VPN とホストの実行環境を統合するパーソナルネットワーク

光来 健一† 廣津 登志夫† 佐藤 孝治† 明石 修† 福田 健介†
菅原 俊治† 千葉 滋‡

† NTT 未来ねっと研究所 ‡ 東京工業大学

要 旨

ユーザが自由に VPN を使うようになり，1 つのホストで複数のネットワークを同時に扱う状況が増えてきている．しかしながら，従来の OS は複数のネットワークを排他的に利用する機構を提供していないため，IP アドレスが衝突するネットワークを同時に扱えず，VPN 内部の機密情報を他のネットワークに漏らしてしまう危険性もある．そこで我々は複数の VPN を扱うホストの実行環境を VPN 毎に分離し，その実行環境と VPN を統合した**パーソナルネットワーク**を提案する．**ポートスペース**と呼ばれるこの分離された実行環境は，プロセスを実行する時のネットワークやファイルシステムの環境であり，プロセスと VPN を密接に関係づけることにより独立したパーソナルネットワークの構築を可能にする．

1 はじめに

近年，仮想プライベートネットワーク（VPN）が自宅や外出先と会社などとの間で構築されるようになり，ユーザが必要に応じて自由に VPN を使えるようになってきている．その結果，1 つのホストが VPN を用いない従来のネットワーク（ベースネットワーク）と共に，複数の VPN を同時に扱う状況が増えてきている．例えば，ユーザは自宅で会社のネットワークとの間の VPN を使いながら，インターネット上のウェブサイトも見ている．いくつかの組織に所属する人ならそれぞれのネットワークとの間の VPN を利用している．また，次第に普及してきている P2P ネットワークも広い意味で VPN の一種と考えることもできる．

しかしながら，1 つのホストで VPN を他の VPN やベースネットワークと同時に扱うのには問題がある．1 つは VPN やベースネットワークの IP アドレスが衝突する場合には同時に使うことができないことである．また，ホスト上のファイルシステムやプロセスを経由して VPN 内部の機密情報が他のネットワークに漏れてしまう危険性もある．この原因は各ホストの OS が複数のネットワークを排他的に利用する機構を提供していないことである．

そこで，我々は複数の VPN を扱うホストの実行環境を VPN 毎に分離し，その実行環境と VPN を統合した**パーソナルネットワーク**を提案する．**ポートスペース**と呼ばれるこの分離された実行環境は，プロセスを実行する時のネットワークやファイルシステムなどの環境である．パーソナルネットワークはポートスペースを介してプロセスと VPN を結びつけることにより，1 つのホストで複数の VPN を排他的に扱うことを可能にする．また，各パーソナルネットワークは完全に独立しているため独自のネットワーク管理を行うことができ，ベースネットワークと同じ IP アドレスを使うこともできる．

ポートスペースは独立した実行環境を実現するために，ネットワーク，ファイルシステムおよびプロセスの名前空間を分離する．ユーザは新しく作ったポートスペースで自由にネットワークを設定したり，ファイルシステムを構築したりできる．ただし，何も設定されていない状態からでは実行環境を作り上げるのが大変なので，ポートスペースを作る時には親ポートスペースを継承して名前空間の一部を引き継ぐことができる．これにより，ユーザはネットワークやファイルシステムを一から設定する必要がなくなり，簡単にポートスペースを作れるようになる．

以下，2 章ではホストが複数の VPN を扱う際の問題を解決するパーソナルネットワークについて述べる．3 章ではポートスペースの設計とパーソナルネットワークの管理について説明し，4 章でその実装について述べる．5 章でポートスペースのオーバヘッドを調べる実験について述べた後，6 章で関連

研究に触れ，7 章で本稿をまとめる．

2　パーソナルネットワーク

2.1　複数 VPN の同時利用の問題

1 つのホストが複数のネットワーク（VPN およびベースネットワーク）を扱うようになるといくつかの問題が生じる．1 つの問題は，それぞれのネットワークの間で IP アドレスやホスト名などの名前空間が衝突する可能性があることである．VPN やベースネットワーク内のホストにプライベート IP アドレスが割り振られている場合，同じ IP アドレスを持つホストがあるとうまくルーティングできない．

もう 1 つの問題はネットワーク間で機密情報が漏れる可能性があることである．例えば，VPN 内部のウェブページとインターネット上のウェブページを 1 つのブラウザで閲覧していると，ユーザの操作ミスやブラウザのバグによって VPN 内の機密情報がインターネットに流出する可能性がある．また，メールクライアントがある VPN から届いたウィルスメールの添付ファイルを実行してしまい，ディスク上の機密情報を他のネットワークに送信してしまうかもしれない．

これらの問題を解決するには，ホスト上でネットワークを排他的に利用できるようにすることが不可欠である．ネットワークを排他的に利用できるならば，IP アドレスなどの名前空間の衝突を気にする必要はなく，意図しない情報の流れも防ぐことができる．しかしながら，従来の OS は複数のネットワークを排他制御する機構を提供していない．従来の OS はネットワークの名前空間を 1 つしか持たず，ホスト上で利用可能な全てのネットワークが全てのプロセスに見えてしまう．そのため，ユーザはそれぞれのネットワークで使われている IP アドレス等が重ならないようにし，ネットワーク間のデータの流れに細心の注意を払う必要があった．このような問題を避けるために，従来の VPN は同時に 1 つだけ使うことを前提としている．

ユーザが必要に応じて自分のための VPN を構築できるようにするには，以下の 2 つの条件を満たす機構が必要である．

- VPN を他のネットワークから完全に分離する
- VPN の構築および利用が容易である

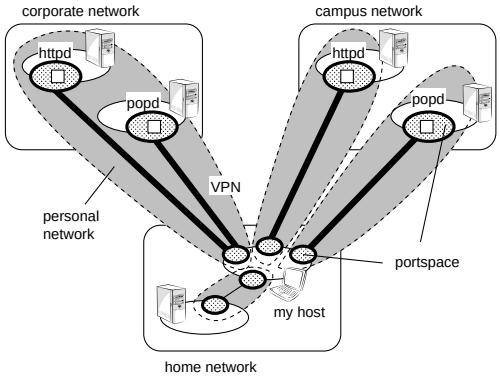

図 1: パーソナルネットワークによる VPN の排他利用

2.2　パーソナルネットワーク

そこで，我々は図 1 のように，複数の VPN を扱うホストの実行環境を VPN 毎に分離し，その実行環境と VPN を統合した**パーソナルネットワーク**を提案する．この分離された実行環境を**ポートスペース**と呼ぶ．ポートスペースはプロセスを実行する時のネットワークやファイルシステムなどの環境であり，その中でサーバプロセスやアプリケーションが動く．パーソナルネットワークはポートスペースと特定の VPN を結びつけることにより，ポートスペースを介して VPN と特定のプロセスを結びつける．

パーソナルネットワークは複数の VPN を 1 つのホストで排他的に扱うことを可能にする．パーソナルネットワーク内部のサーバは内部のアプリケーションからのみ利用することができ，外部からはアクセスできない．そのため，情報の流れをパーソナルネットワーク内部に限定することができる．また，パーソナルネットワークは他のパーソナルネットワークやベースネットワークから独立しているため，独自のネットワーク管理を行うことができる．例えば，パーソナルネットワークの中では独立した IP アドレス空間を提供できるため，内部のホストに対してベースネットワークと同じ IP アドレスを使わせることもできるし，全く別の IP アドレスを割り振ることもできる．

VPN 毎に独立した実行環境を作るために，ポートスペースはネットワークやファイルシステムの名前空間を分離する．ネットワーク空間を分離することにより，ポートスペースは VPN や IP アドレス，TCP や UDP のポートの利用の仕方などを独

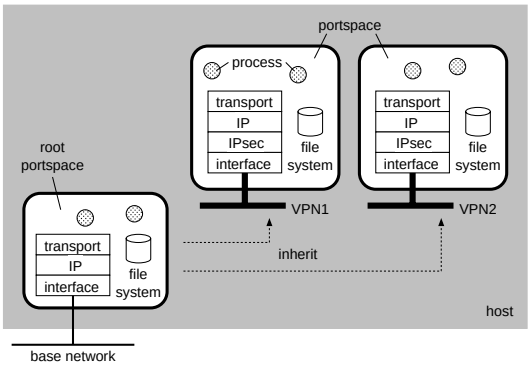

図 2: ポートスペースによるホストの実行環境の多重化

自に管理する．さらに，ポートスペースを作る時には親ポートスペースを継承して名前空間の一部を引き継ぐことができる．ベースレベルの従来の実行環境が全てのポートスペースの親となり，そこから分離して作られた実行環境が子ポートスペースとなる．ポートスペースの継承により，子ポートスペースは必要に応じて親ポートスペースで提供されているネットワークサービスやファイルシステムを利用できる．そのため，ネットワークやファイルシステムを一から構築しなくてよくなり，ユーザが簡単にポートスペースを作れるようになる．

3 設計

パーソナルネットワークはIPsec [5] を用いたVPNとホストの実行環境を多重化したポートスペースからなる．この章ではポートスペースを提供するPersona OS の設計およびパーソナルネットワークの管理について述べる．

3.1 ポートスペース

ポートスペースはプロセスを実行する際の暗黙の環境となり，プロセスは特に意識する必要はない．そのため，アプリケーションやサーバプログラムへの変更は必要ない．Persona OS ではポートスペースを実現するために，ネットワーク空間，ファイルシステム空間，プロセス空間を多重化する．Persona OS の全体像を図 2 に示す．

3.1.1 ネットワーク空間の多重化

ポートスペースはVPN 毎にホストのネットワーク空間を分離する．それぞれのポートスペースは独自のネットワークインタフェース空間，IPsec 空間，IP 空間，トランスポート空間を持つ．それぞれの空間は以下のような情報を管理する．

ネットワークインタフェース空間 IPsec のためのトンネルインタフェースとポートスペース用のループバックインタフェースを管理する．

IPsec 空間 どのポートスペース間の通信に IPsec を使うかというポリシーと確立された通信路を管理する．

IP 空間 IP アドレスやルーティングテーブルなどを管理する．

トランスポート空間 TCP や UDP などのトランスポートプロトコルについて，ポートへのサービスのバインドなどを管理する．

従来の IP アドレスの使い分けによるネットワーク空間の多重化とは違い，各ポートスペースは IP アドレスを自由に使うことができる．従来は IP アドレス毎に提供されるトランスポート空間を利用してネットワーク空間の多重化を行っていた．そのため，IP アドレスもポート番号も同じサービスは提供できなかった．それに対し，ポートスペースはIP 空間も多重化するため，ポートスペースが異なればIP アドレスもポート番号も同じサービスを提供することができる．これにより，1 つのサーバホストが複数のパーソナルネットワークに属する時でも，ユーザはどのパーソナルネットワークからでも同じ IP アドレスとポート番号を使ってアクセスでき，アクセシビリティを損なわない．また，どのようなIP アドレスを割り当てても他のパーソナルネットワークで使われる IP アドレスとは衝突しないため，IPsec における DHCP [6] を利用しやすい．

3.1.2 ファイルシステム空間の多重化

各ポートスペースのファイルシステムは独立しており，ポートスペース内のファイルやディレクトリは他のポートスペース内のプロセスからはアクセスできない．そのため，ファイルシステムを介してポートスペース間で情報を漏らさないようにすることができる．また，ユーザはポートスペースを作

る時にネットワークの設定ファイルなどのシステムファイルを自由に作成することもできる．

3.1.3 プロセス空間の多重化

プロセス空間も多重化され，プロセスは同じポートスペースの中で動いているプロセスしか見ることはできない．そのため，他のポートスペースのプロセスにシグナルを送ったり，IPC や共有メモリを使ったプロセス間通信を行うことはできない．

3.2 ポートスペースの継承

より簡単にポートスペースを作れるようにするために，ユーザは親ポートスペースの状態を継承させて新しいポートスペースを作ることができる．VPN を使わない既存の実行環境はルート・ポートスペースと呼ばれる擬似的なポートスペースとして扱われ，全てのポートスペースの親となる．ポートスペースの継承により，子ポートスペースは親ポートスペースで提供されているネットワークサービスやファイルシステムを初期実行環境として利用することができる．

図 3 のようにホスト B の親ポートスペースの TCP 80 番ポートを継承する場合を考える．ホスト A からホスト B の子ポートスペースの 80 番ポートにリクエストを送ると，そのリクエストは親ポートスペースの 80 番ポートに転送される．転送されたリクエストは親ポートスペースで 80 番ポートをバインドしているサーバプロセス（`httpd`）によって処理される．それに対するリプライは，リクエストが送られてきた子ポートスペースの通信路を使って返される．

子ポートスペースは親ポートスペースのサービスを上書きしたり，その一部を隠蔽することもできる．例えば，親ポートスペースの 80 番ポートでウェブサーバが動いていても，子ポートスペースの 80 番ポートで別のウェブサーバを動かすことができる．これにより，常にサービスのデフォルトのポート番号（HTTP なら 80 番）を利用でき，ユーザのアクセシビリティを低下させずに済む．一方，親ポートスペースの POP サービスだけを継承し，その他のサービスを隠蔽することにより，メール受信専用のポートスペースを作ることもできる．

また，ファイルシステムを継承することにより，

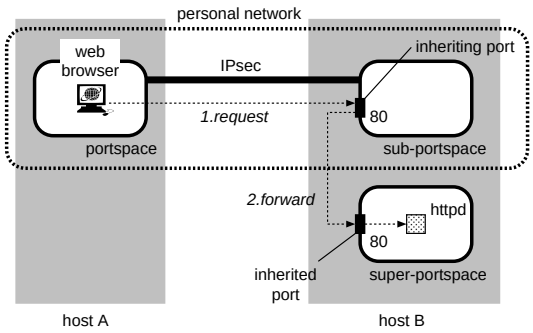

図 3: 継承した HTTP サービスへのリクエストの流れ

子ポートスペースから親ポートスペースのファイルを参照することができる．ユーザはルート・ポートスペースのファイルシステムの実行ファイルやライブラリを継承してポートスペースを作ることができるため，ポートスペースを作るたびに一からファイルシステムを構築する必要はない．子ポートスペースでファイルの内容を変更した場合やファイルを作った場合，ファイルを削除した場合には子ポートスペースにだけ反映される．

継承によりルート・ポートスペースは全てのポートスペースの雛型となり得るので，ルート・ポートスペースのファイルシステムには機密情報を置かないようにする．その代わり，LAN 内の機密情報はローカル・ポートスペースと呼ばれる特殊なポートスペースに置かれ，ユーザは必要に応じてこのポートスペースを継承する．従来の LAN は LAN 内の各ホストのローカル・ポートスペースから成るパーソナルネットワークで置き換えられ，LAN 内の機密情報はその中で扱われる．ただし，このパーソナルネットワークでは VPN は使われず，通常の通信を行うことができる．

3.3 ポートスペース間の通信

パーソナルネットワーク内のポートスペース同士は，ポートスペース間に確立されている IPsec の通信路を用いて通信を行う．パケットは IPsec トンネルモードを用いてカプセル化され，ベースネットワークを通って送られる．パケットをカプセル化することにより，パーソナルネットワークで使われる IP アドレスはベースネットワークから完全に隠蔽される．ポートスペースがパケットを送信する時に

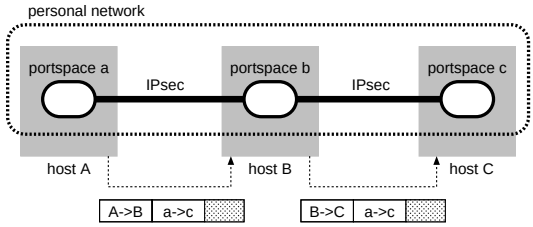

図 4: ポートスペース間のルーティング例

は，送信先のポートスペースの IP アドレスから適切な IPsec の通信路を選択する．パケットを受信したホストはパケットが送られてきた通信路から配送すべきポートスペースを選択する．

全てのポートスペース間に IPsec の通信路がメッシュ状に確立されているとは限らないので，パーソナルネットワーク内ではそのネットワークトポロジに応じた経路制御を行う．各ポートスペースはルーティングテーブルを持ち，全てのポートスペースがルータの役割を果す．従来の OS にはルーティングテーブルがシステムに 1 つしかなかったため，ベースネットワーク上の経路も VPN 上の経路も同じルーティングテーブルに入れられていた．Persona OS では各ポートスペースにルーティングテーブルを持たせることにより，それぞれの経路の衝突や誤用，不正利用を防ぐことができる．

各ポートスペースは IPsec の通信路を設定する時に相手のポートスペースへの経路を動的に生成する．経路情報はベースネットワーク同様に，経路情報プロトコル（RIP）を使って交換する．例えば，図 4 のようなパーソナルネットワークを構築した時，ポートスペース a のルーティングテーブルにはポートスペース b，c へのそれぞれの経路が作られる．ポートスペース a から c へパケットを送信する時は，パーソナルネットワーク用に使われる内側の IP ヘッダの送信元アドレスは a，送信先アドレスは c になる．カプセル化されたパケットはベースネットワークを通って配送されるが，その配送経路はベースネットワークのルーティングテーブルに従う．ホスト A からホスト C への直接の経路がない場合（ホスト B がルータでない場合），一旦ホスト A から B に送られた後，ホスト B から C に送られる．

3.4 パーソナルネットワークの管理

3.4.1 パーソナルネットワークの構築

ユーザは自分のホストのポートスペースに複数のリモートホストのポートスペースを結びつけることにより，自分のホストを中心としたパーソナルネットワークを構築することができる．さらに，直接通信できない組織内部のプライベートホストもパーソナルネットワークに加えることができる．そのためには，まず，組織の入口のホスト（出島ホスト) をパーソナルネットワークに加え，そのホストから内部のプライベートホストをパーソナルネットワークに加える．また，既に構築されたパーソナルネットワークに外部のホストから参加することもできる．この場合，パーソナルネットワーク内で登録されたユーザのみが参加を許可される．

3.4.2 パーソナルネットワークの制約

パーソナルネットワーク内に複数の組織の機密情報が含まれる場合，その中のポートスペースが親ポートスペースのサービスを継承していると情報が漏洩する危険性がある．継承したサービスを利用する時には親ポートスペースで動いているサーバプロセスにアクセスすることになるため，そのサーバプロセスは親子間で共有されていると考えることができる．さらに，そのサービスを継承する他のポートスペースとも共有することになる．もし，そのサーバプロセスがあるパーソナルネットワークの中で他の組織の機密情報を受け取ってしまうと，それ以外のパーソナルネットワークにもその機密情報を流してしまう可能性がある．

この問題を解決するため，チャイニーズウォールモデル [2] を用いてパーソナルネットワークに 1 つの組織の機密情報しか含まれないようにする．チャイニーズウォールモデルとは，ある組織の情報を見たユーザに他の組織の情報へのアクセスを禁止するモデルである．このモデルに基づき，パーソナルネットワークがある組織の機密情報を保持したポートスペースを含んでいる時には，他の組織の機密情報を保持するポートスペースは参加できないようにする．ポートスペースがどの組織の機密情報を保持しているかは親ポートスペースが属しているパーソナルネットワークから判断することができる．

4 実装

我々はポートスペースを提供する Persona OS を FreeBSD 4.7 をベースにして開発した．IPsec には FreeBSD 標準の KAME を用いた．本章ではポートスペースの実装とパーソナルネットワークの管理の実装について述べる．

4.1 ポートスペース

ポートスペースはネットワーク空間を多重化するために，OS カーネル内の以下のようなデータベースを独立して管理する．

- プロトコル・コントロール・ブロック（PCB）・リスト
- IPsec セキュリティポリシー・データベース（SPD）
- IPsec セキュリティアソシエーション・データベース（SAD）
- ルーティングテーブル
- ネットワークインタフェース・リスト

PCB は TCP や UDP のソケット毎に作られ，バインドされているポート等の情報を保持する．SPD は IPsec の適用ポリシーを保持し，SAD は通信路の情報を管理する SA を保持する．これらのデータベースを独立して管理するため，ポートスペース 1 つにつき，カーネルメモリを約 8K バイト消費する．

また，ファイルシステム空間を多重化するために，`chroot` システムコールと同様にプロセスにディレクトリツリーのサブセットを見せる．`chroot` とは違い，親ポートスペースからは分離されたファイルシステム空間が参照できないように，そのサブツリーに対する名前検索を禁止する．

ユーザは `mkportspace` システムコールを用いてポートスペースを作成することができる．ポートスペースを作成したプロセスとその子孫のプロセスがそのポートスペースの中で動く．ポートスペースの中で IPsec の設定を行うことで，IPsec の通信路を介して他のポートスペースと結合される．

4.1.1 ポートスペースの継承

ネットワークに関する継承は，ポートにバインドされているサービスを検索する時に，親ポートスペースの PCB も参照できるようにすることによって実現する．ポート番号から PCB を得る時には，まず，対象のポートスペースで PCB の検索を行い，そのポートスペースで PCB が見つからなければ親ポートスペースで検索する．これを見つかるまで繰り返し，最終的に見つからなければ検索に失敗する．ただし，継承せずに作られたポートスペースに到達した場合や，`hideport` システムコールによって特定のポートに関する継承を抑制している場合，その時点で検索に失敗する．

ファイルシステムに関する継承は，BSD のユニオンファイルシステムを利用して実現した．子ポートスペースでは / をマウントポイントとして，親ポートスペースの `/.filespace/<id>/`（`<id>`はポートスペースの ID）を透過的にマウントする．このマウントは `mkportspace` システムコールの中で行われ，このシステムコールを発行したプロセスが参照していたファイルやディレクトリは新しいファイルシステム上のものに変更される．子ポートスペースにおける読み出しは親ポートスペースのファイルシステムから行われ，変更は子ポートスペースのファイルシステムに反映される．従来のマウント機構では全てのプロセスに同じマウント状態を見せることしかできなかったので，ポートスペースに応じてマウント先を切り替えられるようにした．

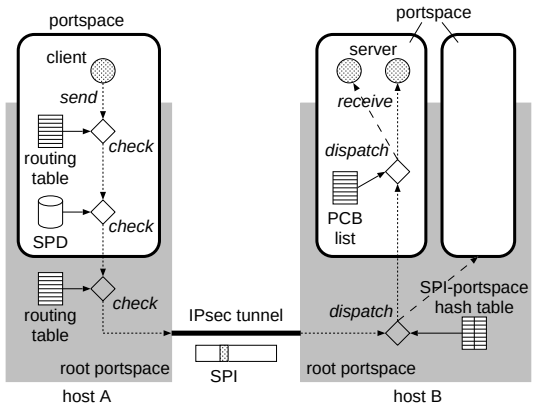

図 5: ポートスペース間の通信

4.1.2 ポートスペース間の通信

ポートスペース間の通信は図 5 に示すように行われる．ホスト A のクライアントプロセスが `send` システムコールを発行すると，カーネルはそのプロセ

スが動いているポートスペースのルーティングテーブルを検索する．そのルーティングテーブルに経路が見つかれば，次に IPsec の SPD からセキュリティポリシーを検索する．SPD に IPsec 通信を許可するポリシーが見つかれば，ポートスペース間で確立されている SA を用いて IPsec トンネルモードの処理を行い，パケットをカプセル化する．次に，カプセル化されたパケットをベースネットワークのネットワークトポロジに従って送信するために，ルート・ポートスペースのルーティングテーブルを検索する．このルーティングテーブルにも経路が見つかれば，カプセル化されたパケットを送信する．

このパケットをホスト B のカーネルが受け取ったら，パケットの IPsec ヘッダのセキュリティパラメータ・インデックス（SPI）をキーにして配送すべきポートスペースを検索する．SPI は IPsec の SA に割り当てられた 2 つのホスト間で一意の値である．ポートスペースが見つかったら，そのポートスペースが管理している PCB リストからパケットを受け取るべきソケットを検索し，見つかればパケットを配送する．

ポートスペースの継承の階層が深くなったとしても，このアルゴリズムがルート・ポートスペースに向かって再帰的に適用されるわけではない．ポートスペースは IPsec の通信路を継承しないため，常に一段のカプセル化しか行われない．

4.2 パーソナルネットワークの管理

各ポートスペースでは unite デーモン（`united`）が動いており，パーソナルネットワークを構築するために他のポートスペースとの間に IPsec の通信路を作成する役割を果す．

4.2.1 パーソナルネットワークの作成

ユーザは `unite` コマンドを用いて，自分のポートスペースを中心としたスター型のパーソナルネットワークを作ることができる．`unite` は図 6 のように自分のホストのポートスペース内で実行され，リモートホストのベースレベルで動いている `united` と通信して新しいポートスペースを作らせる．この通信を行う際，`unite` はポートスペース内で動いているのに対し `united` はベースレベルで動いているため，直接通信することができない．そこで，我々はネットワーク・ポート・トランスレーション（NPT）と呼ぶ機構を開発し，`unite` と `united` の間の通信を可能にした．NPT は IPsec の通信路が存在しないポートスペース同士の通信を親ポートスペースで中継して行う機構である．NPT についての詳細は 4.2.3 節で述べる．

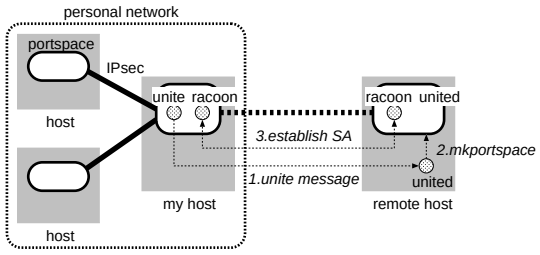

図 6: unite の処理の流れ

次に，これら 2 つのポートスペースの間に IPsec の通信路を確立するために，それぞれのポートスペースで IPsec のセキュリティポリシーを設定し，IKE デーモン `racoon` を起動する．`racoon` 同士がネゴシエーションすることにより SA を確立するが，SA を確立する前はまだ直接通信することはできないので NPT を用いる．

4.2.2 パーソナルネットワークへの参加

ユーザは `reunite` コマンドを用いて，既に作られているパーソナルネットワークに参加することができる．`reunite` コマンドは自分のホストのポートスペース内で実行され，参加する対象のパーソナルネットワークを指定する．現在の実装では，パーソナルネットワークの指定はそのパーソナルネットワーク内部のいずれかのホストのポートスペースを直接指定することによって行う．

`reunite` の処理は図 7 に示されるように，`unite` の場合とほとんど同じであるが，パーソナルネットワークへの参加の権限を確認するための認証を行う点が異なる．リモートホストの目的のポートスペースに IPsec のセキュリティポリシーを設定するために，目的のポートスペースの `united` をアップコールで呼び出して行わせる．この際に，認証をバイパスされないようにカーネルで認証を行なってからアップコールする．

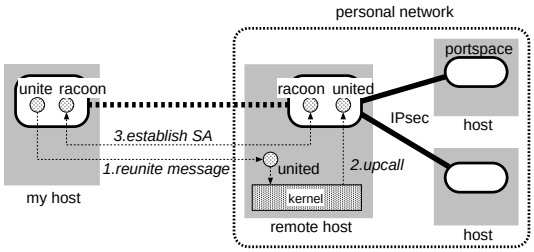

図 7: reunite の処理の流れ

4.2.3 ネットワーク・ポート・トランスレーション

ネットワーク・ポート・トランスレーション（NPT）は，子ポートスペースが親ポートスペースの通信路を用いて通信を行えるようにするために，子ポートスペースにおけるポート番号と親ポートスペースにおけるポート番号の間の相互変換を行う機構である．この機構は IP マスカレードに似ているが，⟨IP アドレス, ポート番号⟩ の代わりに ⟨ ポートスペース ID, 送信元ポート番号, 送信先ポート番号 ⟩ の組を変換する．ユーザは子ポートスペースの中から `setnpt` システムコールを使って，NPT の変換ルールを親ポートスペースの NPT テーブルに登録しておく．

子ポートスペースから送信されたパケットが親ポートスペースの NPT の変換ルールにマッチした時，パケットヘッダのポート番号を書き換え，親ポートスペースの通信路を使ってパケットを送信する．送信先のポートスペースでは，受信したパケットが NPT の変換ルールにマッチした時，パケットヘッダのポート番号を書き換え，パケットを子ポートスペースに送る．図 8 は UDP の 500 番ポート同士で通信する `racoon` が，親ポートスペースの通信路を使って通信する場合の例を示している．

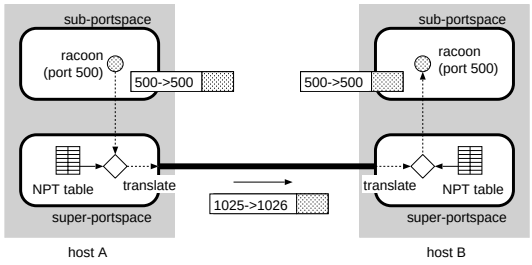

図 8: NPT を使った `racoon` 同士の通信

5 実験

我々の開発した Persona OS を用いてパーソナルネットワークのオーバヘッドを調べる実験を行った．実験には PentiumIII-S 1.4GHz の CPU，512MB のメモリ，Intel Pro/100+の NIC を搭載した PC を 2 台使用し，スイッチを介して 100baseT のイーサネットで接続した．IPsec のプロトコルには ESP を用い，暗号化と認証のオーバヘッドの影響を最小にして測定するために，NULL 暗号化アルゴリズムを用い，認証アルゴリズムは用いなかった．

ポートスペースに伴うオーバヘッドを調べるために，`netperf` ベンチマークプログラム [4] を用いて，TCP/IP と UDP/IP のそれぞれについて往復のレイテンシとスループットを測定した．この測定は（1）ベースネットワークで IPsec を用いた場合，(2）パーソナルネットワークの場合，(3）パーソナルネットワークで測定に用いるサーバを継承した場合の 3 つのネットワーク構成に対して行った．レイテンシの測定ではパケットのデータサイズは 1 バイトとした．

実験結果は表 1 のようになった．ポートスペースによってネットワーク空間を多重化したことにより，レイテンシが最大で 1.5%程度増大しているが，スループットの低下は 0.1%程度に抑えられている．一方，サーバ側で継承を用いることによるオーバヘッドはレイテンシでも 0.2%程度の増大にとどまっている．スループット測定時の CPU 利用率に関しては，ベースネットワークで IPsec を用いた場合は 14.5%であったのに対し，パーソナルネットワークでは継承の有無に関わらず 15.0%であった．

次に，Apache ベンチマークプログラム（`ab`）[1] を用いて，リクエストの並行度を変えながら thttpd ウェブサーバ [7] の性能を調べた．ネットワーク構成は上の実験で用いた 3 種類の構成を対象とした．リクエストにはポートスペースにおけるネットワーク処理に最も負荷をかけられように 0 バイトの HTML ファイルを用いた．

実験結果は図 9 のようになった．リクエストの並行度が 1 の場合はパーソナルネットワークにおけるウェブサーバの性能低下は 1.1%程度に抑えられているが，並行度が増してネットワーク処理の負荷が高くなると，性能低下は 3.9%程度に達する．一方，ウェブサーバを継承した場合に多少性能がよくなっているのは，継承したウェブサーバがベースレベル

表 1: ネットワーク構成別の往復のレイテンシ（μsec）とスループット（$Mbps$）

	TCP		UDP	
	レイテンシ	スループット	レイテンシ	スループット
ベースネットワーク+IPsec	132.40	91.13	126.63	94.00
パーソナルネットワーク	134.43	91.07	128.12	93.85
パーソナルネットワーク（継承あり）	134.75	91.04	128.34	93.80

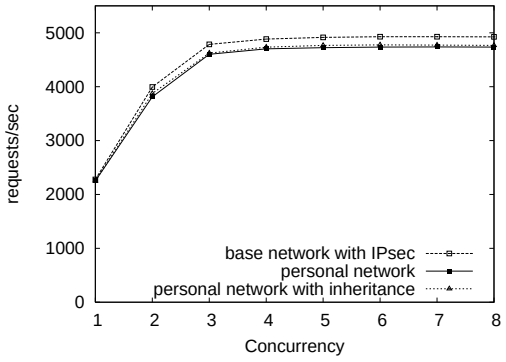

図 9: ネットワーク構成別の thttpd の性能

で動くため，ファイルの読み出しにオーバヘッドのかかるユニオンファイルシステムを使用しないためである．

6 関連研究

ホスト内に分離された実行環境を作るために，仮想的な環境を構築する様々な手法がある．FreeBSD の `jail` は特定のプロセスを専用の IP アドレスとファイルシステムの下で動かすことができる．User Mode Linux [3] などの仮想 OS や VMware [11] などの仮想マシンは，仮想環境毎に異なる OS を動かすことにより，仮想的なプロセスを完全に独立した名前空間の下で動かす．ただし，これらの仮想環境にはそれぞれ異なる IP アドレスを割り当てなければならず，IP アドレスを自由に使うことはできない．

また，これらの仮想環境はポートスペースと違い，ベースレベルの実行環境から仮想環境の IP アドレスやファイルシステムにアクセスできてしまう．これは設計思想の違いであり，従来の仮想環境は管理者がベースレベルから管理するという立場をとっている．それに対して，ポートスペースはベースレベルからも独立しており，管理者に作ってもらうのではなくユーザが作るという立場をとっている．我々はユーザが構築したパーソナルネットワーク内の情報は管理者にも見られるべきではないと考える．一般に，ある組織の管理者はパーソナルネットワーク内に含まれる他の組織の機密情報にまではアクセス権限を持たないためである．

Scandariato らのシステム [8] ではポートスペースと同様，VPN 毎にルーティングテーブルを持たせる．このシステムでは VPN の ID をネットワークインタフェースとソケットに設定することで，ソケット毎に利用するルーティングテーブルを切り替える．それに対して，ポートスペースはプロセスの暗黙の環境なので，ユーザプログラムを書き換えることなく，ルーティングテーブルを使い分けさせることができる．

仮想インターネット [10, 9] ではオーバレイネットワークと呼ばれる仮想ネットワーク毎にホストの環境が作られる．ポートスペースと同様に 1 つのホストが複数のネットワークを扱う時にそれぞれを別々に扱えるようにしている．ただし，パーソナルネットワークのように情報の流れを制限するものではなく，ホストの環境同士の間でルーティングすることもできる．

一方，パーソナル VPN [12] ではユーザ単位で複数の VPN を使い分ける．ユーザというのはアクセス制御でよく用いられる単位でありリーゾナブルであるが，同一のユーザであってもある VPN における権限と別の VPN における権限が同じとは限らない．ポートスペースではユーザをさらに VPN 毎に分割してアクセス制御することができる．

VNAP [13] もネットワークを多重化して使い分けを強制できるが，ネットワーク機器のサポートが必要な VLAN を使って，ネットワーク管理者が行うことを前提にしている．パーソナルネットワークでも LAN の中では IPsec の代わりに VLAN を利用す

ることにより，性能を良くすることが考えられる．

7 まとめ

本稿では，分離されたホストの実行環境とVPNを統合したパーソナルネットワークを提案した．ポートスペースと呼ばれるこの分離された実行環境は，ネットワークやファイルシステムについて独立した名前空間を提供し，VPNが特定のプロセスにだけ使われるようにする．これにより，パーソナルネットワークは他のネットワークから完全に独立する．また，ポートスペースは親ポートスペースの名前空間を継承することにより，ユーザがパーソナルネットワークを作るのを容易にする．

パーソナルネットワークは情報の流れを制限することを1つの目的としているため，ユーザは複数の組織の情報を同時に扱うことができない．しかし，組織の情報には絶対に漏らしてはならない社内の極秘事項からそれほど気にする必要のないグループの予定表まで含まれる．現在のところ，これらの重要度を意識せずに全て漏らしてはならない情報として扱っているが，情報の重要度などに応じてパーソナルネットワーク間での情報のやりとりを可能にしていく必要がある．

参考文献

[1] Apache HTTP Server Project, : Apache HTTP Server Benchmarking Tool, `http://www.apache.org/`.

[2] Brewer, D. and Nash, M.: The Chinese Wall Security Policy, in *Proceedings of the IEEE Computer Society Symposium on Security and Privacy*, pp. 206–214 (1989).

[3] Dike, J.: User Mode Linux, `http://user-mode-linux.sourceforge.net/` .

[4] Jones, R.: Netperf Benchmark, `http://www.netperf.org/netperf/NetperfPage.html`.

[5] Kent, S. and Atkinson, R.: Security Architecture for the Internet Protocol, RFC 2401 (1998).

[6] Patel, B., Aboba, B., Kelly, S. and Gupta, V.: DHCPv4 Configuration of IPsec Tunnel Mode, IPSEC Working Group Internet Draft (2001).

[7] Poskanzer, J.: Tiny/turbo/throttling HTTP Server, `http://www.acme.com/software/thttpd/`.

[8] Scandarioato, R. and Risso, F.: Advanced VPN Support on FreeBSD Systems, in *Proceedings of BSDCon Europe 2002* (2002).

[9] Touch, J. and Hotz, S.: The X-Bone, in *Proceedings of the 3rd Global Internet Mini-Conference at Globecom'98*, pp. 75–83 (1998).

[10] Touch, J., Wang, Y. and Eggert, L.: Virtual Internets, Technical Report ISI–TR–2002–558, Information Sciences Institute (2002).

[11] VMware, Inc., : VMware, `http://www.vmware.org/`.

[12] 宇崎央泰, 千葉滋, 光来健一：特定のユーザーだけが利用可能な仮想プライベートネットワーク, 日本ソフトウェア科学会第 19 回大会 論文集 (2002).

[13] 廣津登志夫, 福田健介, 明石修, 佐藤孝治, 山崎憲一, 菅原俊治：仮想データリンクを用いた多重通信クラスに関する一考察, 第 3 回インターネットテクノロジーワークショップ (2000).