特集●ソフトウェアシステム

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

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できるようになってきている．VPNはインターネットによって隔てられた組織の間に仮想的なネットワークを構築し，安全な通信を行えるようにする．VPNを利用することにより，ユーザはファイアウォールなどの存在を意識することなく，大きなLANの中にいるかのようにリモートホストにアクセスすることができる．

ユーザがVPNを自由に使えるようになるにつれて，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) とともに，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を利用するかもしれない．また，次第に普及してきているピア・ツー・ピア・ネットワークも広い意味でVPNの一種と考えることができる．

しかしながら，1つのホストでVPNを他のネットワークと同時に扱うことには問題がある．1つはそれぞれの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利用することができないことである．もう1つは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があることである．これらは各ホストの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に原因がある．

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特定のVPNを結びつけ

Personal Networks Integrating a VPN with Execution Environments of Hosts.

Kenichi Kourai, Toshio Hirotsu, Koji Sato, Osamu Akashi, Kensuke Fukuda, Toshiharu Sugawara, 日本電信電話株式会社 NTT 未来ねっと研究所, NTT Network Innovation Laboratories, NTT Corporation.
Shigeru Chiba, 東京工業大学, Tokyo Institute of Technology.

コンピュータソフトウェア, Vol.21, No.1(2004), pp.1–11.
[論文] 2003年3月4日受付.

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．この際に，何も設定されていない状態から実行環境を作り上げるのはユーザの大きな負担になるので，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．この機能により，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ることができる．

以下，2 章ではホストが複数の VPN を扱う際の問題と我々が提案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にこの問題が発生する可能性が高くなる．もし，それらのネットワークの中に同じ 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同じ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インターネット上の第三者に向けて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れば，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 ではネットワークの名前空間は 1 つしかなく，ホスト上の全てのネットワークが全てのプロセスに見える．そのため，ユーザはそれぞれのネットワークで使われる IP アドレス等が重ならないようにし，ネットワーク間の情報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．

ユーザが自由に VPN を構築して複数のネットワークを使い分けられるようにするには，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 の構築および利用を容易にする

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

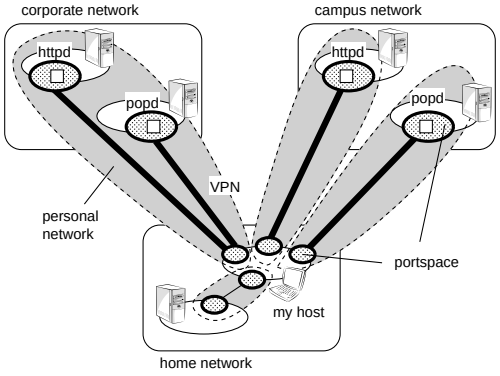

図 1 パーソナルネットワークによる VPN の排他利用

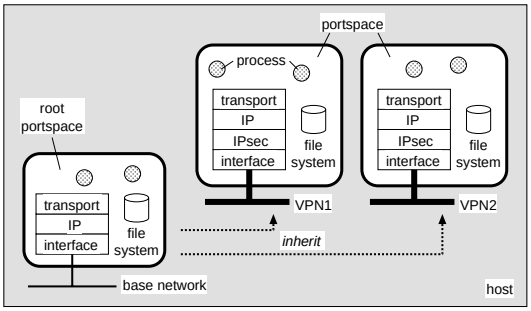

図 2 ポートスペースによるホストの実行環境の多重化

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，その中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，全く別の IP アドレスを割り振ることもできる．また，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のみから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に限定することができる．

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 のポートなどを独自に管理する．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，それから作られる実行環境が子ポートスペースとなる．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する必要がなく，ユーザが簡単にポートスペースを作ることができる．

3 設計

パーソナルネットワークは IPsec [7] 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．IPsec は IP 層にセキュリティを導入する仕組みであり，2 点間の通信の認証および暗号化を行う．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，プロセスはポートスペースの存在を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 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および，プロセス空間を多重化する．Persona 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情報を管理する．

ネットワークインタフェース空間 IPsec を用いてパケットをカプセル化するためのトンネルインタフェースと，ポートスペース毎のループバックインタフェースを管理する．

IPsec 空間 IPsec 通信をどのポートスペース間で許可するかというポリシーとポートスペース間で確立された IPsec 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，ネットワークポー

トへのサービスのバインドなどを管理する．

ポートスペースの特徴は IP アドレスを自由に使うことができる点である．従来のネットワーク空間の多重化は，IP アドレス毎に提供されるトランスポート空間によるものであったため，IP アドレスもポート番号も同じであるサービスを提供することはできなかった．それに対して，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であるサービスを提供することができる．このため，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に，どのパーソナルネットワークからでも同じ IP アドレスと同じポート番号でそのサーバホストにアクセスでき，ユーザのアクセシビリティを損なわずに済む．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の IP アドレスと衝突しないことが保証できるため，IPsec における DHCP [9] も利用しやすい．

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際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ができる．

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，各プロセスには同じポートスペース内で動いているプロセスしか見えない．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ったりすることは禁止される．

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

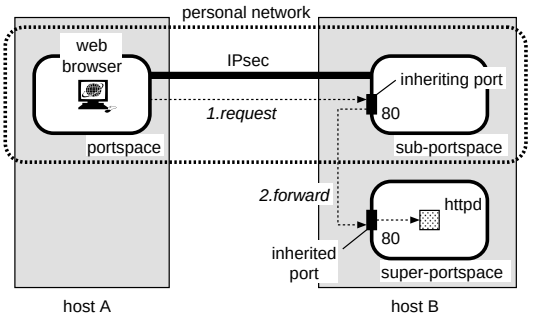

図 3 継承した HTTP サービスへのリクエストの流れ

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 番ポートのサービス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．このリクエストに対するリプライは子ポートスペースの通信路を使って返される．

継承に加えて，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，その一部を隠蔽したりすることもできる．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，常にサービスをデフォルトのポート番号 (HTTP なら 80 番) で動かすことができ，ユーザのアクセシビリティを低下させずに済む．また，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．

ファイルシステムを継承することにより，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に置かれた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作成・削除をした場合は，子ポートスペースのファイルシステムにだけ反映される．

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ポートスペースに置き，機密情報を扱う必要がある場合には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．

継承はポートスペースの独立性を弱めるが，いくつかの制約により安全性の低下を防ぐことができる．継承により親ポートスペースの資源は子ポートスペースからもアクセスされる．しかし，親ポートスペースを利用する権限を持つユーザだけに子ポートスペースの作成・利用を許しているため，安全性は損なわれない．一方，親ポートスペースのサービスを継承することにより，ポートスペース間で情報の流れが発生する可能性がある．この問題に対処するために，我々はパーソナルネットワークの構築に制約を設けている．この制約については 3.5.2 節で詳しく説明する．

3.3 ポートスペースのライフサイクル

基本的なポートスペースのライフサイクルは以下のようになる．ポートスペースはユーザによって作成され，その際に管理者権限は必要とされない．作成されたポートスペースの中ではポートスペースを作成したプロセスとそのプロセスが生成した子プロセスが動き，ポートスペース内の全てのプロセスが終了すればポートスペースも自動的に破壊される．それに加えて，ユーザは破壊されたポートスペースを後で復元することができる．ポートスペースによって多重化されたファイルシステム空間はポートスペースの破壊後も保持されており，ユーザは元のポートスペースの ID を指定して新たなポートスペースを作成することにより，そのファイルシステム空間を再利用することができる．さらに，同じ親ポートスペースを継承して同じネットワーク設定を行えば，元のポートスペースと同等の実行環境が復元される．

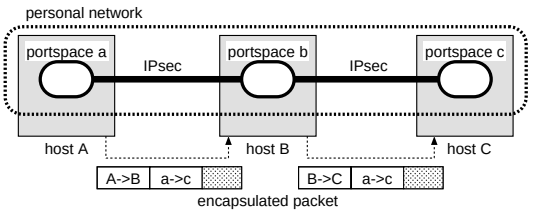

図 4 ポートスペース間のルーティング例

3.4 ポートスペース間の通信

パーソナルネットワーク内のプロセスは，ポートスペース間に確立された IPsec 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るため，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隠蔽される．ポートスペースがパケットを送信する時には，送信先のポートスペースの IP アドレスから適切な通信路を選択する．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．

IPsec 通信路は全てのポートスペース間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 はルーティングテーブルを 1 つしか持たなかったため，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 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．

各ポートスペースは IPsec 通信路を設定する際に相手のポートスペースへの経路を動的に生成する．経路情報は経路情報プロトコル (RIP) を使って交換される．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 と c への経路が作ら

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内で使われる内側の 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．一方，このパケットがカプセル化されてベースネットワークで配送される際に使われる外側の IP ヘッダの送信元・送信先アドレスは，ホスト A と B，B と C の間で書き換えられる．

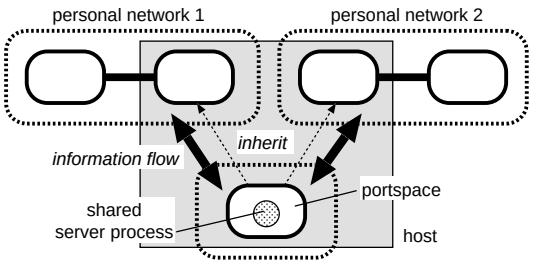

図 5 ポートスペースの継承に伴う情報の流れ

3.5 パーソナルネットワークの管理

3.5.1 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接続することにより，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．この際に，リモートの組織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こともできる．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加えればよい．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を作成したユーザのみが参加を許可される．

3.5.2 ネットワークの構築における制約

パーソナルネットワーク内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場合，機密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，このサーバプロセスは同じサービスを継承している他のポートスペースにも共有されることになる．もし，これらのポートスペースが属しているパーソナルネットワークがそれぞれ異なる組織の機密情報を扱っていたとすると，共有しているサーバプロセスを介して機密情報の流れが発生することになる (図 5)．

この問題を解決するために，チャイニーズウォールモデル [2] に基づいてパーソナルネットワークを構築する．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セキュリティモデルである．このモデルを強制することにより，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を参加させることができない．このようにして，同じサービスを継承しているポートスペースが属するパーソナルネットワークは全て，同じ組織の機密情報しか含まないようになる．その上，ある組織の機密情報を扱うパーソナルネットワークを作成できるのはその組織のユーザだけなので，情報の流れは組織の内部で閉じており，情報の機密性が保たれる．

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．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

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 は通信路の暗号化などの情報を管理するセキュリティアソシエーション (SA) を保持する．これらのデータベース毎に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．

4.1.1 ポートスペースの継承

ネットワークサービスの継承は，ポートにバインドされているサービスを検索する際に親ポートスペースの PCB リストも参照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の PCB リストを検索する．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．

ファイルシステムの継承は，BSD の union ファイルシステムを利用して実現した．子ポートスペースの `/` に対して，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．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いため，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．この操作は `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．子ポートスペースでの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．

4.1.2 ポートスペース間の通信

ポートスペースは独自に IPsec を設定することで他のポートスペースに接続することができる．ポートスペース間の通信は図 6 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

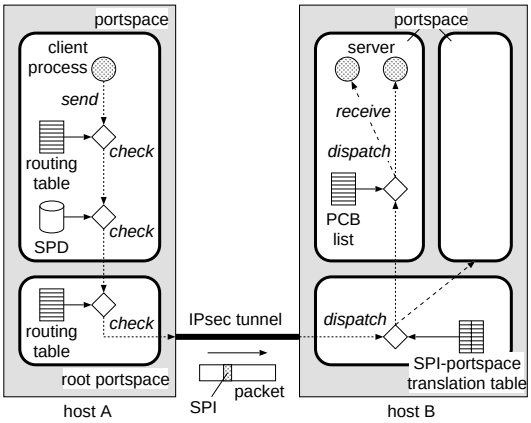

図 6 ポートスペース間の通信

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．経路が見つかれば，次に IPsec の SPD から適用ポリシーを検索する．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で送信するために，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．そして経路が見つかれば，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 に割り当てられる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 PCB を検索して対応するソケットを見つけ，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，常に一回だけカプセル化が行われる．

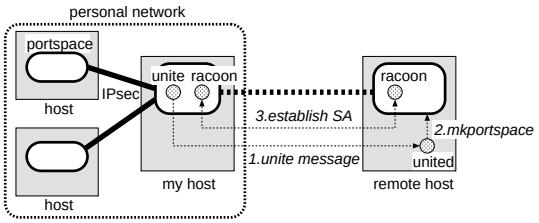

図 7　unite の処理の流れ

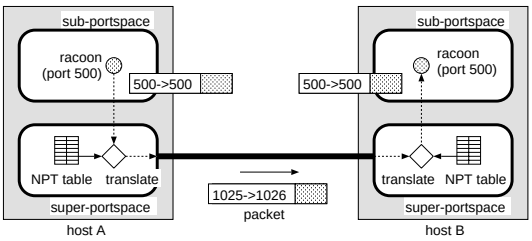

図 8　NPT を使った racoon 同士の通信

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 通信路を確立する役割を果す.

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 7 のように自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 **united** と通信してホストレベルのユーザ認証を行った後，**united** に新しいポートスペースを作成させる．この時点ではポートスペース内で動いている **unite** とリモートホストの **united** との間に通信路が存在しないため，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) と呼ぶ機構によりこのような通信を可能にした．NPT は IPsec 通信路が存在しないポートスペース間の通信を親ポートスペースで中継して行う．NPT についての詳細は 4.2.2 節で述べる.

次に，これらの 2 つのポートスペース間で IPsec 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 の適用ポリシーを設定し，IKE デーモン (**racoon**) を起動する．**racoon** 同士のネゴシエーションにより SA が確立されるが，ネゴシエーションが完了するまでの通信には NPT を用いる.

4.2.2　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・送信先ポート番号⟩ の組を変換する．この変換のために，ユーザは子ポートスペースの中で **setnpt** システムコールを使い，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転送する．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.

4.2.3　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，既に作成さ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内部のホストの IP アドレスとポートスペース ID の組を直接指定する.

reunite の処理は図 9 に示されるように **unite** の場合とほぼ同じであるが，ホストの管理者だけでなくポートスペースの作成者によっても参加の権限が確認される点が異なる．この認証にはチャレンジ&レスポンス方式を用い，バイパスされないようにカーネルで行う．まず，リモートホストの **united** はカーネルによって生成されたチャレンジを **reunite** に送る.

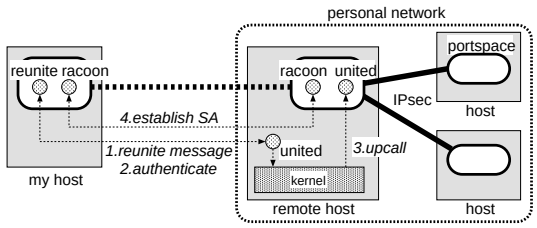

図 9 reunite の処理の流れ

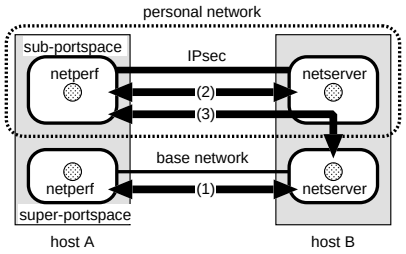

図 10 実験に用いた 3 つのネットワーク構成の模式図

reunite はそのチャレンジと目的のポートスペースを利用するためのパスワードからレスポンスを計算して送り返す．united がそのレスポンスをカーネルに渡し，カーネルは正しいレスポンスであれば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 IPsec の適用ポリシーを設定させる．アップコールは united が作成した専用のソケットにデータを送ることで実現している．

4.2.4 チャイニーズウォール

パーソナルネットワークにチャイニーズウォールモデルを適用するために，ポートスペースにどの組織の機密情報を保持しているかを表わすラベルをつける．現在の実装では，ラベルは組織に対応するサブネットアドレスとネットマスクから成る．組織の LAN 内の機密情報を扱うローカル・ポートスペースとそれを継承したポートスペースにはその組織を表わすラベルがつけられる．これらのポートスペースで扱われる情報を機密情報とそうでない情報とに自動的に分離するのは困難なため，現在はポートスペース内の全ての情報を機密情報として扱っている．その他のポートスペースにはヌルラベルがつけられる．ヌルラベルはどの組織の機密情報も保持していないことを表わす．

unite や reunite を実行する際に，パーソナルネットワークは接続される 2 つのポートスペースのラベルを比較することでチャイニーズウォールモデルを強制する．参加するポートスペースのラベルと参加されるポートスペースのラベルを比べて，一方がヌルラベルである場合，または，双方が同じラベルである場合にだけ参加を許可する．一方がヌルラベルであった場合には，機密情報の流れを考慮してそのラベルをもう一方のポートスペースのラベルにつけかえる．

5 実験

我々の開発した Persona 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 [6] 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 と UDP 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を用いた場合，(3) パーソナルネットワークで測定に用いるサーバ (netserver) 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 (図 10)．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．ポートスペースを用いることによって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 16%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ほぼ 20%であった．

次に，Apache ベンチマークプログラム (ab) [1] を

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

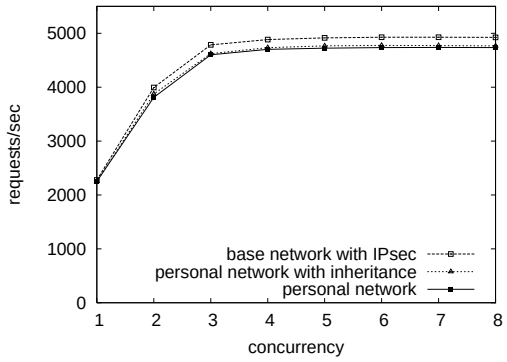

図 11　ネットワーク構成別の thttpd の性能

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 [10] の性能を調べた．ネットワーク構成は上の実験と同じ 3 つ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るように 0 バイトの HTML ファイルを用いた．

この実験結果は図 11 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で動き，ファイルの読み出しにオーバヘッドのかかる union 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` [5] 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．複製可能なネットワークスタック [14] はポートスペースと同様に，独立したプロトコルスタックとファイルシステムを提供する．User Mode Linux [3] などの仮想 OS や VMware [13] 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ない．

これらの仮想環境はポートスペースとは違い，ベースレベルの実行環境から仮想環境のサービスやファイルシステムにアクセスできる．これはシステム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ためである．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，作成者であるユーザが管理するという立場をとっている．我々はユーザが他の組織の機密情報を扱うために構築したパーソナルネットワーク内のサービスやファイルシステムには，システム管理者でもアクセスできるべきではないと考えている．一般に，ある組織のシステム管理者は他の組織の機密情報に対してアクセス権限を持たない．

1 つのホストで複数の VPN を使い分けられるようにする研究もいくつかある．仮想インターネット [12] では仮想ネットワークとホストの環境が結びつけられる．ホストの環境はポートスペースと同様に 1 つのホストが複数の仮想ネットワークを別々に扱うことを可能にする．ただし，仮想ネットワークとホストの環境の結びつきは変更可能なので，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のには向かない．

VNS [8] や Scandariato らのシステム [11] では，ルータにおいて VPN 毎に異なるルーティングテーブルを持たせる．これにより，複数の VPN で同じ IP アドレスを使うことが可能になる．パーソナルネットワークはこれらのシステムと違い，ルータ以外のホス

トでも複数の VPN を扱うことができる．ルータでは VPN 毎にルーティングテーブルを分離しさえすれば情報の流れを制御できるが，その他のホストではアプリケーションを介した情報の流れも考慮しなければならない．

パーソナル VPN [15] ではユーザ単位で複数の VPN を使い分ける．ユーザはアクセス制御でよく用いられる単位であるが，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 VPN 毎に分けてアクセス制御することができる．

VNAP [16] もネットワークを多重化して使い分けを強制するが，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 を統合する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組織間の情報の流れを制限することを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極秘事項からそれほど気にする必要のない予定表まで含まれる．現在のところ，このような重要度を意識せずに全てを機密情報として扱っているが，情報の重要度が低ければチャイニーズウォールモデルを緩めることにより，パーソナルネットワーク内で複数の組織の情報を扱うことも可能であると考えられる．一方，複数の組織の重要度の高い情報を同時に扱うには，個々の情報の流れを厳密に追跡する必要があり，アプリケーションや OS の大幅な変更が必要になると思われる．

謝辞

本稿の執筆にあたり有益な助言を下さった査読者の方々に感謝致します．

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 Symposium on Security and Privacy*, 1989, pp. 206–214.

[3] Dike, J.: A User-Mode Port of the Linux Kernel, *in Proceedings of the 4th Annual Linux Showcase & Conference*, 2000.

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

[5] Kamp, P. and Watson, R.: Jails: Confining the Omnipotent Root, *in Proceedings of the 2nd International SANE Conference*, 2000.

[6] Kent, S. and Atkinson, R.: IP Encapsulating Security Payload (ESP), RFC 2406, 1998.

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

[8] Lim, L., Gao, J., Eugene Ng, T., Chandra, P., Steenkiste, P. and Zhang, H.: Customizable Virtual Private Network Service with QoS, *Computer Networks*, Vol. 36,No. 2–3(2001), pp.137–151.

[9] Patel, B., Aboba, B., Kelly, S. and Gupta, V.: Dynamic Host Configuration Protocol (DHCPv4) Configuration of IPsec Tunnel Mode, RFC 3456, 2003.

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

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

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

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

[14] Zec, M.: Implementing a Clonable Network Stack in the FreeBSD Kernel, *Proceedings of the USENIX Annual Technical Conference*, 2003, pp. 137–150.

[15] 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.

[16] 廣津登志夫, 福田健介, 光来健一, 明石修, 佐藤孝治, 菅原俊治: 仮想ネットワークアーキテクチャによるネットワークワイドな保護機構, **情報処理学会論文誌**, Vol. 44,No. SIG11(2003), pp.180–190.