GSCIP の Windows への実装に関する検討

細尾　幸宏†　　鈴木　秀和‡　　渡邊　晃†

名城大学理工学部†　　名城大学理工学研究科‡

1. はじめに

企業ネットワークのセキュリティを確保するためにグループを定義することは有効な方法である．しかし，IPsec のような既存の技術では通信グループの構成が頻繁に変化する場合や，個人単位や部門単位のグループ定義が混在した場合，それらに柔軟に対応するためには管理負荷が高くなり，導入が難しくなる．

我々は FPN（Flexible Private Network）と呼ぶ柔軟性とセキュリティを兼ね備えたネットワークの概念を提唱し，FPN を実現するためのネットワークアーキテクチャとして GSCIP（Grouping for Secure for Communication for IP）を提案している[1]．現在，GSCIP は FreeBSD に実装され動作検証を行っており，有効なアーキテクチャであることが確認されている．今後 GSCIP をより多くの人に利用してもらい，評価を受けるためには Windows に実装することが必須である．そこで本稿では GSCIP を Windows に実装する方法について検討した．

2. GSCIP

GSCIP では，共通暗号鍵と通信グループを 1 対 1 に対応付けることにより，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，IPsec に比べて大幅に管理負荷を軽減することができる．

GSCIP を構成するプロトコル郡として DPRP（Dynamic Process Resolution Protocol），Mobile PPC（Mobile Peer to Peer Communication）および NAT-f（NAT-free Protocol）がある．DPRP は通信に先立って通信経路上の GSCIP 対応装置が情報を交換し，認証や通信の可否を判断する処理を行う．Mobile PPC は通信中に一方の端末が移動した場合，エンド端末同士で IP アドレスの変化情報を交換し，端末の上位ソフトウェアに対して IP アドレスの変化を隠蔽し，通信の継続を実現する．NAT-f はプライベートアドレス空間にいる相手装置と通信を開始するとき，NAT-f に対応した NAT ルータに NAT テーブルを強制的に生成させることにより，プライベートアドレス空間とグローバルアドレス空間の違いを意識するとこのない通信を実現する．

現在，GSCIP は FreeBSD の IP 層に実装されており，基本動作を確認済みである．GSCIP モジュールは IP 層の一部を改造し，適切な場所から呼び出すサブルーチンとして実現されている．

3. Windows への実装

Windows は 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 層を改造して実装することができない．しかし，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）を用いて実現することができる．NDIS の概要を図 1 に示す．NDIS 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 が規定する NDIS ドライバは図 1 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

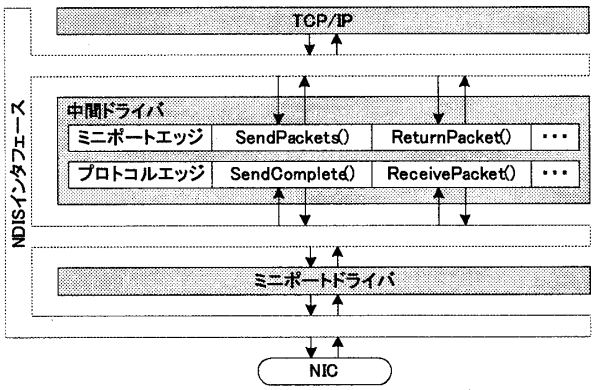

図 1　NDIS の概要

"A Study of Implementation of GSCIP for Windows"

† Yukihiro Hosoo and Akira Watanabe
Faculty of Science and Technology, Meijo University

‡ Hidekazu Suzuki
Graduate School of Science and Technology, Meijo University

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ドライバはここで動作する．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はなく，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，登録しておくことができる．登録されたモジュールはNDISインタフェースが決まった動作時に呼び出し，そこで動作を行う．

GSCIPは中間ドライバに実装し，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ドライバと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．

NDISドライバにはパケット送受信時にFreeBSDのIP層にはない特有の動作がある．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う際に送信処理の成否に関する情報をすぐには受け取らず，後で実行されるミニポートドライバからの結果通知処理によって結果を取得する．受信時はミニポートドライバが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．上位モジュールはそのパケットの処理終了をミニポートドライバに通知し，そこでリソースを開放する．

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，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，処理対象パケットのフィルタリングを行うなどの処理が必要になる．また，送受信時のNDIS特有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，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，上位モジュールは上記ネゴシエーションパケットの受信を通知されると管理していないパケットを受信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．

これらをふまえたNDISへの実装を図2に示す．

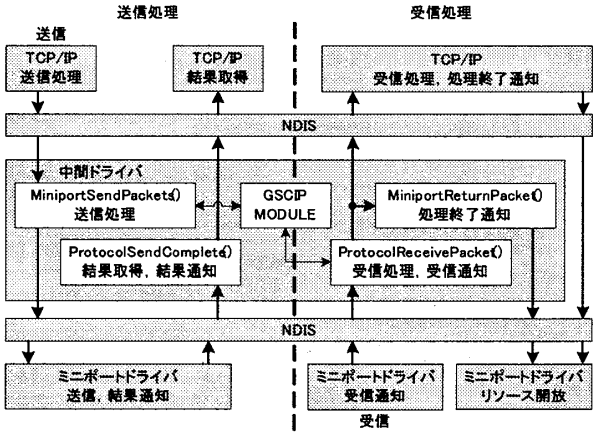

図2　GSCIPの実装と動作

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．

現在，GSCIPの基幹プロトコルであるDPRPの実装を完了し，基本的な動作を確認済みである．

4. 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の検討を行った．今後は全GSCIP機能の実装を完了させ，性能評価を行う．

参考文献

[1] 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.