GSCIP の Windows への実装に関する検討

細尾幸宏[†]　鈴木秀和[†]　渡邊晃[†]

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． 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，その有効性が確認されている．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．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[†]　HIDEKAZU SUZUKI[†]　AKIRA WATANABE[†]

Concerns about security against illegal access and risks of information leak within intranets have been rapidly increasing. IPsec has a problem that management loads get high in the environment where changes in the system configuration due to the relocation of the host, etc. occur frequently. Consequently, we are proposing "GSCIP" (Grouping for Secure Communication for IP) as a network architecture providing both flexibility and security. Currently, GSCIP has been implemented in FreeBSD in a manner directly modifying its IP layer, and the effectiveness of this system is already confirmed. Henceforth, it is indispensable to implement GSCIP in Windows in order to get a larger appraisal of this system and promote its widespread use. Accordingly, we propose in this paper the method of actually implementing GSCIP in Windows, and describe the results of our evaluation of implementing "DPRP" (Dynamic Process Resolution Protocol) (which is the core protocol of GSCIP) in Windows.

1. はじめに

企業ネットワークでは不正アクセスや情報漏洩，改ざんなど様々な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重要な課題となっている．外部からの不正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存在するが，イントラネット内部で発生する脅威に対しては適切なセキュリティ対策がないのが現状である．このような現状に対応するために通信グループを構築し，同一グループのメンバ間で行われる通信の安全性を確保することは有効な方法である．

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[1]．IPsec のトランスポートモードは個人単位での通信グループを構築し，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通信路を確立する．この方法は詳細な通信グループの定義が可能であるが，多くの設定が必要であり，規模が大きくなると管理負荷が大きくなる．トンネルモードはゲートウェイ間に安全な通信路を確立するが，きめ細かい通信グループを構築することができない．トランスポートモードとトンネルモードは互換性がないために通信グループの構成単位が個人単位と部門単位の混在するような環境では管理負荷が大きく，導入が困難である．

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）を提唱している．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 現在は FreeBSD に実装され，有用性が証明されている[2].

今後，GSCIP の普及を促進するためには Windows への実装が不可欠である．そこで本稿では 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 DPRP（Dynamic Process Resolution Protocol）[3]を実装し，評価実験を行ったので報告する．

以降，2 章で FPN と GSCIP について述べ，3 章で Windows への実装方法について説明する．4 章で性能評価実験の結果と評価について述べ，5 章でまとめる．

[†]名城大学大学院理工学研究科
Graduate School of Science and Technology, Meijo University

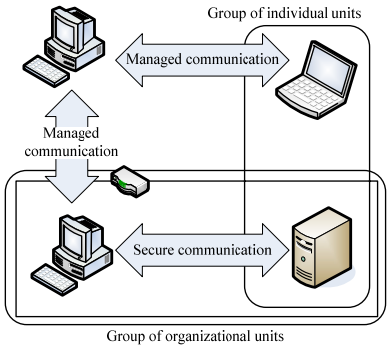


図 1　FPN のグルーピングと通信の概念

Fig.1　Grouping of FPN and concept of communication

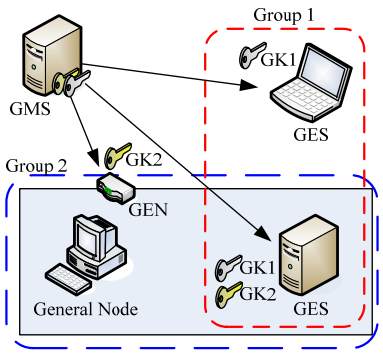


図 2　GSCIP によるグループ定義の方法

Fig.2　GSCIP's way of defining groups

2.　FPN と GSCIP

2.1　FPN の概要

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.

2.1.1 位置透過性

個々の端末やサブネットワークが移動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が維持される．このとき，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とによって，ネットワークの管理者は設定情報を更新する必要がない.

2.1.2 移動透過性

端末が通信中に移動すると端末の 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. これは上位ソフトウェアが IP アドレスを通信識別子として管理しているためである．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することにより移動しても通信を継続できるようにする.

2.1.3 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. これは一般には NAT 越え問題と呼ばれている. 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.

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．インターネット上への適用は，ホームネットワークを含む通信グループの構築が可能である．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 の構築は現時点では想定していない.

2.2　GSCIP の概要

GSCIP とは FPN を実現するための通信アーキテクチャで複数のプロトコルから構成されている．構成プロトコルには位置透過性を実現する DPRP，移動透過性を実現する Mobile PPC（Mobile Peer-to-Peer Communication）[4]，アドレス空間透過性を実現する NAT-f（NAT free Protocol）[5]，および NAT と共存可能な暗号通信方式 PCCOM（Practical Cipher COMmunication）[6]がある．GSCIP によるグループ定義の方法を図 2 に示す．GSCIP に対応した機器をここでは GE（GSCIP Element）と呼び, ホストタイプの GES（GE realized by Software）とルータタイプの GEN（GE for Network）がある．GES はソフトウェアとして各端末にインストールされる. GEN は配下にサブネットが存在し，サブネット内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 間はこの GK によって相互認証や通信の暗号化を行う.

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．通信グループの定義は管理装置 GMS（Group Management Server）で行う．グループ定義情報とそれに対応する GK は，GE の立ち上げ時に確実な認証の元で配送される．また，GK は定期的に更新される．

GSCIP では通信に先立って動的処理解決プロトコル DPRP を実行し，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する．これにより通信パケットの処理内容が決定され，各 GE 内に動作処理情報テーブル PIT（Process Information Table）が生成される．PIT にはコネクション ID（送信元/宛先 IP アドレス，ポート番号，プロトコル番号の組），処理内容（暗号化/復号，透過中継，破棄），およびグループ番号が記述されている．GE はパケット送受信時に自身が保持する PIT を検索し，記述されている処理内容に従ってパケットの処理を行う．

2.3 DPRP ネゴシエーションの処理内容

図 3 に DPRP の動作を示す．図 3 は 2 台の GES の通信経路上に 1 台の GEN が存在する場合を示している．GES1 が GES2 と通信を開始する際，まず自身の 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に一時的に待避し，DPRP ネゴシエーションを開始する．DPRP ネゴシエーションには ICMP 上で定義された 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 つの制御パケットが用いられる．

ネゴシエーションを開始する GES1 は終端 GE+を決定するため DDE を GES2 に向けて送信する．DDE にはトリガパケットのコネクション ID をセットして通信パケットの宛先へ送信する．DDE の宛先が GES の場合はその GES が終端 GE になる．もし DDE の宛先が一般端末だった場合，一般端末からの応答パケット ICMP ECHO REPLY を最初に受信した GE が終端 GE となる．次に，RGI によって始点 GE の決定と GE 設定情報の収集を行う．RGI にはグループ番号をセットし，送信元 IP アドレスへ送信する．RGI を中間 GE（GEN）が受信すると，RGI の内容に自身のグループ番号を追加していく．RGI の宛先となっている始点 GE（GES1）は RGI により報告された情報を元に各 GE の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処理情報から PIT を仮生成し，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 を受け取った各 GE は記載されている

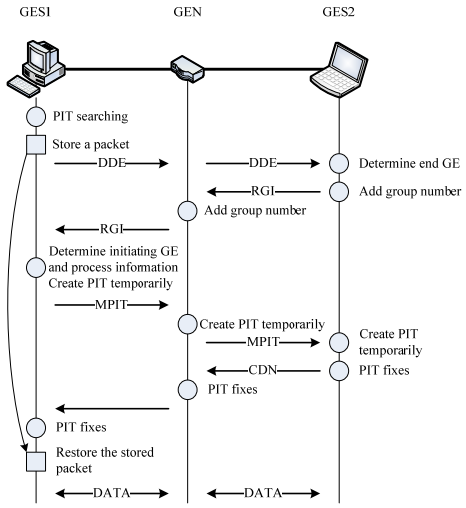


図 3　DPRP の動作

Fig.3　Processing sequence of DPRP

動作処理情報を自身の PIT に仮登録する．終端 GE は PIT 生成後，DPRP ネゴシエーションの完了を通知するため CDN を始点 GE へ向けて送信する．CDN を受信した各 GE は PIT の内容を確定させる．始点 GE は待避していたパケットを復帰させ，生成された PIT に従って通信を開始する．以降のデータ通信は各 GE において PIT の内容に従って処理される．DPRP により，通信相手の認証，通信可否の判定，使用する暗号鍵を決定することができる．

3. Windows への実装

GSCIP はすでに FreeBSD の IP 層を改造する方法で実装されており，その動作は確認済みである．しかし，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 のように IP 層を直接改造することができない．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．本論文で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，FreeBSD の場合と同等の GSCIP 機能を実現する．

3.1 NDIS の概要

NDIS とは Windows カーネル内でのネットワークドライバ処理手順と，外部モジュール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 ドライバは中間ドライバとミニポートドライバから構成されている．NDIS インタフェースは NDIS ドライバによる通信の中継機能やライブラリを提供す

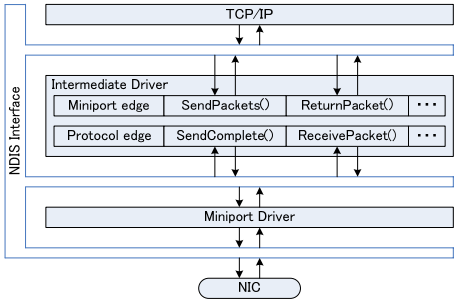


図 4　NDIS の概要

Fig.4　Outline of NDIS

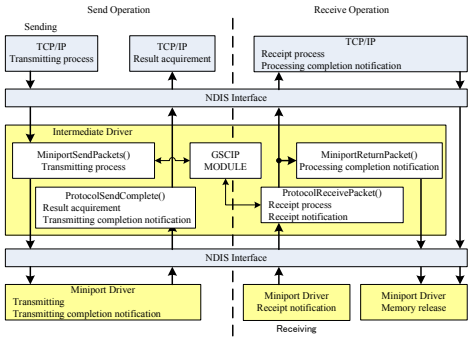


図 5　NDIS の送受信動作

Fig.5　Transmission/Reception processing of NDIS

る．NDIS ドライバは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インタフェースから所定の動作時に呼び出される．

3.2　NDIS のパケット送受信処理

NDIS ドライバ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 の送受信動作を図 5 に示す．プロトコルスタックの上位モジュールがパケットの送信を行うと，NDIS は中間ドライバの MiniportSendPackets()を呼び出す．このモジュールは上位から受け取った送信パケットを下位のミニポートドライバへ中継する．この中継時に送信パケットに対する処理を行うことができる．中間ドライバが内部でパケットを独自に生成して送信する場合も同様に MiniportSendPackets()を呼び出すことでパケットを送信することができる．送信処理を行った各モジュールはFreeBSDにおいてはその処理結果を取得するまで待

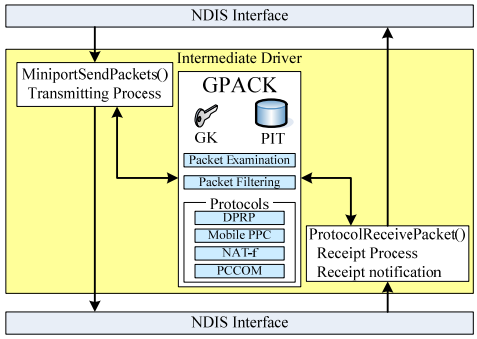


図 6　GSCIP の実装

Fig.6　Implementation of GSCIP

機し，結果取得後に次の動作を行う．しかし，NDIS の送信処理では送信を行ったモジュールは結果取得を待たず，次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が報告され，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し，さらに上位のモジュールに対して通知する．これにより，送信動作に係わったモジュールが処理の結果を非同期で取得する．

受信時はミニポートドライバが受信パケットのメモリを確保し，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．このモジュールは下位からのパケット受信の通知を上位モジュールへ中継する．この時に受信したパケットに対する処理を行うことができる．受信時の通知を受け取り，中継や処理を行ったモジュールは通知や受信パケットに対する処理を終えると処理終了をミニポートドライバに通知する．ミニポートドライバはこの通知を受けて受信パケットのメモリを開放する．

3.3　GSCIP の移植

FreeBSD で開発した GSCIP を実現するためのモジュール群を GPACK (Gscip PACKage) と呼び，PIT や DPRP などの GSCIP を構成するプロトコルをこの GPACK に集約している．GPACK は対象パケットに対する処理の有無や処理内容の判断を行い，必要に応じて構成プロトコルを呼び出して処理を実行させる．図 6 に示すように，GPACK は中間ドライバの一部に組み込まれる．GPACK はほぼそのままの形で Windows へ流用可能であるが，Windows と FreeBSD で提供されている API の違いに関わる修正や MAC ヘッダに対する処理の追加が必要である．また，NDIS ドライバはデータ

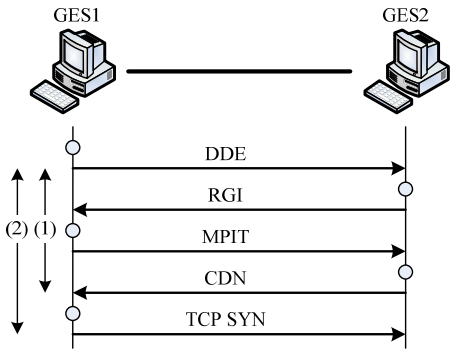


図 7　測定ポイント
Fig.7　Measurement points

リンク層で送受信される全てのパケットごとに呼び出されるが，GSCIP は TCP/UDP パケットに対してのみ処理を行うため，処理対象外のパケットのフィルタリングを行う必要がある．さらに，送受信時の NDIS 特有の結果通知処理に関わる修正が必要である．

GSCIP は通信パケットに対応する PIT がないとき DPRP を実行する．DPRP のネゴシエーション用制御パケットは GSCIP 内で生成される．ここで，生成した制御パケットの送信完了通知を上位モジュールに通知すると，管理していないパケットの通知を取得したとしてカーネルがクラッシュを起こす可能性がある．このため，制御パケットについては送信完了通知処理時に通知された情報から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．

具体的な処理は以下のとおりである．パケット送信時には MiniportSendPackets()から GPACK を呼び出す．GSCIP では PIT がある場合は PIT に従った処理（暗号化など）を行い，PIT がない場合はパケットを待避し，DPRP 処理を実行する．このとき，DPRP 制御パケットにはパケット記述子に，DPRP の制御パケットであることを示す情報を付加しておく．パケット記述子とはパケットの属性などの情報を集積したものである．送信処理終了後，処理結果を ProtocolSendComplete()にて取得するが，制御パケットに関してはここで通知を破棄し，その他の通信パケットは上位モジュールへ通知を中継する．

パケット受信時には ProtocolReceivePacket()から GPACK を呼び出す．受信パケットが制御パケットの場合は DPRP によるパケットへの処理後，MiniportReturnPacket()を経由してミニポートドライバに処理終了の通知を行う．一般の通信パケットについては PIT に従った処理（復号など）を実行後，上位モジュールへ通知する．

表 1　オーバヘッドの測定結果
TABLE 2 Measurement results of the overhead

[1] DPRP Negotiation time	[2] time until the communication starts
0.22 msec	0.24 msec

表 3　FTP スループットの測定結果
TABLE 4 Measurement results of the throughput of FTP

	With GSCIP	Without GSCIP
Throughput	92.33 Mbps	92.39 Mbps

4.　評価

GSCIP の基幹プロトコルである DPRP と GPACK の機能を Windows に実装し，所定の動作を実行することを確認した．100BASE-TX の Ethernet において，GES1 と GES2 を直接接続し，FTP 接続による性能測定を行った．性能測定に使用した装置は CPU が Pentium4 2.4GHz，メモリが 1GB である．

4.1　DPRP 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 7 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．ネゴシエーション時間は 0.22 ミリ秒，通信開始までの時間は 0.24 ミリ秒となった．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．

4.2　FTP 接続のスループット

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，通信性能に影響を与える可能性がある．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 スループットを比較した．同等の条件とするため，暗号化等は行わず平文での通信とした．

FTP のスループット値は 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，測定結果は 5 回行った結果の平均値をとった．表 2 に測定結果を示す．GSCIP 実装時では 92.33Mbps，DPRP 未実装時では 92.39Mbps となった．GSCIP 実装時と未

実装時の差は0.06%程度であり， 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．

5. まとめ

FreeBSD に実装されていた GSCIP を Windows へ移植する方法について述べた．GSCIP の基幹プロトコル DPRP の実装と評価を行い，通信開始時のオーバヘッドは十分小さいことを示した．また，一般通信においても PIT 検索にかかわる処理は十分小さく通信中の TCP/UDP 通信に影響を与えないことを示した．GSCIP には他に移動透過性を実現する Mobile PPC, NAT 越えを実現する NAT-f などのプロトコルがあり，順次 Windows への移植を行う予定である．

謝辞 本研究の一部は，日本学術振興会科学研究費補助金（特別研究員奨励費 20・1069）の助成を受けたものである．

参考文献

[1] Kent, S. and Seo, K.: Security Architecture for the Internet Protocol, RFC 4301, Dec. 2005.

[2] 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, マルチメディア，分散，協調とモバイル（DICOMO2005）シンポジウム論文集，Vol.2005, pp.441-444，Jul.2005.

[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.

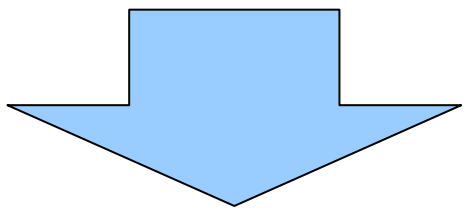
[6] 増田真也，鈴木秀和，岡崎直宣，渡邊晃：NAT やファイアウォールと共存できる暗号通信方式 PCCOM の提案と実装, 情報処理学会論文誌, Vol.47, No.7，July.2006.

GSCIPのWindowsへの実装に関する検討

名城大学大学院理工学研究科
細尾 幸宏

背景

- ユビキタスネットワークの普及
  - セキュアな通信
  - 移動しながらの通信
  - どこからでも自由なアクセス



柔軟性とセキュリティを兼ね備えたグループ通信を実現する

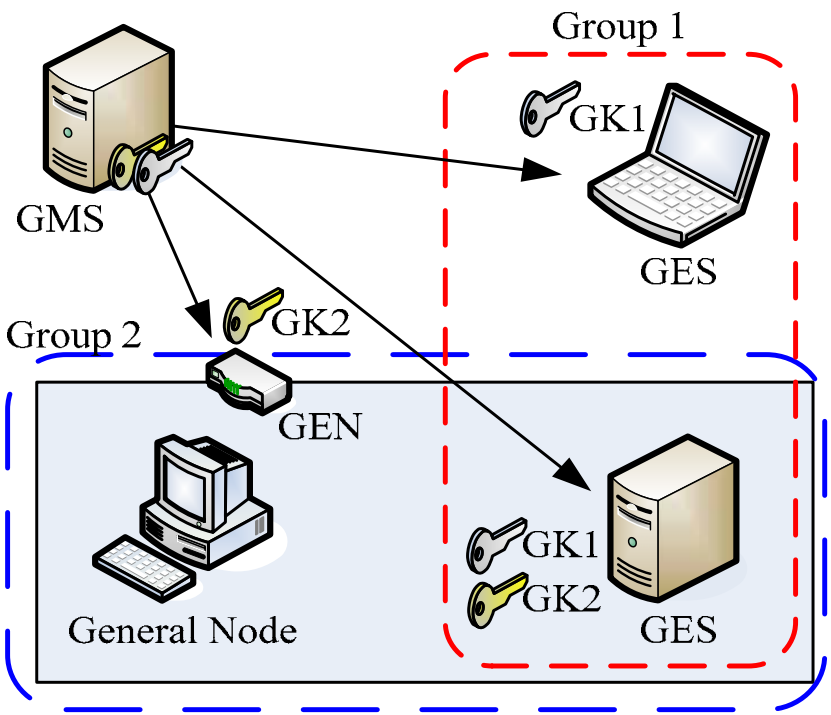
GSCIP (Grouping for Secure Communication for IP)

GSCIP

- **GSCIPの実現する機能**
  - **ネゴシエーションと位置透過性**
    - DPRP (Dynamic Process Resolution Protocol)
      - **通信相手と経路上にあるGEに対してネゴシエーションと認証**
      - **ネットワークの構成変化に動的に対応**
  - **移動透過性**
    - Mobile PPC (Mobile Peer to Peer Communication)
      - **IPアドレスの変化を隠蔽し，移動通信をエンドエンドで実現**
  - **アドレス空間透過性**
    - NAT-f (NAT – free Protocol)
      - **対応NATルータに外部から強制的にNATテーブルを生成し，アドレス空間の違いを意識しない通信をエンドエンドで実現**

GSCIPのグルーピング

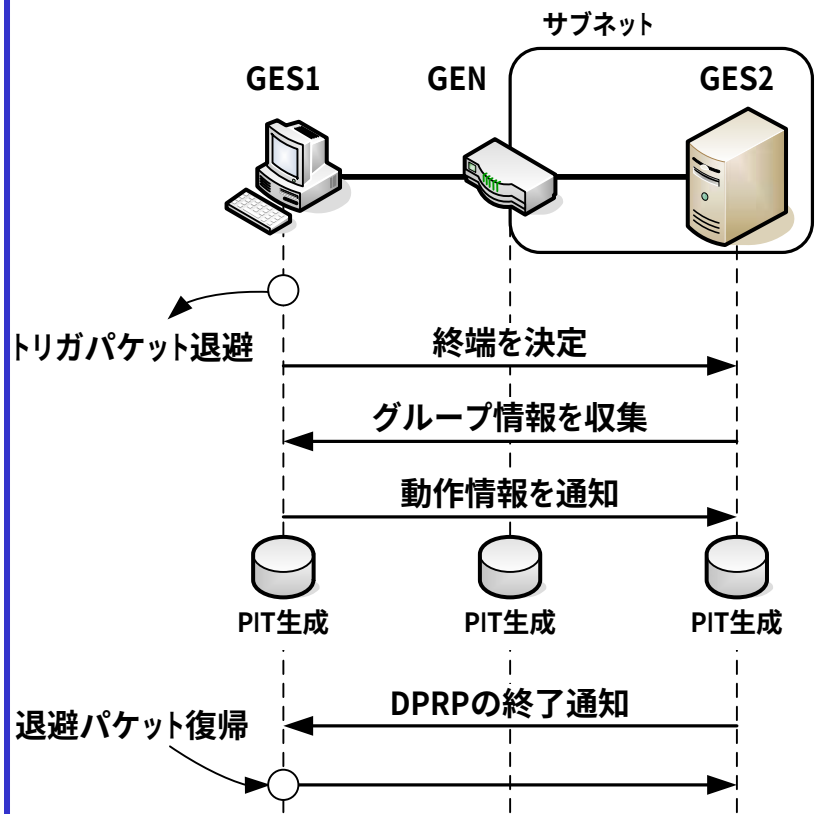
GE　：GSCIP対応装置
GES：ソフトウェア型
GEN：ルータ型
GMS：管理装置



- GMSがグループ鍵GKを各GEへ配送
- GKによって通信グループを構築
- 同一グループ間の通信はGKにより暗号化
- GKと通信グループを1：1に対応付け
- IPアドレスに依存しないグループ定義

DPRP (Dynamic Process Resolution Protocol)

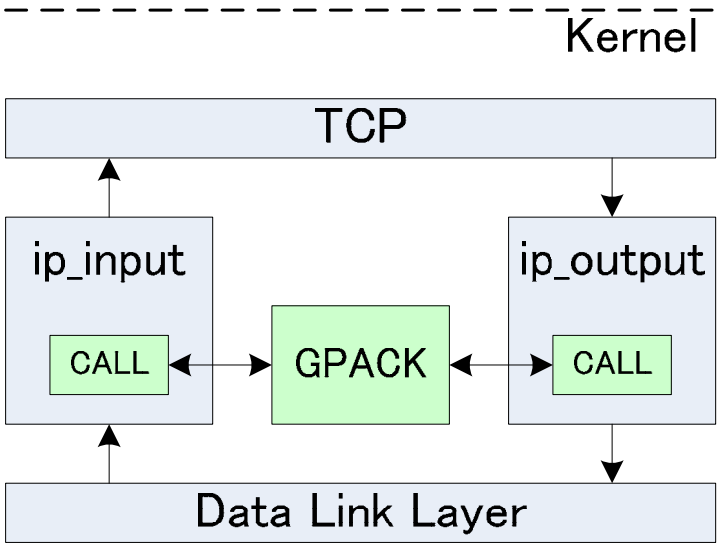
- 通信開始の際に各GEの情報を知るためにDPRPを行う



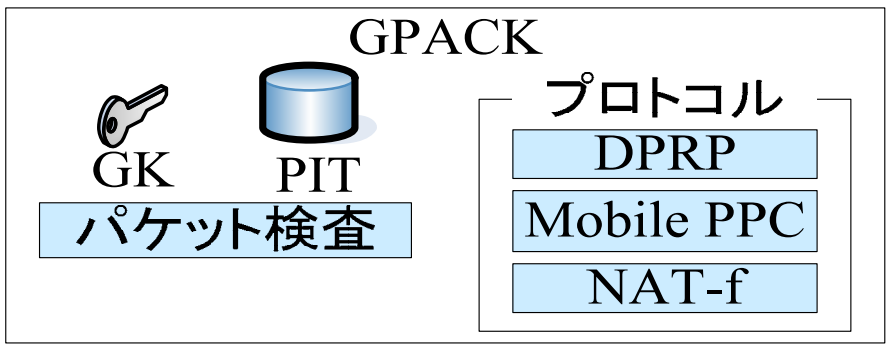
PIT (Process Information Table)
通信パケットに対する処理を定義する動作処理情報
(暗号化/復号，透過中継，破棄) 等を格納

- 終端GEを決定
- 経路上の各GEのグループ情報を収集し，動作処理情報を決定
  - グループ情報によって通信相手が同一グループであるか確認，認証
- 動作処理情報テーブルPIT を生成
- 以降の通信はPITに定義された動作処理情報に従って動作

GSCIPの現状



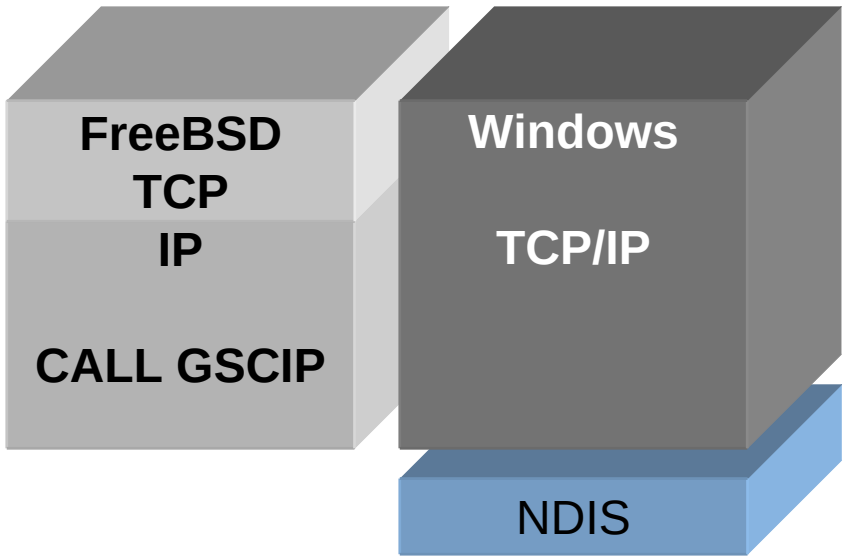
- FreeBSDではIP層にモジュール呼び出しを追加
- 動作と有効性を確認済み



GSCIPの評価や普及にはWindowsへの実装が不可欠

Windows

- WindowsはTCP/IPなどのOSがブラックボックス
  - FreeBSDのようにIP層を直接改造できない



外部に仕様が公開されたインタフェース

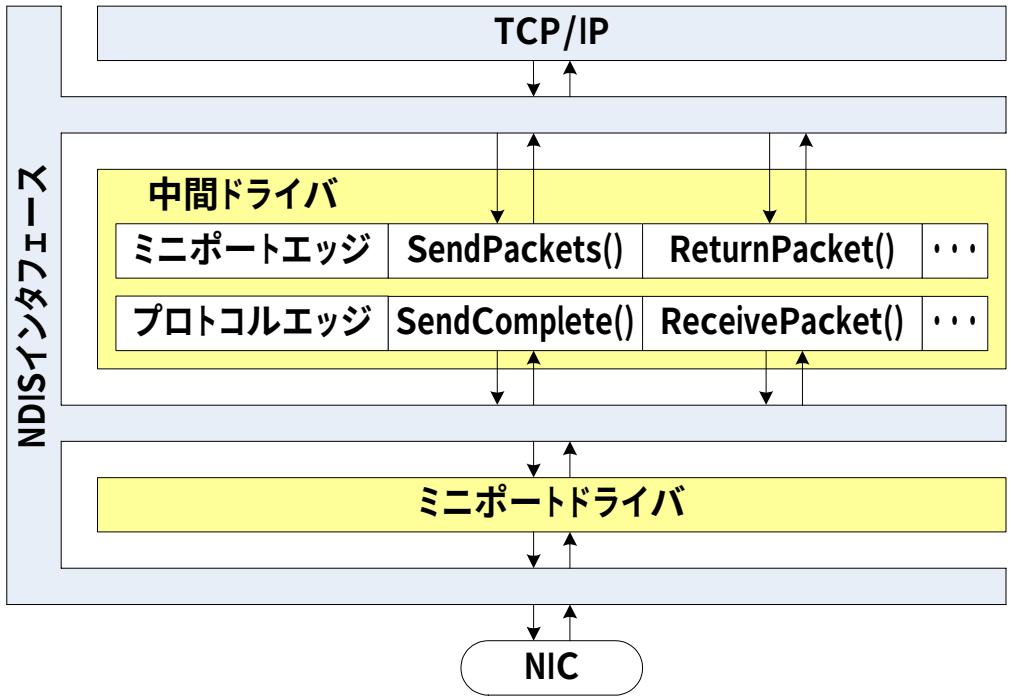
ネットワークの機能拡張ができるインタフェース

NDIS (Network Driver Interface Specification)

NDISの概要

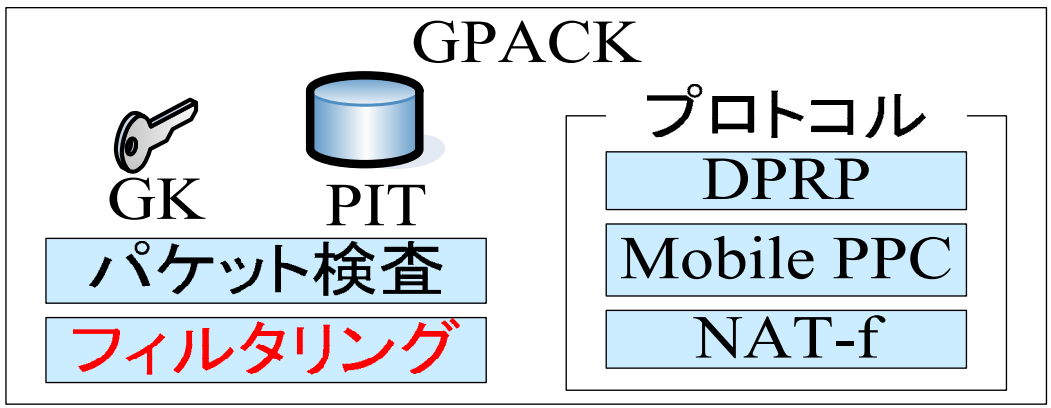
- NDISはネットワークに機能を追加できるインタフェースとそこで動作するドライバの動作手順を定める

- データリンク層の機能の一部
- NDISドライバは仕様として公開された機能を実行するモジュール群として作成し，NDISインタフェースに登録
- NDISインタフェースは各モジュールを必要に応じて呼び出す



GPACK (Gscip PACKage)

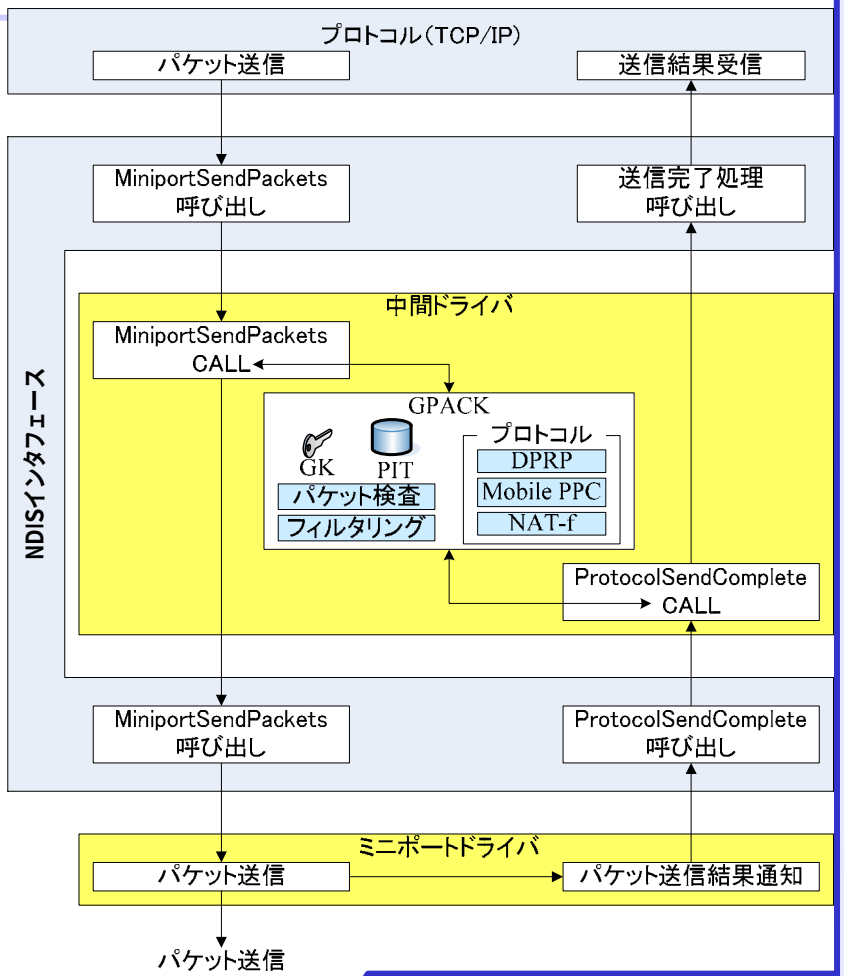
- GSCIPの機能を1つのモジュールに集約
- データリンク層で動作するためフィルタリングを行う



NDISの送信動作

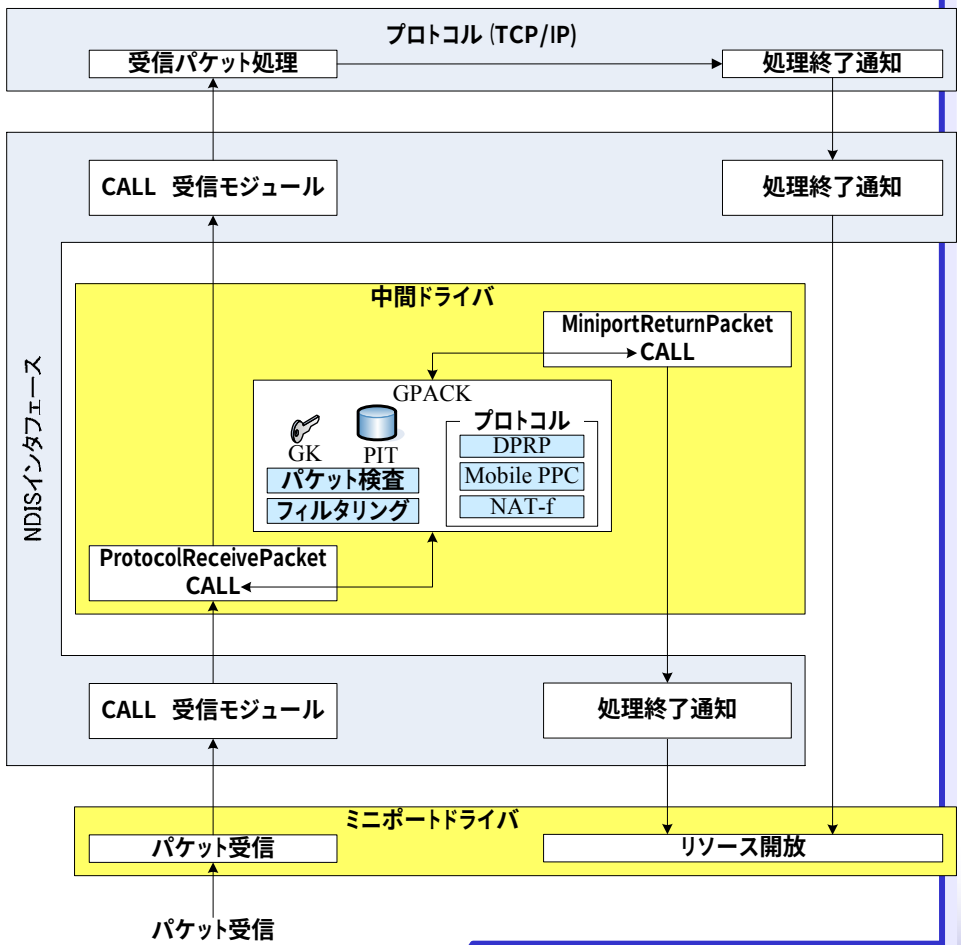
GSCIPは中間ドライバとして実装

- 送信動作
- MiniportSendPackets
  - 送信パケットを中継
  - GSCIPを呼び出し，送信時の処理を行う
- ProtocolSendComplete
  - パケット送信処理の結果を通知



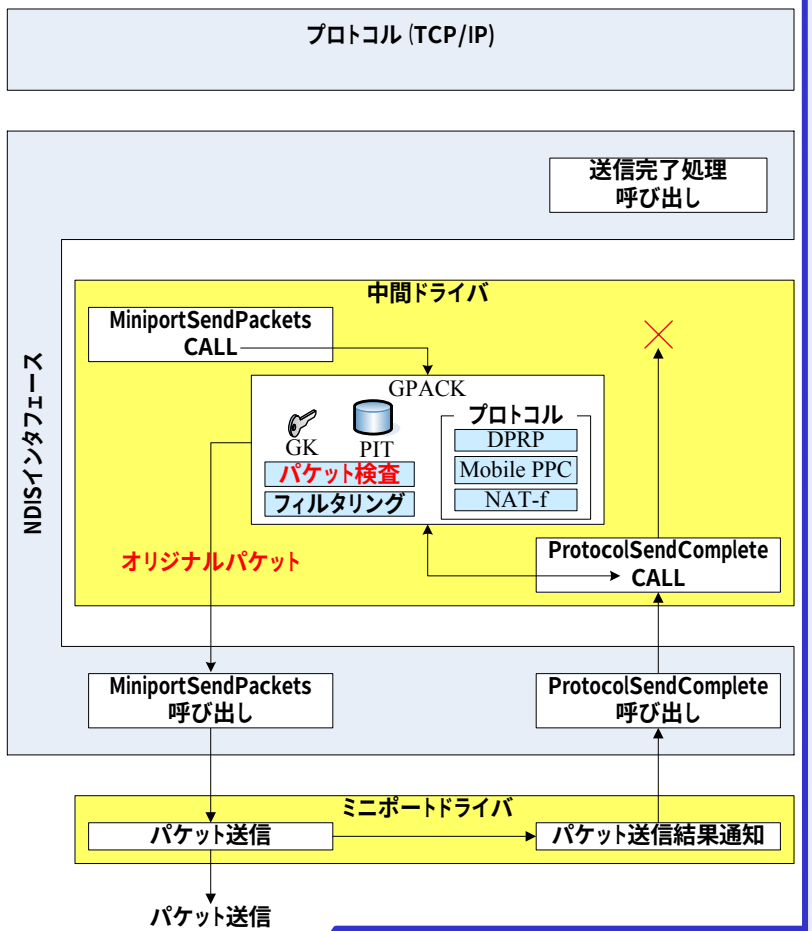
NDISの受信動作

- **受信動作**
- ProtocolReceivePacket
  - **パケットの受信を通知**
  - GSCIPを呼び出し，**受信時の処理を行う**
- MiniportReturnPacket
  - **パケットへの処理終了を通知**



送信処理完了通知
SendComplete

- GSCIPのプロトコルはオリジナルパケットを作成し，通信を行う
  - TCP/IPが関与しないパケットに関するSend Completeが行われるとクラッシュを引き起こす原因になる
- ProtocolSendCompleteにGSCIP独自パケットの判断処理を追加
- TCP/IPが関与しないパケットを通知しない



評価

100BASE-TXのEthernet

2台のPCを直接接続

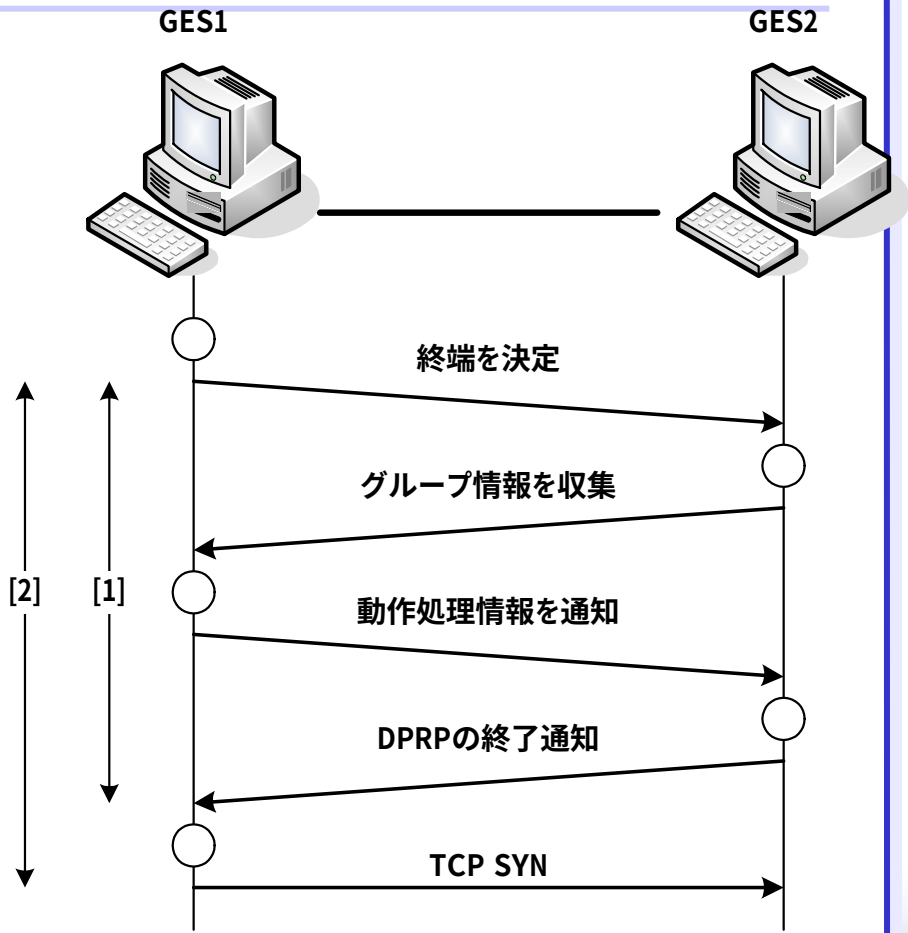
OS　Windows XP

CPU Pentium4　2.4GHz

メモリ　1280MB

評価

- オーバヘッド時間
  - [1] 通信に先立って行われるネゴシエーション時間
  - [2]トリガパケット送信までのオーバヘッド
- スループット
  - FTP接続で500MBのファイルをダウンロード



結果

- 通信に先立って発生するオーバヘッドは十分に小さい
- スループット低下率は約0.06%
- 通信に対する影響はほとんどみられない

オーバヘッド時間

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

スループット

単位：Mbps

GSCIP	実装時	未実装時
スループット	92.33	92.39

まとめ

- GSCIPをWindowsのインタフェースNDISを用いて実装する方法についての検討を行った
- GSCIPの基幹プロトコルDPRPを実装し，評価を行った
- 今後はGSCIPの全機能を実装し，性能評価を行う