GSCIP の Windows への実装に関する検討

細尾幸宏

近年，イントラネット内で発生する不正アクセスや情報漏洩などの脅威に対するセキュリティへの関心が高まっている．既存のネットワークセキュリティ技術である IPsec はホストの移動などによるシステム構成の変化が頻繁に発生するような環境では管理負荷が高くなるという課題がある．そこで，我々は柔軟性とセキュリティを兼ね備えたネットワークアーキテクチャとして GSCIP（Grouping for Secure Communication for IP）を提案している．現在，GSCIP は IP 層を直接改造する方法で FreeBSD に実装されており，その有効性が示されている．今後，GSCIP の評価や普及を目指すうえで Windows への実装が不可欠であるが，FreeBSD と同様の方法で実装することはできない．そこで，本論文では GSCIP を Windows へ実装する方法を検討し，GSCIP の基幹プロトコルである DPRP （Dynamic Process Resolution Protocol）の実装と評価を行った．

A Study of Implementation of GSCIP for Windows

YUKIHIRO HOSOO

In recent years interest in anti-security measures for menaces such as illegal access and information leaks in intranet rise. As for IPsec that is existing network security technology, management load becomes higher in the environment where changing the system configuration by the movement of the host occurs frequently, and introduction is difficult. Therefore, we suggest GSCIP (Grouping for Secure Communication for IP) as the network architecture that had security and flexibility. GSCIP is implemented by a method to remodel the IP layer directly by FreeBSD, and the effectiveness has been shown now. Implementation to Windows will be indispensable to aim at the evaluation and the spread of GSCIP in the future, but it is impossible to mount by the method similar to FreeBSD. In this article, I examine a method to implement GSCIP to Windows and implemented DPRP (Dynamic Process Resolution Protocol) which is a key protocol of GSCIP and evaluate it.

1. はじめに

企業ネットワーク内で発生する不正アクセスや情報漏洩, 改ざんなどの被害が増加しており, イントラネット内でのセキュリティ対策が課題となっている. 外部からのアクセスに対してはファイアウォールなどの強固な対策が施されているが, イントラネット内部で発生する脅威に対するセキュリティ対策が今後さらに重要になる．

ネットワークセキュリティの代表的な既存技術として IPsec がある．IPsec は通信に先立ち暗号化や認証に必要な情報を動的に生成して安全な情報交換が可能な VPN を構築する．しかし，IPsec を利用するには多くの設定が必要であり，システム構成が頻繁に変化したり，通信グループの定義が個人単位と部門単位が混在したりするような環境では管理負荷が大きく，導入が難しい．

そこで我々はイントラネット内のセキュリティ対策と運用管理負荷の軽減を両立し, ホストがあらゆる空間を自由に移動することが可能なネットワークの概念として FPN（Flexible Private Network）の構築を目指している[1], [2]. また，FPN を実現する手段として GSCIP（Grouping for Secure Communication for IP）と呼ぶネットワークアーキテクチャを提案している．GSCIP はシステム構成の変化に動的に対応して動作処理情報を生成する動的処理解決プロトコル DPRP （Dynamic Process Resolution Protocol）[3]，通信中に移動して IP アドレスが変化してもエンド端末同士で通信の継続が可能な Mobile PPC（Mobile Peer to Peer Communication）[4]，グローバルアドレス空間からプライベートアドレス空間への通信開始を可能とする NAT-f（NAT-free Protocol）[5]などのプロトコル群によって構成され, 同一グループ間の通信は NAT やファイアウォールと共

存可能な暗号通信方式 PCCOM（Practical Cipher COMmunication）[6]によって暗号化を行う．

現在 GSCIP は FreeBSD に実装されており，有効性が証明されている．今後，評価や普及を目指すためには Windows への実装が不可欠である．本稿では GSCIP を Windows へ実装する方法について検討し，さらに GSCIP の基幹プロトコルである DPRP を実装し，評価実験を行ったので，報告する．

以下，第 2 章で FPN の概念と GSCIP，第 3 章で Windows への実装方法，第 4 章で Windows への DPRP の実装，第 5 章で実装された DPRP の性能評価，第 6 章でまとめについて述べる．

2. FPN と GSCIP

2.1 FPN

FPN とはネットワークのあるべき姿を示した概念である．FPN のグルーピングと通信の概念を図 1 に示す．FPN では個人単位と部門単位が混在する通信グループを構築できる．グループ内の通信はその安全性が保障され，異なる通信グループに属する端末や，通信グループに属していない端末からのアクセスを拒否することができる．FPN はこのようなネットワークにおいてさらに以下に示す位置透過性，移動透過性，アドレス空間透過性を実現したものである．

1．位置透過性

個々の端末や部門単位のサブネットは移動可能であり，端末が特定のサブネットの内外を往復するなどしてネットワーク構成が変化しても，あらかじめ定義されている通信グループの関係は維持される．このとき，ネットワークの管理者は設定情報を更新する必要はなく，システムが自動的にネットワーク構成の変化を学習する．この位置透過性は端末が通信を行っていない状態での移動を想定している．

2．移動透過性

端末が通信中の状態のまま移動することがある．このとき，端末の IP アドレスが変化するため，そのままでは通信を継続することができない．これは TCP や UDP を管理する情報に通信ペアの IP アドレスが含まれているためである．そのため，上位アプリケーションに対して IP アドレスの変化を隠蔽し，通信を継続できるようにする．これを移動透過性と呼ぶ．

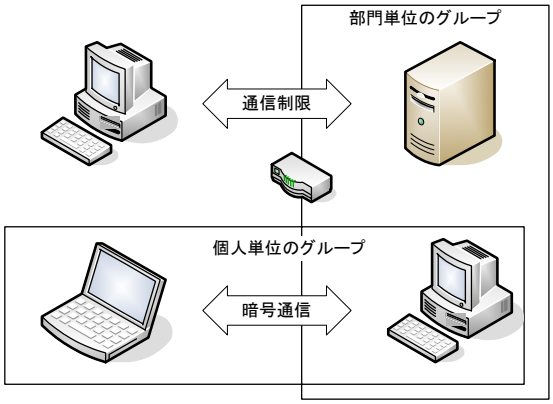

図 1　FPN のグルーピングと通信の概念

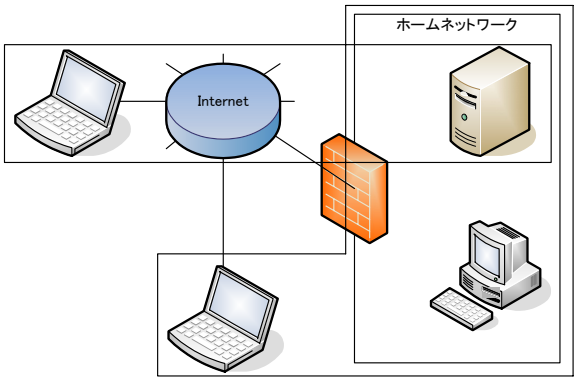

図 2　インターネット上の FPN

3．アドレス空間透過性

IPv4 環境ではプライベートアドレス空間とグローバルアドレス空間が存在し，両者の間では自由な通信ができない．これはアドレス変換装置 NAT によってプライベートアドレス空間がグローバルアドレス空間から隠蔽されるためである．端末と NAT が連携してアドレス空間の違いを意識することなく通信できるようにする．これをアドレス空間透過性と呼ぶ．

FPN の適用範囲はイントラネット内，およびホームネットワークを含むインターネット上の 2 つが想定され，システム構成に応じて管理付加を抑えられる．イントラネット内への FPN の適用は多段構成のネットワークにも対応でき，かつセキュリティも確保することができる．なお，企業ネットワークとインターネットとの間には強固なファイアウォールが設置され自由な通信ができないため，両者をまたがる FPN の構築は想定していない．一方，ホームネットワークのファイアウォールは企業のものほど強固ではなく，インターネットの延長に近い．そのため，図 2 に示すようにインターネットとホームネットワークをまたがった

FPN の適用によりグルーピングを実現させる．

2.2 GSCIP

GSCIP とは FPN を実現するための通信アーキテクチャである．GSCIP のグループ定義を図 3 に示す．GSCIP 対応機器を GE（GSCIP Element）と呼び，ホストタイプの GES，ルータタイプの GEN がある．GEN はサブネットを構成し，配下の一般端末を保護する．GSCIP では同一の暗号鍵を持つ GE を同一の通信グループとして定義する．この暗号鍵をグループ鍵 GK（Group Key）と呼び，同一の通信グループに所属する GE 間の通信はこの GK によって暗号化される．このように通信グループとグループ鍵を 1 対 1 に対応付けることで IP アドレスに依存しない通信グループを定義することができる．グループ定義は管理装置 GMS（Group Management Server）で定義され，GK は GMS からグループ情報と共に配送される．GK は定期的に更新されるほか，通信グループ内のシステム構成が変更されたときにも更新される．

GSCIP では通信に先立って動的処理解決プロトコル DPRP によって通信端末と通信経路上のすべての GE 間でグループ情報を相互に交換して，通信パケットの処理内容を決定し，動作処理情報テーブル PIT（Process Information Table）を生成する．PIT には送信元/宛先 IP アドレス，ポート番号，プロトコル番号，処理内容（暗号化/復号/透過中継/破棄），およびグループ鍵情報が記述されている．GE はパケット送受信時に自身が保持する PIT 検索し，記述されている動作処理情報に従ってパケットの処理を行う．

3. Windows への実装

Windows は TCP/IP モジュールを含む OS がブラックボックスになっており，FreeBSD に実装された GSCIP のように直接 IP 層を改造して実装を行うことができない．その代わりに，Windows には機能を拡張するために複数のインタフェースが外部に公開されている．GSCIP はこの中でネットワークの機能を拡張できる NDIS（Network Driver Interface Specification）に着目し，これを用いて FreeBSD の場合と同等の機能を実現することができる．

3.1 NDIS の概要

NDIS とは Windows カーネルのネットワークスタック内での処理手順などを規定したネットワークドライバの仕様と，それらドライバとのインタフェースを規定したものである．NDIS の概要を図 4 に示す．NDIS が規定する NDIS ドライバは図 4 中の中間ドライバやミニポートドライバを指し，NDIS インタフェースは NDIS ドライバ間の通信の中継やライブラリを提供する．NDIS はデータリンク層の機能の一部であり，NDIS ドライバはこのレベルで動作する．NDIS ドライバは通信時のパケットの送受信時に呼び出されて動作を行うだけでなく，ネットワークドライバとして必要な機能を実現するモジュール群として作成し，登録しておく．登録されたモジュールは NDIS インタフェースが所定の動作時に呼び出し，そこで動作を行う．GSCIP は中間ドライバとして実装し，IP 層の改造と同様の動作を実現する．中間ドライバは TCP/IP のようなプロトコルと NIC を操作するミニポートドライバの間でデータ転送を中継するように動作する．

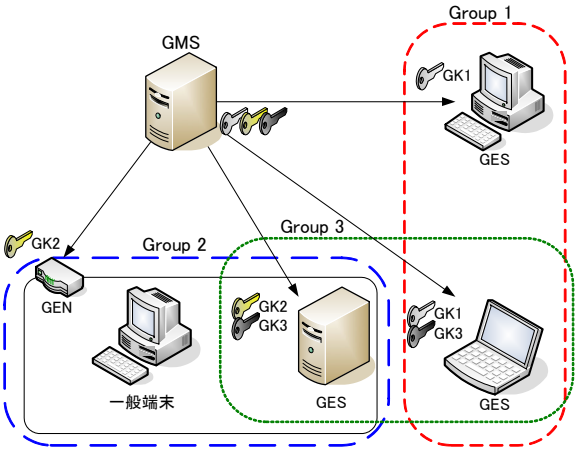

図 3　GSCIP のグループ定義

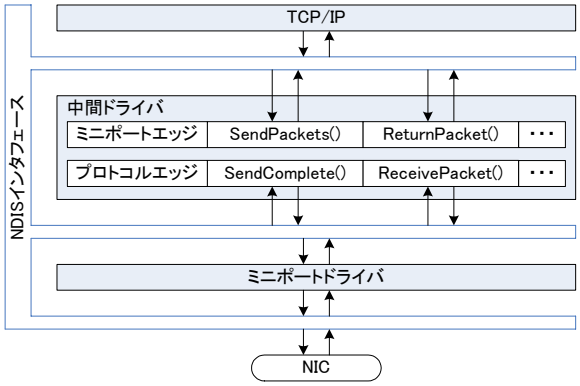

図 4　NDIS の概要

3.2 NDIS の送受信動作

NDIS ドライバにはパケット送受信時に

FreeBSD の IP 層にはない特有の送受信動作を行う．中間ドライバを介して行われる NDIS の送信動作を図 5 に，受信動作を図 6 に示す．プロトコルスタックの上位モジュールはパケットの送信を行った際，NDIS は中間ドライバの MiniportSendPackets()を呼び出す．中間ドライバはこのモジュール内で送信パケットに対する処理を行う．中間ドライバがパケットを中継あるいは独自に作成して送信する場合も同様にミニポートドライバの MiniportSendPackets()が呼び出される．パケット送信時，送信処理の成否に関する情報をすぐには受け取らず，別の処理を行うことができる．送信処理が終了すると，ミニポートドライバから結果の通知処理が実行され，NDIS は中間ドライバの ProtocolSendComplete()を呼び出す．この動作によって送信したパケットの処理結果を取得でき，同様の通知をさらに上位のモジュールに対して行う．これによって上位モジュールは処理の結果を順に取得する．

受信時はミニポートドライバが受信したパケットのメモリなどのリソースを管理し，パケットの受信を上位モジュールへ通知する．このとき，NDIS は中間ドライバの ProtocolReceivePacket()を呼び出す．中間ドライバはこのモジュール内で受信したパケットに対しての処理を行うことができ，さらにパケットの受信を上位モジュール通知する．上位の各モジュールはパケットに対する処理終了後にパケットへの処理終了を通知する．この動作によってミニポートドライバはパケットのリソースを開放する．

3.3 NDIS への実装概要

FreeBSD で開発した GSCIP のモジュールはほぼそのまま Windows へ流用可能であるが，Windows と FreeBSD で提供されている API の違いへの対応やデータリンク層で動作するために MAC ヘッダに対する処理の追加が必要である．また，データリンク層で送受信される全てのパケットに対して処理をするべく NDIS ドライバが呼び出される．しかし，GSCIP では動作対象を IP 層での処理を利用する TCP/UDP パケットに定めているため，処理対象パケットのフィルタリングを行う必要がある．また， 送受信時の NDIS 特有の結果通知処理などの動作に対応する必要がある．

GSCIP は通信開始時に DPRP によって通信相手とのネゴシエーションを行う．このとき，

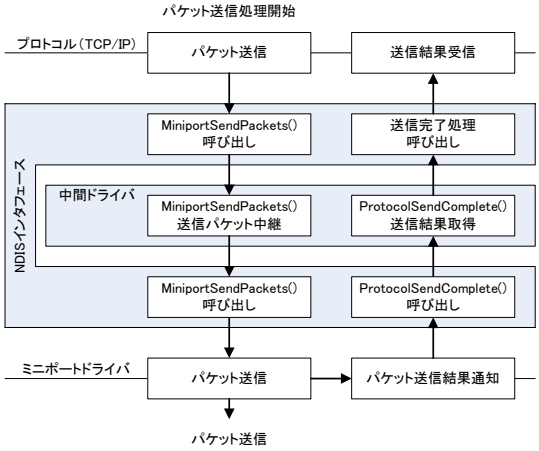

図 5　NDIS 送信処理手順

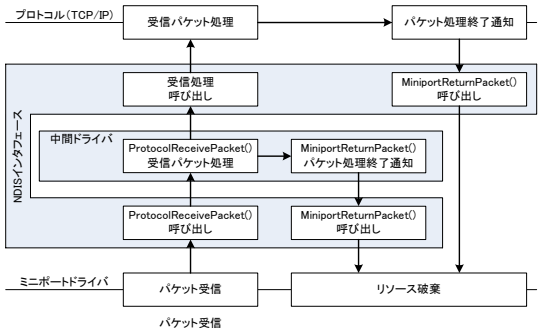

図 6　NDIS 受信処理手順

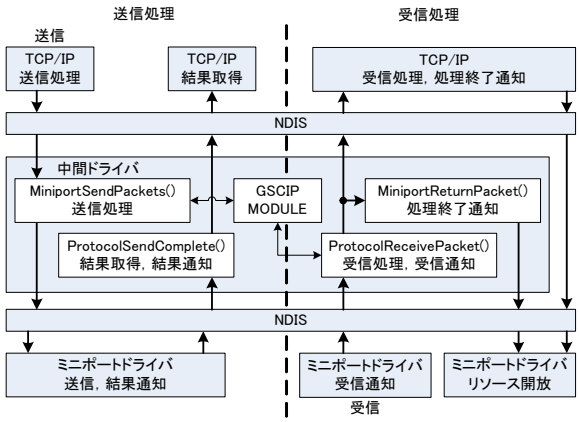

図 7　NDIS への実装

トリガとなった通信パケットを一時的にカーネル内に待避し，ネゴシエーションパケットを送信するが，このパケットは GSCIP が動作するスタックより上位モジュールにその送信結果を知らせる必要はない．また，上記ネゴシエーションパケットの送信完了通知を上位モジュールが受け取ると管理していないパケットの通知を取得したことに起因してクラッシュを起こす可能性がある．そこでネゴシエーショ

ンパケットについては, 送信完了通知処理時と受信処理時にパケットの判別を行い, 下位モジュールで全ての処理を完結させる必要がある.

上記をふまえたNDISへの実装を図7に示す. パケット送信時にはNDISからMiniportSendPackets()が呼び出される. ここでGSCIPモジュールを呼び出し, パケットの待避や暗号化, ネゴシエーションパケットの生成などの処理を行う. 送信処理終了後, ミニポートドライバから通知される処理結果をProtocolSendComplete()で取得する. ネゴシエーションパケットに関してはここで通知を破棄するが, その他の通信パケットは上位モジュールへ通知する.

パケット受信時にはNDISからProtocolReceivePacket()が呼び出されるので, ここからGSCIPモジュールを呼び出す. GSCIPモジュールは受信パケットがネゴシエーションパケットの場合は上位モジュールへ通知をせずにMiniportReturnPacket()を経由してMiniport Driverに処理終了の通知を行い, その他の通信パケットについては上位モジュールへ通知する. これらの動作によって本来の通信に影響を与えず, DPRPネゴシエーション処理を行うことができる.

4. DPRPの実装

4.1 DPRPの動作

図8にDPRPの動作を示す. GES1がGES2と通信を開始する際, まずPIT検索を行う. 該当するPITがない場合は通信パケットをカーネル内へ一時的に待避させ, DPRPネゴシエーションを行う. DPRPネゴシエーションはICMPベースのDDE（Detect Destination End GE), RGI（Report GE Information), MPIT（Make Process Information Table）およびCDN（Complete DPRP Negotiation）という4つの制御パケットを用いて行う. DDEにはトリガパケットの送信元/宛先IPアドレスとポート番号, プロトコル番号の組であるCIDをセットして通信パケットの宛先へ送信する. DDEを受信したGES2が終点GEとなり, RGIを生成する. RGIにはグループ鍵情報などの設定情報やネゴシエーションを行うGE間の認証を行う識別子をセットし, CIDの送信元IPアドレスへ送信する. RGIを受信したGES1が始点GEとなり, 収集した設定情報を元に動作処理情報を決定する. GES1は決定した自身に関する動作

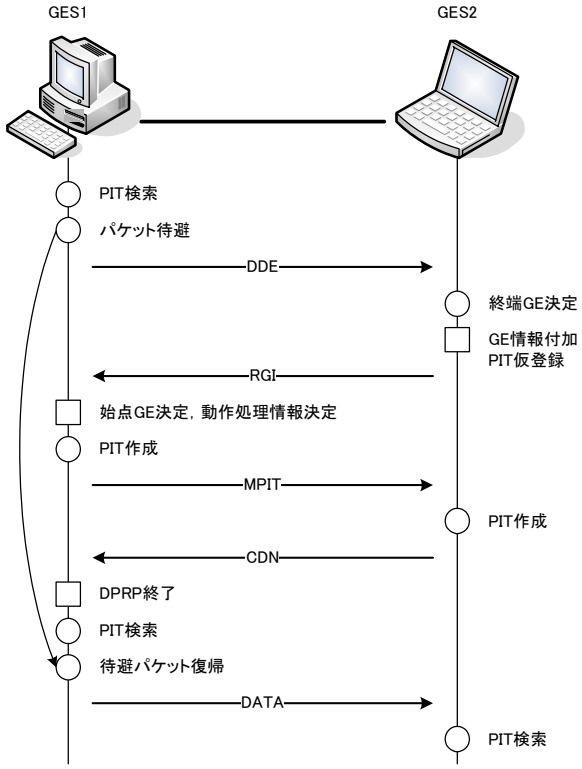

図 8　DPRPネゴシエーション

処理情報からPITを生成し, その他の動作処理情報をMPITにセットして終点GEへ向けて送信する. MPITを受け取ったGES2は記載されている動作処理情報からPITを生成する. PIT生成後, DPRPネゴシエーションの完了を通知するためのCDNを生成し, 始点GEへ向けて送信する. CDNを受信したGES1は待避していたパケットを復帰させ, ネゴシエーションによって生成されたPITに従って通信を開始する.

4.2 DPRPの実装

DPRPを実行する場合, ネゴシエーションのためのオリジナルパケットを作成する. NDISで扱うパケットはパケット記述子によって管理され, パケット記述子内にDPRPのネゴシエーションパケットであることを示す情報を付加しておく. これにより, パケット送信後に行われる送信完了通知時にパケットの判別を行い, 上位への通知の有無を判断する. ネゴシエーションパケットを受け取った場合, そのパケットの受信は上位モジュールへ通知しない. DPRPのネゴシエーションパケットはICMPパケットがベースになっているため, 上位モジュールへ通知してもICMPパケットとして処理されるだけだが, この処理は冗長であるため,

NDIS ドライバ以下で全ての処理を完結させる.

DPRP によるネゴシエーションが終了すると，待避したパケットを開放し，本来の通信が開始される．待避したパケットは GSCIP モジュール内から自身のパケット送信モジュール MiniportSendPackets()を呼び出して送信処理を行う．GSCIP では DPRP によって作成した PIT にしたがってパケットの暗号化/復号を行う.

5. 性能評価

Windows へ実装した DPRP の性能測定を行った．100BASE-TX の Ethernet において，GES1 と GES2 を直接接続し，FTP 接続を行った場合の DPRP の性能測定を行った．性能測定に使用した各装置の使用は CPU が Pentium4 2.4GHz, メモリが 1256MB である．DPRP ネゴシエーションのオーバヘッド時間を測定した．また，GSCIP では TCP/UDP パケットを送受信する際，必ず PIT 検索を行うため，通信性能に影響がある可能性がある．そのため，PIT 検索のオーバヘッドを調査するために GSCIP 実装時と未実装時の FTP スループットを暗号化しない状態で比較した．各 GE はあらかじめグループ番号とグループ鍵を保持しているものとした.

5.1 ネゴシエーションのオーバヘッド

オーバヘッドの測定にはデバッグ出力モニタツール DebugView を用いた．測定対象は図 9 に示す DPRP ネゴシエーション時間（DDE～CDN 間）[1]と，TCP の最初の SYN パケットが GES1 から送信されるまでの時間（通信開始までの時間）[2]である．オーバヘッドの測定結果を表 1 に示す．測定結果は DPRP ネゴシエーションを 5 回行った結果の平均値である．DPRP のネゴシエーション時間は 0.23 ミリ秒，通信開始までの時間は 0.25 ミリ秒となった.

5.2 FTP のスループット値

FTP のスループット値は Windows 標準のコマンドプロンプト上からの DOS コマンドによって FTP 接続を行い，表示される結果を採用した．測定方法は GES2 から 500MB のファイルをダウンロードした．測定結果は FTP によるダウンロードを 5 回行った結果の平均値である．DPRP 実装時と未実装時における FTP スループット値を表 2 に示す．DPRP 実装時では 92.33Mbps，DPRP 未実装時では 92.39Mbps となった.

これらの測定結果より，DPRP は通信に先立って行われるネゴシエーションであることを考えると TCP 通信にはほとんど影響を与えることがないといえる.

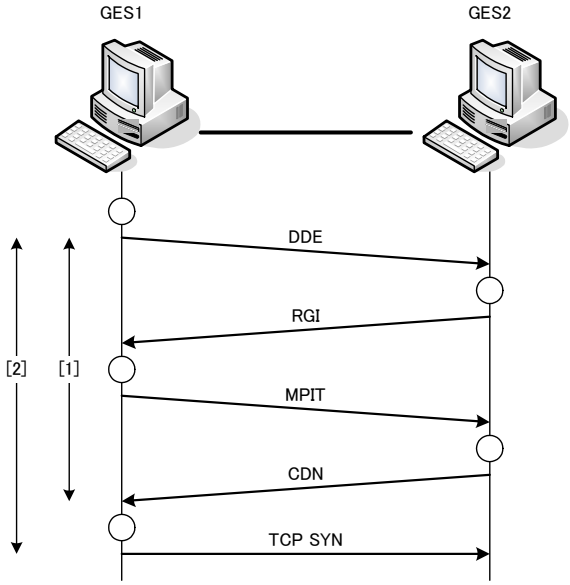

図 9　測定ポイント

表 1　オーバヘッドの測定結果

単位:ミリ秒

[1] ネゴシエーション時間	[2] 通信開始までの時間
0.22	0.24

表 2　FTP スループットの測定結果

単位:Mbps

	GSCIP実装時	GSCIP未実装時
スループット	92.33	92.39

FTP スループットでは GSCIP 実装時と未実装時の差は 0.06%程度であり，GSCIP による PIT 検索のオーバヘッドはほとんど通信に影響を与えることはないことがわかる．NDIS へ実装された GSCIP はデータリンク層で動作するため，UDP 通信に対しても同様の性能を得ることができる.

6. まとめ

FreeBSD に実装された GSCIP を Windows の NDIS を用いて実装する方法について述べた．基幹プロトコル DPRP の実装と評価を行い，DPRP が通信の開始時と通信中の双方において TCP/UDP 通信に影響を与えず実行できることを示した．今後は GSCIP の構成プロトコルである Mobile PPC, NAT-f および PCCOM の全ての機能を実装させ，Windows における GSCIP を実現する.

参考文献

[1] 鈴木秀和, 竹内元規, 加藤尚樹, 増田真也, 渡邊晃：フレキシブルプライベートネットワークを実現するセキュア通信アーキテクチャ GSCIP の提案，2005-DICOMO2005 シンポジウム.

[2] 名城大学理工学部，渡邊研究室：http://www.wata-lab.meijo-u.ac.jp/research/fpn1.html

[3] 鈴木秀和，渡邊晃：フレキシブルプライベートネットワークにおける動的処理解決プロトコル DPRP の実装と評価，情報処理学会論文誌，Vol.47，No.11，pp.2976-2991，Nov.2006.

[4] 竹内元規，鈴木秀和，渡邊晃：エンドエンドで移動透過性を実現する Mobile PPC の提案と実装, 情報処理学会論文誌, Vol.47, No.12，pp.3244-3257，Dec.2006.

[5] 鈴木秀和，宇佐見庄五，渡邊晃：外部動的マッピングにより NAT 越え通信を実現する NAT-f の提案と実装，情報処理学会論文誌，Vol.48，No.12，Dec.2007.

[6] 増田真也，鈴木秀和，岡崎直宣，渡邊晃：NAT やファイアウォールと共存できる暗号通信方式 PCCOM の提案と実装，情報処理学会論文誌，Vol.47，No.7，July.2006.

謝辞

本研究を行うに当たり，多大なるご指導，ご鞭撻を賜りました渡邊晃教授に心より感謝いたします．また，有益な助言および検討を頂きました渡邊研究室の皆様に深く感謝いたします．