可能性がある．

図 1 の VPN クライアントは，所属しているローカルネットワークには 192.168.11.0/24 から 192.168.13.0/24 のネットワークが存在し，192.16.11.51/24 の IP アドレスを割り当てられている．一方，VPN 接続先のリモートネットワークには 192.168.9.0/24 から 192.168.11.0/24 のネットワークが存在し，VPN サーバから 192.168.9.52 の IP アドレスを割り当てられている．リモートネットワークとローカルネットワークは，プライベート IP アドレスの体系が似ており，192.168.11.0/24 のネットワークが重複して存在している．

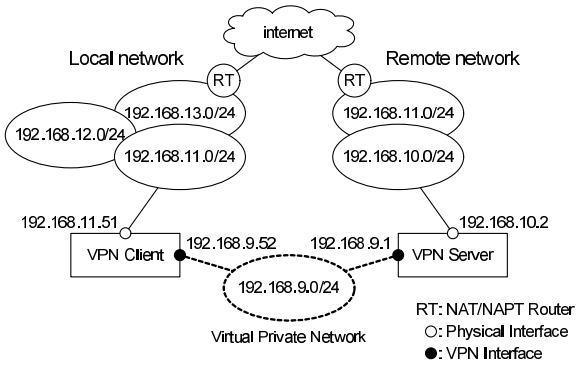

図 1: リモートネットワークに接続中の VPN クライアント

2.1.2　アドレス衝突の問題点

VPN クライアントは，ホストのアドレス衝突が発生した場合，経路表にアクセスするホストへの経路を加えることで，一方のホストのみへアクセスできる．しかし，2 つのネットワークのホストへ，同時にアクセスできない問題がある．

2.1.3　既存のアドレス衝突の解決策

既存の解決策として，NAT/NAPT（Network Address Translation / Network Address Port Translation）を用いて，VPN クライアントと VPN サーバの間で，IP アドレスやポート番号の変換を行う手法がある．NAT/NAPT とは，ルータやファイアウォールなどで，パケットの送信元/あて先 IP アドレスや送信元/あて先ポート番号を，予め決められた変換ルールによって，相互に変換することである．

この NAT/NAPT を，VPN クライアントと VPN サーバの間に適用し，必要なサービスを提供するサーバの IP アドレスやポート番号を他の IP アド

レスやポート番号に変換するルールによって，アドレス衝突を回避する．

しかし，この手法では，様々なネットワークから接続することを想定して，複数のルールを記述する必要がある．また，NAT/NAPT を適用すると，ホスト間の通信が一方向からしか通信を開始できない非対称ネットワークとなるため，NAT/NAPT を適用できないサービスがある．

2.2 DNS サーバによる名前解決

Web やメール等のサービスを提供するホストは，管理運用上の都合や，利用者の利便性のために，FQDN（Fully Qualified Domain Name）で表現されたホスト名でアクセスされる．これらのホストへ実際にアクセスするには，IP アドレスが必要であるため，ホスト名から IP アドレスに名前解決する必要があり，通常は DNS（Domain Name System）サーバを用いて名前解決する．このとき，どの DNS サーバで名前解決するかという問題がある．本稿では，DNS サーバを用いて，IP アドレスとホスト名の名前解決を行うことを，DNS 名前解決と呼ぶ．

2.2.1 複数 DNS サーバ

DNS 名前解決するホストでは，DNS サーバに障害が発生しアクセスできなくなることに備え，優先度をつけて複数の DNS サーバを指定できる．しかし，最も優先度の高い DNS サーバにアクセスできない場合のみ，次に優先度の高い DNS サーバが使用される．このため，優先度の低い DNS サーバで DNS 名前解決が成功する場合でも，最も優先度の高い DNS サーバで DNS 名前解決が失敗すると，DNS 名前解決に失敗することになる．

2.2.2 DNS 名前解決の問題点

プライベート IP アドレスで運用されている内部ネットワークは，その組織内のプライベート IP アドレスとホスト名の DNS 名前解決するために，内部向け DNS サーバがある．VPN 接続を行う VPN クライアントは，所属するローカルネットワークと VPN 接続先のリモートネットワークにある 2 つの内部向け DNS サーバを指定する．VPN クライアントは，それぞれのネットワークごとに閉じた DNS 名前解決が行われている場合，同時に 1 つのネットワークの DNS 名前解決しかできない問題が発生する．

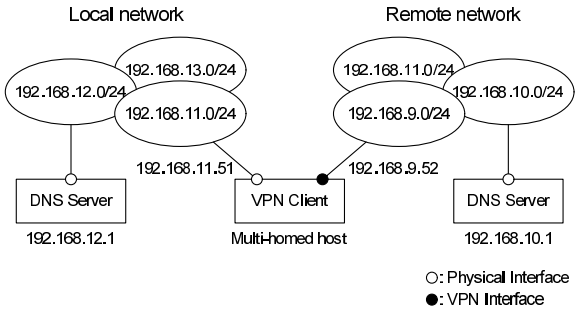

図 2: マルチホームホストの VPN クライアント

2.2.3 既存の DNS 名前解決

hosts ファイル　マシン上で名前解決は，リゾルバと呼ばれる部分で行われる．リゾルバでは，DNS 名前解決以外にも，ホスト名と IP アドレスを列挙した，hosts ファイルを用いて名前解決できる．この hosts ファイルは，最も優先して名前解決されるため，アクセスに必要なホスト名と IP アドレスを予め記述しておく手法がある．

しかし，IP アドレスとホスト名を静的に対応付けて記述しているため，IP アドレスに変更があった場合など，名前解決に不整合が発生する．ホストごとに管理するファイルであるため，変更が生じたとき，維持管理が問題となる．

リゾルバによる DNS サーバ選択　リゾルバで，ドメインのゾーンごとに，DNS 名前解決する DNS サーバを選択する手法が考えられる．ドメイン名を元に，DNS サーバを選択できるため，ホスト名から IP アドレスを求める正引きは正しく行える．しかし，プライベート IP アドレスからホスト名を求める逆引きを行う場合は，DNS サーバの選択基準がないため，DNS 名前解決できない．

2.3 マルチホームホスト

複数のネットワークインターフェイスがあり，複数の IP アドレスが割り当てられているホストを，マルチホームホストと呼ぶ．図 2 の VPN 接続を行う VPN クライアントは，VPN インターフェイスが追加され，リモートネットワークの IP アドレスが新たに割り当てられることから，マルチホームホストといえる．

2.3.1 デフォルト経路

図 2 の VPN クライアントは，所属するローカルネットワーク 192.168.11.0/24 内のルータをデフォルト経路とする．デフォルト経路とは，パケット転送時のあて先が，経路表に登録してある経路情報と一致する経路がない場合に，選択される経路である．

1 つのインターフェイスを持つホストは，デフォルト経路として，そのネットワーク内のルータを指定するだけで，他のネットワークのホストと通信できる．しかし，マルチホームホストは，複数のルータが存在し，複数のデフォルト経路を設定できないため，ネットワークに応じた経路設定をする必要がある．

2.3.2 VPN 経路問題

VPN クライアントは，リモートネットワークのルータをデフォルト経路に設定すると，ローカルネットワークにおいて，所属しているネットワーク以外へアクセスできなくなる．ただし，VPN 通信に必要な VPN クライアントから VPN サーバへの経路は，VPN ソフトウェア側で固定的に設定するため，デフォルト経路の影響は受けない．

VPN クライアントが，リモートネットワークとローカルネットワークのホストへ同時にアクセスするには，両方のネットワーク構成を把握し，経路表にネットワークごとの経路情報を設定する必要がある．しかし，一般の利用者が，リモートネットワークとローカルネットワークの状況を判断して，経路設定するのは難しいという問題がある．

2.3.3 ポリシー経路制御

通信ごとに経路変更や柔軟な経路制御を行うために，ポリシー経路制御（Policy Routing）と呼ばれる技術がある．ポリシー経路制御とは，IP アドレスやポート番号などによって通信を識別する情報と，その通信の経路を指定した情報を組み合わせたポリシーを元に，経路制御することである．

ポリシー経路制御は，柔軟に経路制御できるが，ネットワークの構成を把握し，各ネットワークに合わせたポリシーを記述する必要があり，ポリシーの記述が難しいという問題がある．

3 ポリシーベース VPN の提案

本章では，問題点の解決策として，仮想的に独立したネットワーク環境を構築するポリシーベース VPN の提案の概要について述べる．

3.1 独立ネットワーク環境

プロセスは，複数の IP アドレスがある場合，通信に使用する IP アドレスを選択して，様々なネットワークのホストと通信する．このとき，プロセスに対して，提供する IP アドレス等のネットワーク情報を限定することで，仮想的に独立したネットワーク環境にする．

独立ネットワーク環境とは，図 3 のように，プロセスグループごとに，接続するネットワークを限定させることである．図の場合，プロセス 1，2 が物理的なネットワーク環境に，プロセス 3，4 が VPN によって構築された仮想ネットワーク環境に，それぞれ所属している．

VPN に独立ネットワーク環境を適用することで，既存の通信中のプロセスに影響を与えずに，必要最小限のプロセスのみ VPN 通信ができる．本稿では，それぞれ独立したネットワーク環境について，ホストが物理的に所属しているネットワーク環境のことを実ネットワーク環境と呼び，VPN によって構築されたネットワーク環境のことを VPN 環境と呼ぶ．

3.2 環境構築ポリシー

利用者は，独立したネットワーク環境の構築や，どのプロセスをどのネットワーク環境に所属させるかを決定するためのポリシーを記述する．ポリシーは，プロセスをどの環境に所属させるかを記述するだけであり，ポリシー経路制御よりも簡潔に記述できる．このため，一般の利用者は，ネットワークの構成を把握する必要がなくなり，容易にポリシーを記述できる．

3.3 アドレス衝突

VPN 接続時に独立ネットワーク環境では，リモートネットワークとローカルネットワークが，実ネットワーク環境と VPN 環境でそれぞれ閉じているため，IP アドレスの衝突が発生しない．

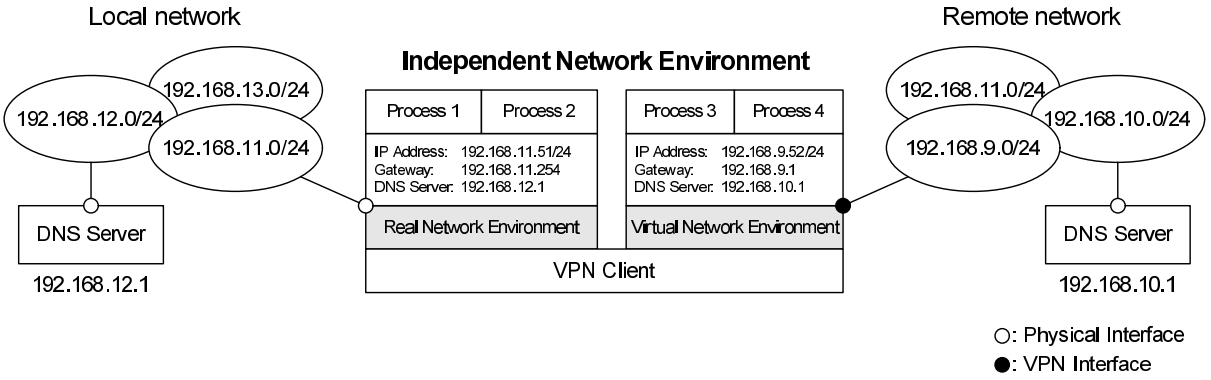

図 3: 独立ネットワーク環境

図 3 では，各ネットワーク環境で閉じているため，192.168.11.0/24 のネットワークが重複せず，プロセスは目的のホストへアクセスできる．

3.4 DNS 名前解決

プロセスは，リゾルバによって各ネットワーク環境に関連付けられた DNS サーバで名前解決するため，プライベート IP アドレスに対応するホスト名などは期待通りに DNS 名前解決できる．

図 3 では，実ネットワーク環境での DNS 名前解決は 192.168.12.1 の DNS サーバが選択され，VPN 環境での DNS 名前解決は 192.168.10.1 の DNS サーバが選択されるため，プロセスは対応したネットワークの DNS 名前解決ができる．

3.5 VPN 経路

VPN クライアントは，VPN 接続時に新たなデフォルト経路ができる点が問題であったが，ポリシーベース VPN は実ネットワーク環境と VPN 環境で経路が互いに独立しているため，マルチホームホストではない．これにより，ネットワーク環境ごとにデフォルト経路を設定でき，複雑な経路設定をする必要がなくなる．

図 3 では，実ネットワーク環境でのデフォルト経路は 192.168.11.254 のルータが選択され，VPN 環境でのデフォルト経路は 192.168.9.1 の VPN サーバが選択されるため，複雑な経路設定をせずに，プロセスは目的のホストと通信できる．

4 実装方針

本章では，独立ネットワーク環境の構築のために，仮想ソケット方式と仮想経路表方式の，2 通りの実装方針の概要について述べる．

4.1 仮想ソケット方式

プログラムが TCP や UDP で通信を行う場合，通信の確立に必要な操作などは，OS とプログラム間のインターフェイスを通じて行う．このインターフェイスとは，UNIX では socket などのシステムコール，Windows では Winsock などである．本稿では，これらネットワークに関係する操作を行うインターフェイスを，Socket API と呼ぶ．

4.1.1 仮想ソケット

通信するプロセスは，Socket API を通じてソケット作成やデータ転送などを行う．そこで，プロセスが Socket API を通じて何らかの操作を行うときに，その操作や得られる情報を制限することで，仮想的に閉じたネットワーク環境を構築する．本稿では，この手法で独立ネットワーク環境を構築することを，仮想ソケット方式と呼ぶ．

図 4 の (1) では，VPN 制御プロセスで VPN に接続するなどの処理を行い，その上の仮想ソケットで VPN 環境を構築する．各 VPN 環境に属するプロセスは，ソケットではなく，仮想ソケットの提供する Socket API を用いて，通信を行う．

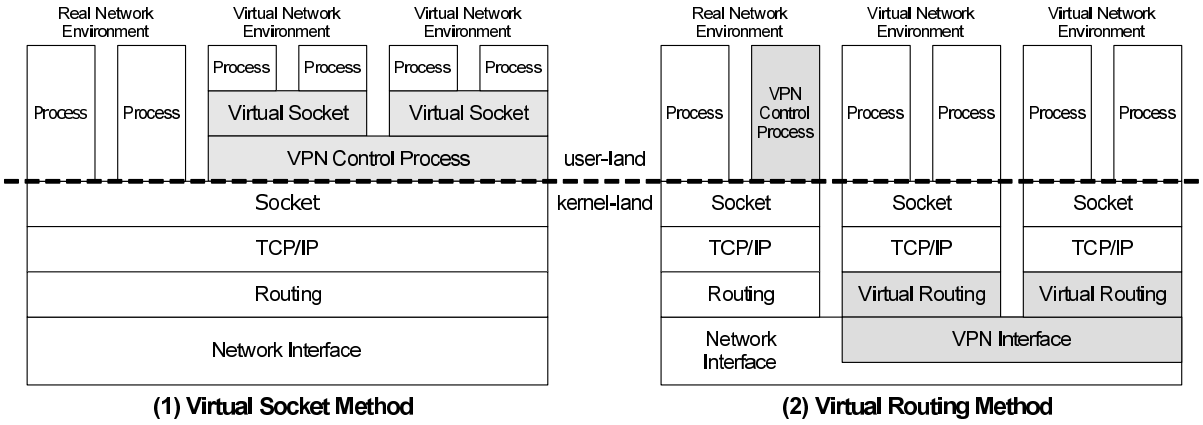

図 4: 独立ネットワーク環境のプロトコルスタック

4.1.2 Socket API のフック

仮想ソケット方式は，プロセスから Socket API が呼び出されるとき，別に用意した Socket API を呼び出されるように，Socket API のフックを行う．別に用意した Socket API で，ネットワーク環境ごとにソケット作成などを行うことよって，プロセスのネットワーク環境を独立させる．

4.2 仮想経路表方式

経路表は，パケット転送先の経路決定や，始点アドレスを決定する．経路表は，ネットワークアドレスとそのネットワークにパケットを転送する経路である次の転送先 IP アドレスを記した経路情報があり，複数の経路情報が登録されている．TCP/IP は，経路表を参照し，あて先 IP アドレスとネットワークアドレスをアドレスの先頭から比較し，最も一致する部分が長かった経路情報を元に，経路決定する．

4.2.1 仮想経路表

通常のネットワーク環境では，マシン上に 1 つの経路表があり，すべての通信はその経路表を参照して経路決定する．そこで，マシン上に複数の仮想的な経路表を構築し，プロセスごとに別々に経路制御することで，独立したネットワーク環境を構築する．本稿では，この手法で独立ネットワーク環境を構築することを，仮想経路表方式と呼ぶ．

図 4 の (2) では，VPN 制御プロセスで VPN に接続するなどの処理を行い，新たな経路として VPN インターフェイスを構築し，その上の仮想経路表で VPN 環境を構築する．VPN 環境に属するプロセスの通信は，通常通り TCP/IP の処理をし，ネットワーク環境ごとに用意された経路表を用いて経路制御する．

5 おわりに

本稿では，既存の VPN の問題点に言及し，独立ネットワーク環境を構築するポリシーベース VPN による解決策を述べた．また，仮想ソケット方式と仮想経路表による，2 通りのポリシーベース VPN の実装方針を述べた．このポリシーベース VPN の実現により，柔軟に VPN を構築することができると考える．

今後は，ポリシーベース VPN を様々な OS で実装し，独立ネットワーク環境の有効性の検証を行っていく．

参考文献

[1] Kent, S. and Atkinson, R.: “Security Architecture for the Internet Protocol”, RFC2401, Nov. 1998.

[2] Hamzeh, K., Pall, G., Verthein, W., Taarud, J., Little, W. and Zorn, G.: “Point-to-Point Tunneling Protocol”, RFC2637, Jul. 1999.

[3] James Yonan: “OpenVPN - An Open Source SSL VPN Solution”, http://openvpn.net

[4] C, Dalton and T. H. Choo: “An Operating System Approach to Securing E-Services”, Comm. of the ACM, 44(2):58-64, Feb 2001.