アドレス空間透過性を実現するNAT-fの実装と評価

鈴木 秀和　　渡邊 晃

名城大学大学院理工学研究科

Implementation and Evaluation of NAT-f Actualizing Address Area Transparency

Hidekazu Suzuki and Akira Watanabe

Graduate School of Science and Technology, Meijo University

1　はじめに

グローバルアドレスの枯渇問題を解決するため，企業や家庭等のネットワークに対してプライベートアドレスを導入し，インターネットとの接点にアドレス変換 NAT（Network Address Translation）を実装した装置を設置する形態が一般となっている．しかし NAT のアドレス変換テーブル（以下 NAT テーブル）はプライベートアドレス空間からグローバルアドレス空間へのアクセス時にのみ生成されるため，逆方向のアクセスを開始することができない．この制約を緩和するために，NAT テーブルを予め静的に生成しておくポートフォワーディング機能があるが，ポート番号 1 つに対して 1 台の端末しか設定できない上，動的に変更できないため汎用性に欠ける．

一般に企業ネットワークでは堅固なファイアウォールが設置されており，外部から開始される通信を遮断していたため，NAT の制約が表面化することはなかった．しかしホームネットワークでは企業のような堅固なファイアウォールは必要とされず，外出先からホームネットワークに自由にアクセスしたいという要求が十分に考えられる．アドレス不足を根本的に解決するために IPv6 技術があるが，ホームネットワークへの導入はほとんど進んでおらず，導入が始まったとしても IPv4/v6 の混在環境が当分続くことが想定されるため，NAT の制約を除去することは有益である．上記課題を解決する技術を NAT 越えと呼ぶこともあるが，本稿ではこの機能をアドレス空間透過性と呼ぶ．

これまでにアドレス空間透過性を実現するための技術がいくつか検討されている．STUN（Simple Traversal of UDP Through NATs）[1] は端末間の通信に先立ち，ホームネットワークの端末とインターネット上の専用サーバが連携して NAT テーブルを生成する．しかし UDP 通信に限られることや，Symmetric NAT に対応できないなどの課題がある．また専用のサーバが必要となり，今後の P2P 通信の発展を考えるとこのような構造は好ましくない．UPnP（Universal Plug and Play）[2] はホームネットワークの端末と NAT ルータが連携して NAT テーブルを動的に生成する．しかしホームネットワークの端末が NAT テーブルの情報を通信相手に通知することにより外部からの通信開始を実現しているため，インターネット側から自由にアクセスすることはできない．またこれらの解決策はアプリケーションが STUN や UPnP に対応する必要があり，用途が限定されてしまう．そのためアプリケーションに依存することのないネットワークレベルでの解決策が望まれる．このような解決策として NATS（NAT with Sub-Address）[3] や IPv4+4[4] などがある．NATS はサブアドレスと呼ぶ独自の IP アドレス体系を導入し，ポート番号の代わりに IP in IP Tunneling を用いて NATS ルータを通過する．しかし NATS ルータは全送受信パケットに対してカプセル化／デカプセル化を行う必要があり，負荷が大きい．またカプセル化により通信性能が低下するなどの課題がある．IPv4+4 は IP ヘッダを拡張して，NAT ルータとホームネットワークの端末の IP アドレスを同時に記載し，NAT ルータで IP ヘッダの変換処理を行うことによりアドレス空間透過性の実現を試みている．しかし通信を行う全ての機器に機能を実装する必要があり，導入が難しい．またこれらはユーザが専門的な知識を持つ必要があり，容易に導入することが難しい．

我々はこれらの課題を解決するアドレス空間透過プロトコルとして NAT-f（NAT-free）[5] を提案している．本方式はグローバルアドレス空間の端末が通信開始に先立ち，NAT-f ルータとネゴシエーションすることにより，NAT テーブルを動的に生成する．NAT-f は第三の装置が必要なく，既存のシステムを利用して P2P でアドレス空間透過性を実現することができる．本稿では NAT-f を FreeBSD に実装し，動作検証および性能測定を行ったので，その結果について報告する．

以降，2 章で NAT-f の動作概要，3 章で実装について述べる．4 章で動作検証実験の結果と性能評価について述べ，5 章でまとめる．

2　NAT-f

本論文で提案する NAT-f はネットワークレベルの解決策であり，P2P でアドレス空間透過性を実現する．NAT-f はユーザが外出先から自宅のホームネットワーク内へ自由にアクセス可能にすることを目的としている．NAT-f ではインターネット上の端末（以下 GN）と NAT-f に対応した NAT ルータ（以下 NAT-f ルータ）が連携し，GN とプライベート IP アドレスをもつホームネットワークの端末（以下 PN）の通信に必要な NAT テーブルを動的かつ強制的に生成する．GN はパケットを送受信する際，NAT テーブルの情報に合致するように IP アドレスとポート番号を変換する．これにより，NAT-f ルータは GN からの通信を PN へ転送することができる．本方式によれば，TCP と UDP のどちらにも対応でき，カプセル化の必要がないのでオー

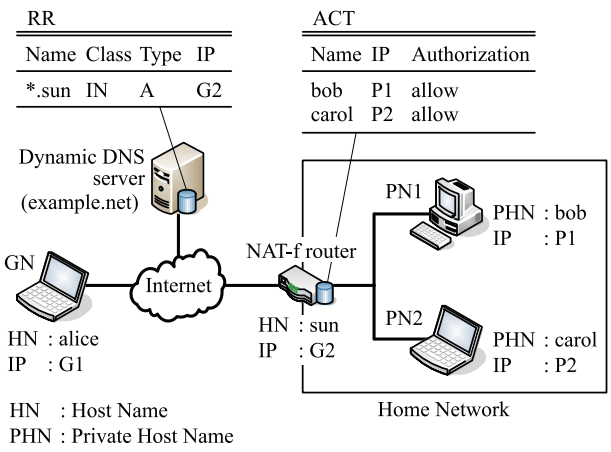


図 1: システム構成と初期設定情報

バヘッドが少ない．更には，Symmetric NAT にも対応できる等の利点がある．

2.1 動作概要

図 1 に NAT-f のシステム構成と初期設定情報を示す．GN と NAT-f ルータは NAT-f 機能を実装し，ダイナミック DNS サーバ（以下 DDNS サーバ）と PN の改造は不要である．DDNS サーバはワイルドカード機能を有効にしておく．事前準備として，ユーザは NAT-f ルータのホスト名とグローバル IP アドレスを DDNS サーバに登録する．NAT-f ルータのホスト名はインターネット側からホームネットワークを特定するために利用される．また PN を特定するための名前，プライベート IP アドレス，および外部からのアクセスの可否を NAT-f ルータのアクセス制御テーブル ACT（Access Control Table）に登録する．PN の名前はユーザが自由に決めることが可能で，インターネット上でユニークである必要はなく，ホームネットワーク内で PN を識別できればよい．一般のホスト名と区別するため，これをプライベートホスト名と呼ぶ．NAT-f による通信は以下の 3 つのフェーズから構成される．

(1) **DNS 名前解決処理**

図 2 に DNS の名前解決処理を示す．GN は PN1 と通信を開始する際，NAT-f ルータの FQDN“sun.example.net”の先頭に PN1 のプライベートホスト名 “bob”を付加して DDNS サーバに名前解決の依頼を行う．DDNS サーバは GN に対し，ワイルドカード機能により NAT-f ルータの IP アドレス “G2”を応答する．GN がこの応答を受信すると，GN のカーネルにおいて PN1 のプライベートホスト名と NAT-f ルータの IP アドレスを取得する．さらに NAT-f ルータの IP アドレスを仮想 IP アドレス “V1”に書き換え，これらの関係を名前関連テーブル NRT （Name Relation Table）へ保存する．仮想 IP アドレスとは通信相手となる PN を一意に特定するために割り当てる IP アドレスであり，GN の内部でのみ有効な値である．ここで生成した NRT は後に実行する NAT-f ネゴシエーションで通知すべき情報を特定するために用いられる．GN のアプリケーションへは仮想 IP アドレス “V1”を PN1 の IP アドレスとして報告する．

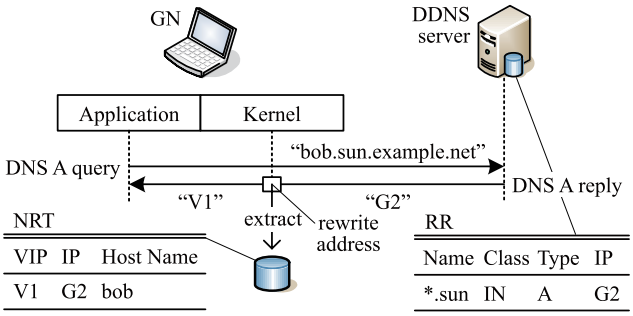


図 2: DNS 名前解決処理

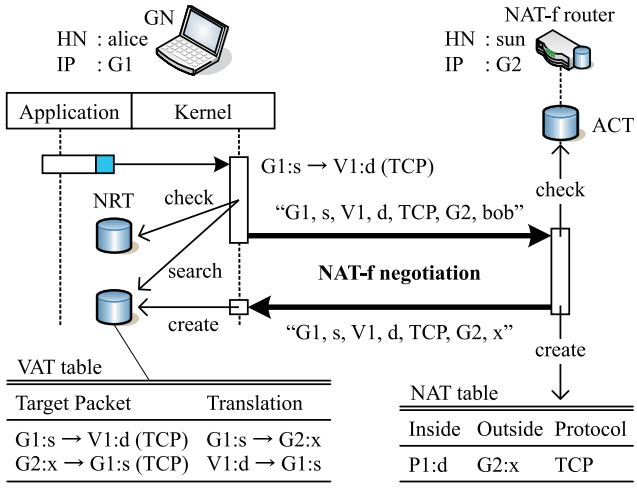


図 3: NAT-f ネゴシエーション処理

(2) **NAT-f ネゴシエーション処理**

図 3 に NAT-f ネゴシエーション処理を示す．GN は NAT-f ルータ宛の TCP/UDP パケットを送信する際，カーネルにおいて送信元／宛先 IP アドレスとポート番号，およびプロトコルタイプより仮想アドレス変換（VAT; Virtual Address Translation）テーブルを参照する．VAT テーブルとは PN の対応付けられた仮想 IP アドレス，ポート番号と NAT-f ルータの IP アドレス，ポート番号の相互変換関係が記されたテーブルで，NAT-f ネゴシエーション完了時に生成される．該当する情報が存在すれば，VAT テーブルに従って IP アドレスとポート番号を変換する．該当する情報が存在しない場合，宛先 IP アドレス “V1”より NRT を参照して仮想 IP アドレスに関連付けられた情報を取得する．そして TCP/UDP パケットを一時的に待避させてから NAT-f ネゴシエーションを実行する．GN はネゴシエーションのトリガーとなった TCP/UDP パケットの送信元／宛先 IP アドレス “G1, V1”とポート番号 “s, d”，プロトコルタイプ “TCP”，および NRT から取得した NAT-f ルータのグローバル IP アドレス “G2”と PN1 のプライベートホスト名 “bob”を NAT-f ルータに通知する．

NAT-f ルータがこの通知を受信すると，PN1 のプライベートホスト名 “bob”より ACT をチェックする．一致するプライベートホスト名が存在し，かつアクセスが許可されてい

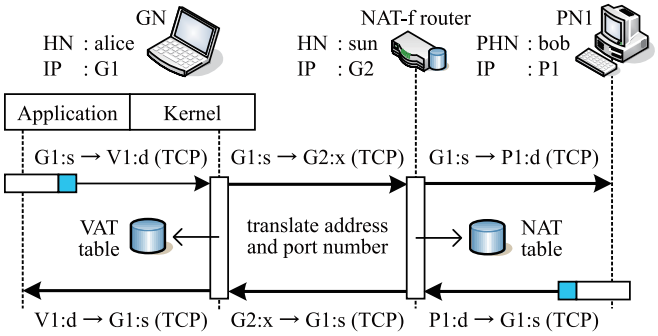


図 4: 仮想アドレス変換処理

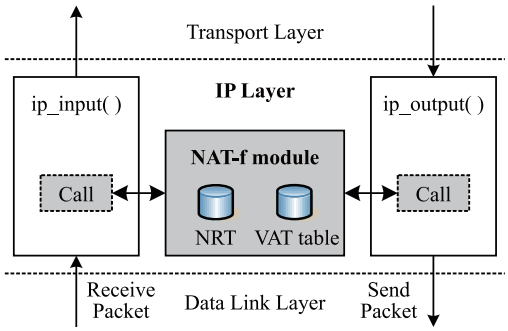


図 5: GN の実装概要

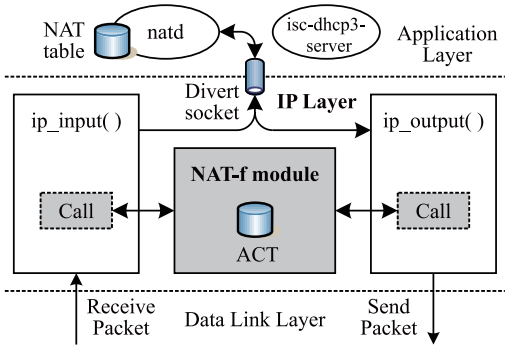


図 6: NAT-f ルータの実装概要

れば，受信した情報と該当する PN のプライベート IP アドレスから NAT テーブルを強制的に生成する．NAT-f ルータは先ほど GN から受信した送信元／宛先 IP アドレスとポート番号，プロトコルタイプ，NAT-f ルータのグローバル IP アドレス，および NAT テーブルに生成された変換後の送信元ポート番号 “x”を GN へ応答する．

GN が NAT-f ルータからの応答を受信すると，取得した情報から VAT テーブルを生成する．その後，一時的に待避していた TCP/UDP パケットを復帰させて NAT-f ネゴシエーションを完了する．

(3) **通信中の仮想アドレス変換処理**

図 4 に仮想アドレス変換処理を示す．復帰した TCP/UDP パケットは VAT テーブルに基づいて，宛先 IP アドレスとポート番号を “V1:d”から “G2:x”に変換して送信する．NAT-f ルータは既に NAT テーブルが生成されているため，通常の NAT 処理により宛先 IP アドレスとポート番号を “G2:x”から “P1:d”に変換して，該当する PN “bob”へパケットを転送する．GN が PN からの応答パケットを受信した際は，VAT テーブルに基づいて送信元 IP アドレスとポート番号を “G2:x”から元の “V1:d”に戻してからアプリケーションへと渡す．

このような処理を行うことで，GN から PN への通信開始を可能とし，アドレス空間透過性を実現することができる．

2.2 同時通信

図 1 において GN が “bob”と通信中に同一ホームネットワークの “carol”へ通信を開始する場合，GN は DNS 応答パケットに記載された NAT-f ルータの IP アドレスを仮想 IP アドレス “V2”に書き換えて，アプリケーションに報告する．アプリケーションは “bob”宛の通信パケットは “V1”，“carol”宛の通信パケットは “V2”として生成するため，カーネルでは “bob”に関するネゴシエーションと “carol”に関するネゴシエーションを区別して実行することがきる．もし DNS の応答に対してこのような仮想 IP アドレスの書換えを行わないと，アプリケーションは “bob”と “carol”への通信パケットを区別できず，ネゴシエーションで通知すべき情報を NRT から特定することができない．

FTP や IP 電話のようなアプリケーションはポート番号が変化するが，通信パケットの宛先は常に同一の仮想 IP アドレスとなるため，NRT からネゴシエーションに必要な情報を特定することができる．これによりポート番号が変化してもその都度，ネゴシエーションにより VAT テーブルと NAT テーブルが生成される．このように仮想 IP アドレスを導入することにより，汎用的なアプリケーションの利用が可能になる．

3 実装

NAT-f モジュールを FreeBSD の IP 層に実装した．図 5 に GN の実装概要を示す．実装箇所と既存カーネルの修正箇所は網掛け部分である．NAT-f モジュールは IP 層の入出力関数 `ip_input()`，`ip_output()` から呼び出され，NAT-f 対応の処理を行い，パケットを元の場所に差し戻す．NAT-f ネゴシエーションを実行する際，最初の TCP/UDP パケットを一時待避するが，パケットデータが格納されているメモリ領域は解放せず，カーネル内に留めておく．このパケットはネゴシエーションが完了した時点で，`ip_output()` へ渡すことにより，一時中断していた通信を即座に開始することができる．これによりネゴシエーションに伴う遅延を最小限に抑えることが可能になる．NRT，VAT テーブルはカーネルメモリ空間に作成し，不要になったら削除する．仮想 IP アドレスは NRT に登録された順に割り当てる．VAT テーブルはハッシュテーブルとして実装する．

図 6 に NAT-f ルータの実装概要を示す．NAT-f ルータには今回実装した NAT-f モジュールに加えて，NAT デーモン natd，DHCP サーバデーモン isc-dhcp3-server（いずれも FreeBSD で標準的に利用されるアプリケーション）を動作させる．NAT-f ルータが受信したパケットは divert ソケットを通じて，natd でアドレス変換処理が行われる．これらのデーモンは改造を必要と

せず，そのまま利用することができる．ACT はユーザが手動で設定する必要があるが，PN が Windows 端末の場合は DHCP 処理時に自動生成することも可能である．NAT テーブルを生成するときは，GN から送信されてきた NAT-f ネゴシエーションの情報と ACT の内容から疑似パケットと呼ぶパケットを生成する．疑似パケットとは PN から GN へパケットが送信されたように見せかけたものであり，図 4 における PN から NAT-f ルータへ送信されるパケットと同一の宛先，送信元情報とする．このパケットを`ip_input()` へ渡すと，natd は PN から GN へ送信されたパケットと判断して，NAT テーブルを生成する．NAT-f ルータは変換後の疑似パケットの内容をもとに，GN に対する NAT-f ネゴシエーション応答パケットを生成する．疑似パケットは実際には送信されず，ネゴシエーションの応答パケットを生成した後，破棄される．

4 動作検証と評価

4.1 動作検証

GN“alice”から PN1“bob”への FTP 接続を行った結果，ポート番号が変化してもファイルのダウンロードを行えることを確認した．また PN1“bob”と PN2“carol”に対して同時に HTTP 通信できることを確認した．

4.2 評価方法

図 1 の機器構成において，GN と PN が通信を行う場合の NAT-f の性能を測定した．性能測定に使用した各装置のスペックは CPU が Pentium4 3.0GHz，メモリが 512MB である．またネットワーク環境は 100BASE-TX の Ethernet であり，GN，NAT-f ルータ，DDNS サーバをスイッチングハブで接続した．

NAT-f ネゴシエーションのオーバヘッドを明らかにするために，NAT-f ネゴシエーションを開始してから実際の通信が開始されるまでの時間をネットワークアナライザ Ethereal により測定した．次に，TCP/UDP 通信パケットに対して行う仮想アドレス変換処理がスループットに与える影響を明らかにするために，Netperf によりスループットを測定した．比較のために NAT-f を実装しない環境についても測定を実施した．NAT-f 未実装環境では予め NAT ルータにポートフォワーディング機能を設定し，GN から PN へ通信を開始できる状態とした．測定結果はいずれも 10 回試行の平均値である．

4.3 性能評価結果

(1) **NAT-f ネゴシエーションのオーバヘッド**

表 1 に NAT-f ネゴシエーションのオーバヘッドを示す．GN が通知パケットを送信してから応答パケットを受信するまでの時間は 347.5 マイクロ秒，実際の通信が開始されるまでの時間は 368.5 マイクロ秒であり，ネゴシーションのオーバヘッドは十分に小さいことがわかった．これは NAT-f ネゴシエーションのトリガーとなった通信パケットをカーネル内で待避し，復帰処理を行った結果である．NAT-f ネゴシエーションは通信開始に先立つネゴシエーションであるため，実際の通信にはほとんど影響を与えないといえる．

(2) **GN-PN 間のスループット**

表 1: NAT-f ネゴシエーションのオーバヘッド

	オーバヘッド（μsec）
ネゴシエーション時間	347.5
通信開始までの時間	368.5

表 2: Netperf によるスループット測定値

メッセージサイズ（MB）	NAT-f 実装（Mbps）	NAT-f 未実装（Mbps）
64	93.21	93.22
128	93.21	93.20
256	93.21	93.20
512	93.20	93.22
1024	93.20	93.23

表 2 に Netperf によるスループット測定値を示す．NAT-f 実装時，未実装時のスループットはどのメッセージサイズにおいても 93Mbps 程度であり，両者の間には有意差が認められなかった．NAT-f は IP フォワード機能を利用した場合と同等のスループットが得られており，仮想アドレス変換処理によるオーバヘッドはほとんど無いことがわかる．

5 まとめ

アドレス空間透過性を P2P で実現する NAT-f は，ホームネットワーク内の複数の端末と同時に通信できることから，実用性は高いものと考えられる．NAT-f の実装を行い，性能評価を行った結果，NAT-f は実際の通信にほとんど影響を与えることなく，NAT-f ルータの NAT テーブルを動的に生成できることを確認した．スループットを測定した結果，仮想アドレス変換処理が通信性能に与える影響はほとんど無いことを確認した．

今後は本方式を自宅ネットワークへのアクセスだけに限定せず，異なるアドレス空間で汎用的に適用できるように改良を行う予定である．

参考文献

[1] J. Rosenberg and J. Weinberger, “STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs),” RFC3489 (2003).

[2] UPnP Forum. http://www.upnp.org/

[3] K. Kondo, “Capsulated Network Address Translation with Sub-Address(C-NATS),” Internet Draft, draft-kuniaki-capsulated-nats-05.txt (2003).

[4] Z. Turanyi and A. Valko, “IPv4+4,” ICNP2002, pp.290-301 (2002).

[5] 加藤尚樹，柳沢信成，鈴木秀和，宇佐見庄五，渡邊晃, “ インターネットから家庭ネットワークへの接続を可能とする NATF プロトコルの検討と実装 ”，情報学ワークショップ（WiNF）2005 論文集，pp.142-146 (2005).

Multimedia, Distributed, Cooperative and Mobile (DICOMO) Symposium
Jul. 5th-7th, 2006
Program No. 4D1

アドレス空間透過性を実現するNAT-fの実装と評価

名城大学大学院理工学研究科
鈴木 秀和　　渡邊 晃

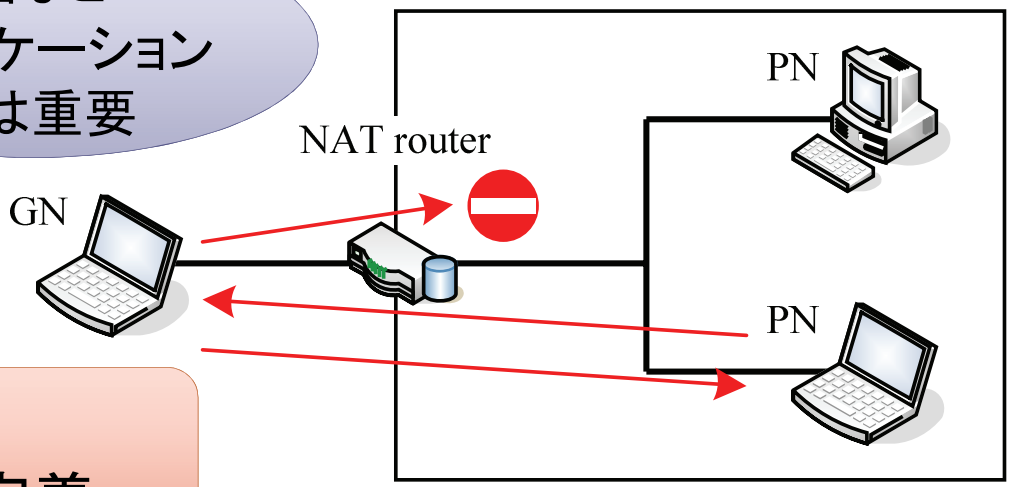
P2P通信の弊害

- ホームネットワークではNATが利用されている
  - ホームネットワーク＝プライベートアドレス空間
  - 片方向通信の制約
    - 外部への通信時にのみNATテーブルが生成されるため

IP電話などP2P
アプリケーション
では重要

NAT越えの研究

本発表では
アドレス空間透過性と定義



アドレス空間透過技術の分類

アプリエーションレベル

- GNとPNに専用のアプリケーション
- 専用のサーバを設置
- NATルータはそのまま

STUN
TRUN, ICE
Teredo
(UPnP)

ネットワークレベル

- OSに改良が必要
  ➔アプリケーションに依存しない
- 専用の機器・NATルータを設置

NATS
AVES
IPv4+4
IPNL

NAT越えの技術的仕組み

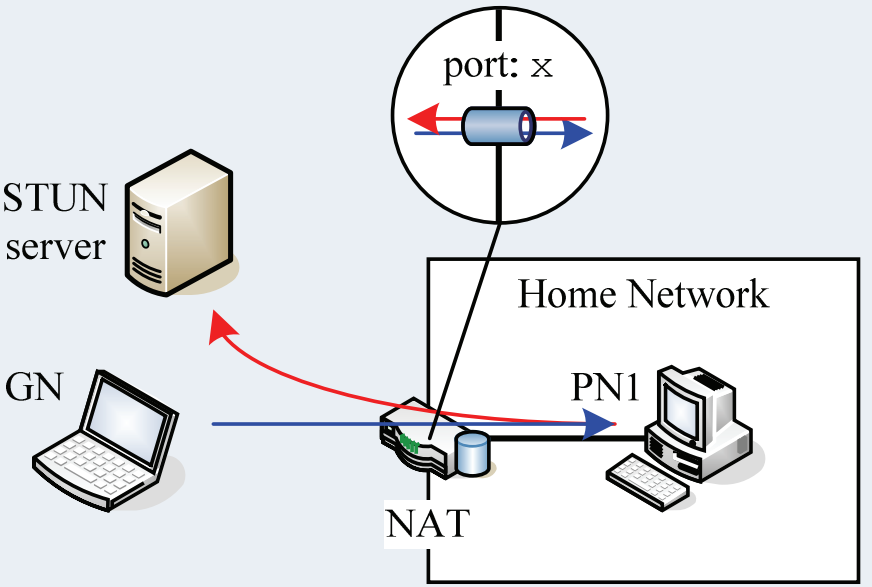
*STUN（Simple Traversal of UDP Through NATs）: RFC3489
Cisco　J. Rosenberg

アプリケーションレベル

- UDP Hole Punching
  - 内側からNATテーブルを生成
  - 外側からマップされたポートに向けて送信

NATの処理をそのまま実行

STUN*



プロトコルやNATの種類が限定される

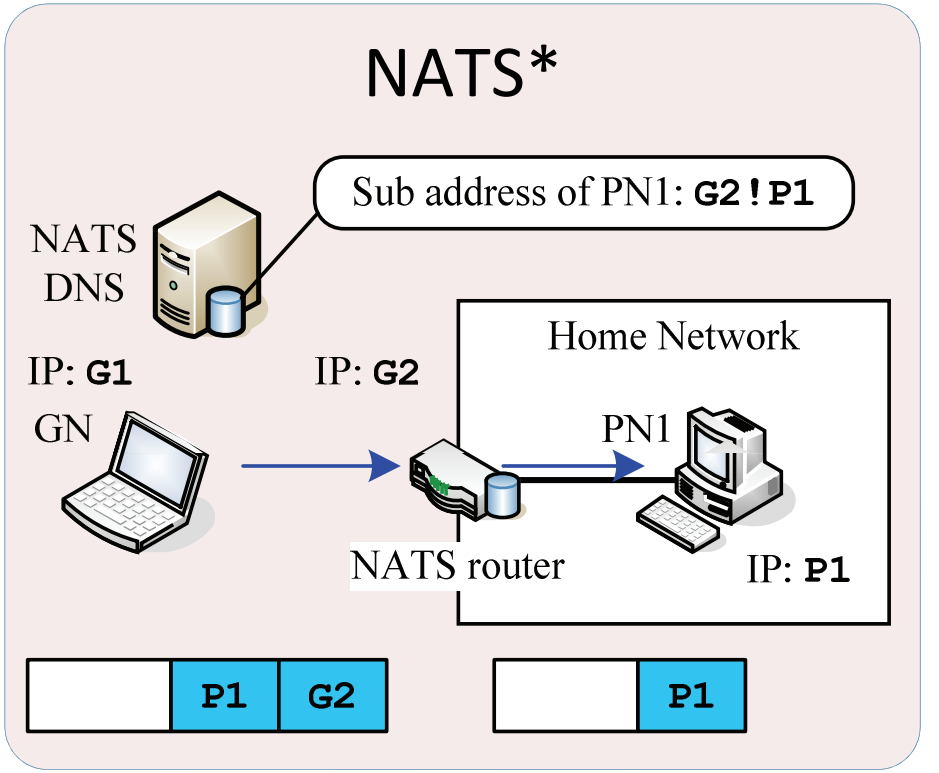
NAT越えの技術的仕組み

*NATS（NAT with Sub-Address）:
(株)インテック・ネットコア社
近藤邦昭氏の提案技術

ネットワークレベル

- IP in IP Tunneling
  - PN宛IPパケットをNATルータ宛でカプセル化
  - NATルータはデカプセル化

NATの処理を実行しない



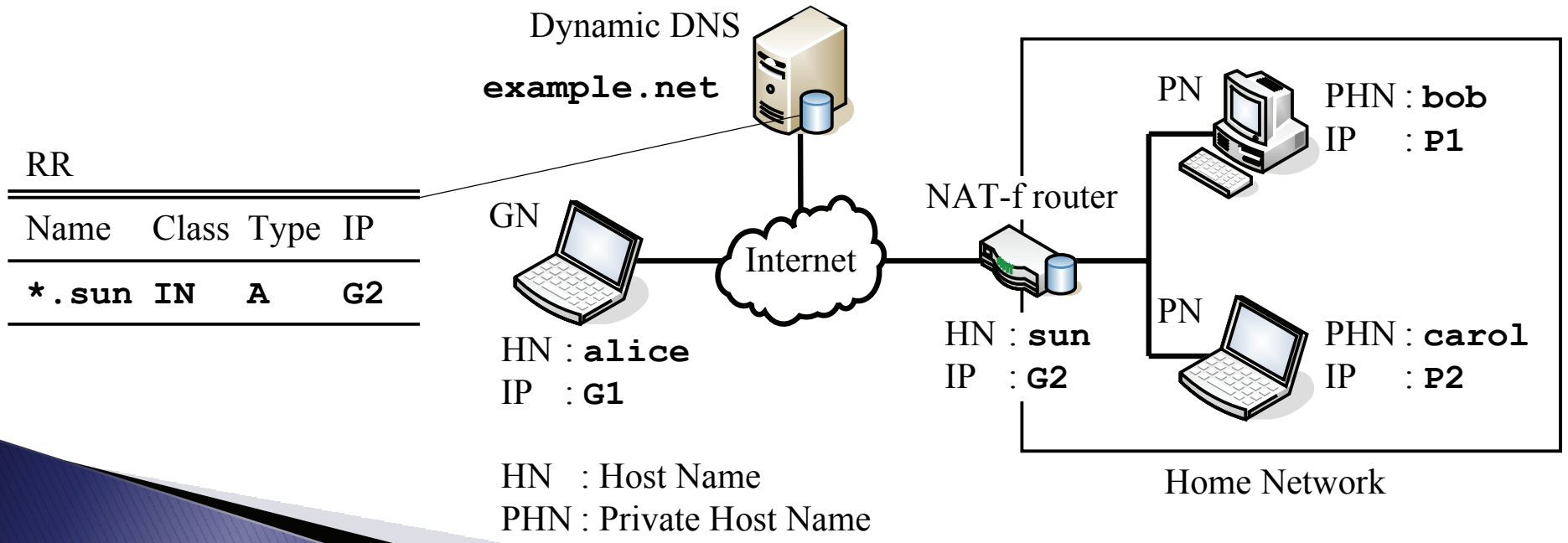
プロトコルやNATの種類は限定されない

NAT-f (NAT-free) protocol

- 目的
  - ユーザが外出先からホームネットワークへ自由にアクセス可能にする
- ネットワークレベルの解決策
  - 特殊なサーバを必要とせず，P2Pで実現
  - GNとNATルータにNAT-fプロトコルを実装
    ➔両者が連携してNATテーブルを外部から動的に生成
  - TCP/UDP，Symmetric NATに対応

システム構成・事前準備

- GN　：　NAT-f対応端末
- DDNS　：　ダイナミックDNSサーバ
  - ワイルドカード機能を有効に
  - NAT-fルータのホスト名とIPアドレスを登録

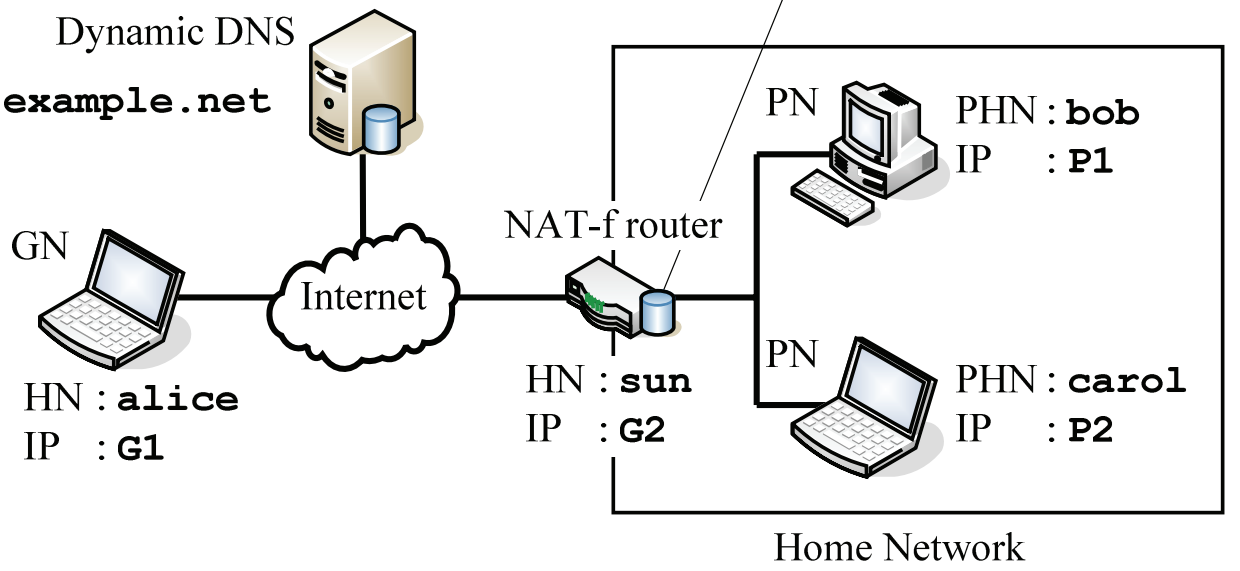


システム構成・事前準備

- NAT-fルータ　：　NAT-f対応NATルータ
  - アクセス制御テーブルACT（Access Control Table）
    - PNの名前とプライベートIPアドレス
      アクセス許可情報
- PN　：　一般端末

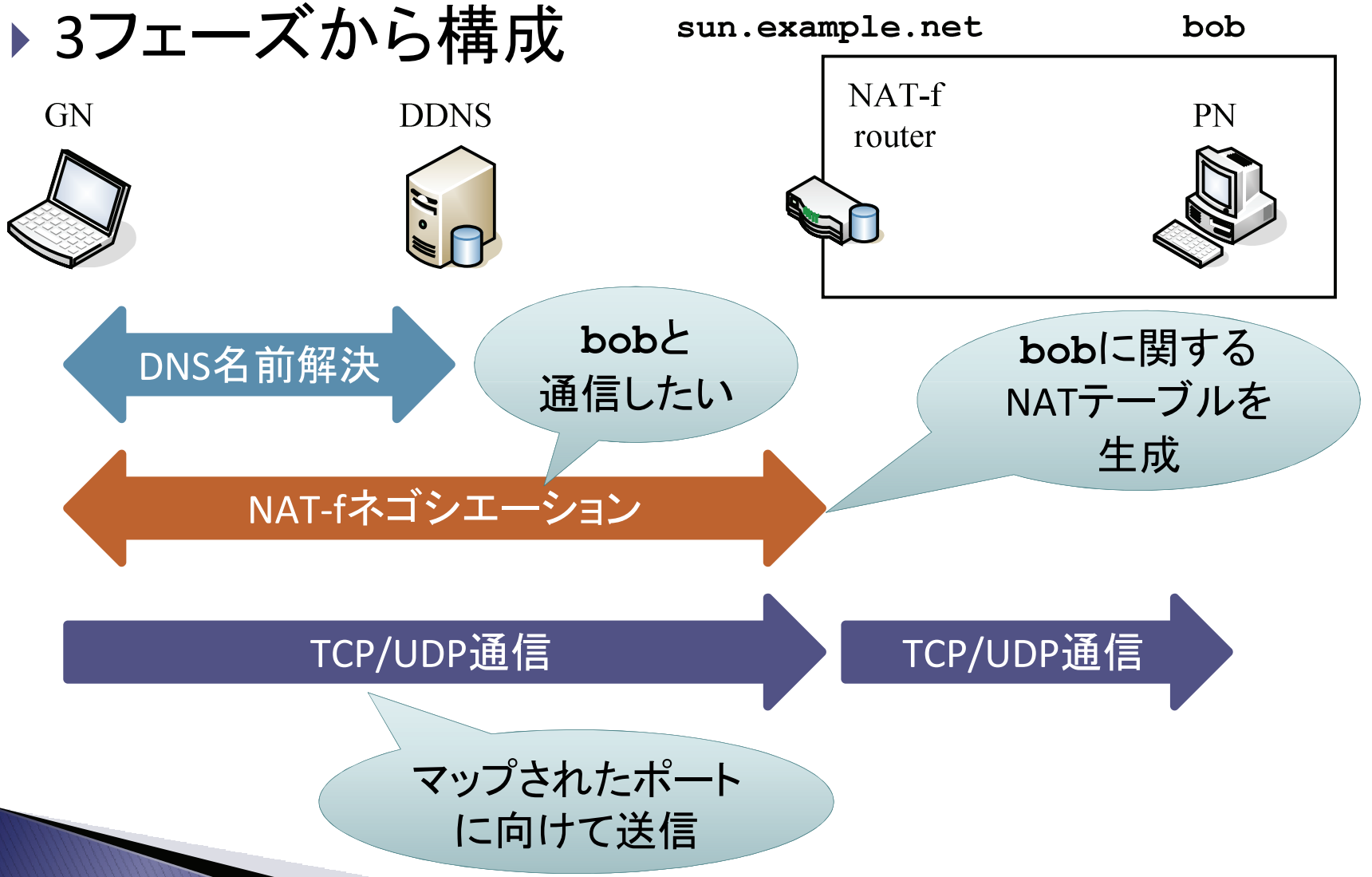
ACT

Name	IP	Authorization
bob	P1	allow
carol	P2	allow



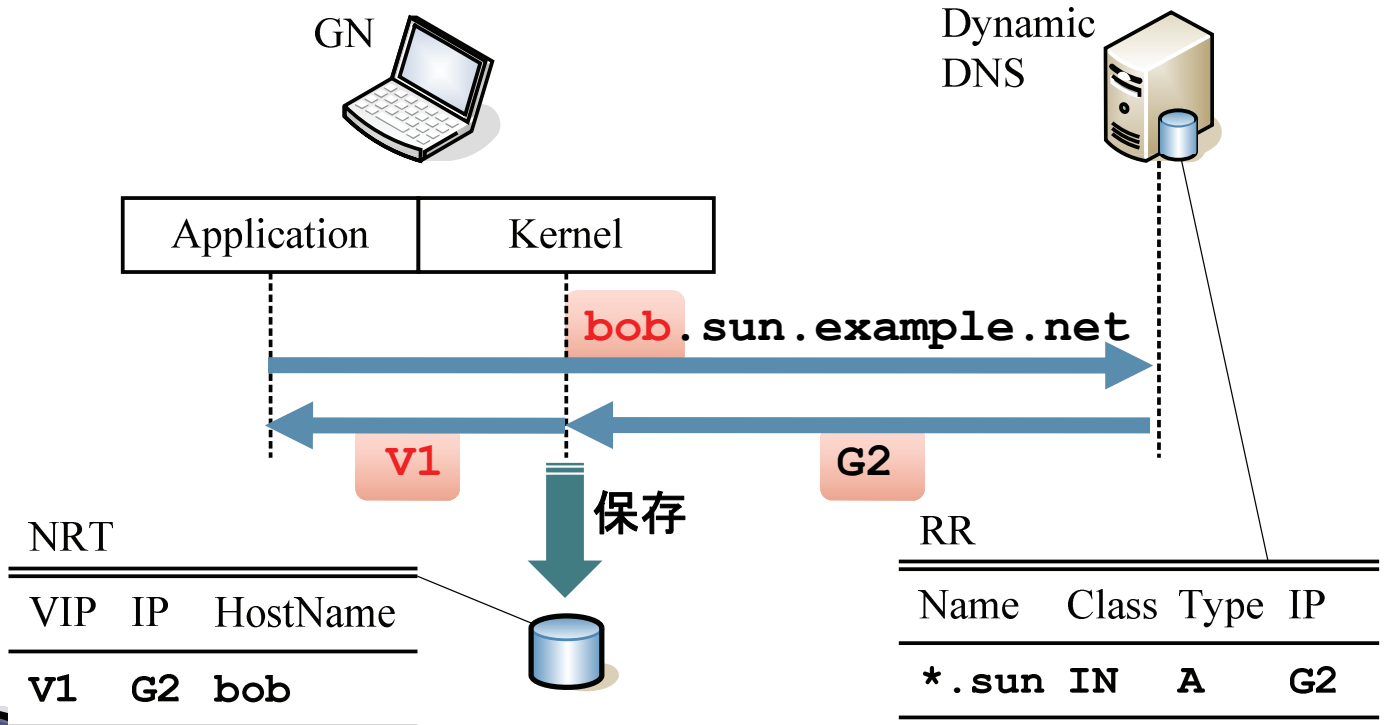
NAT-f動作概要

- 3フェーズから構成



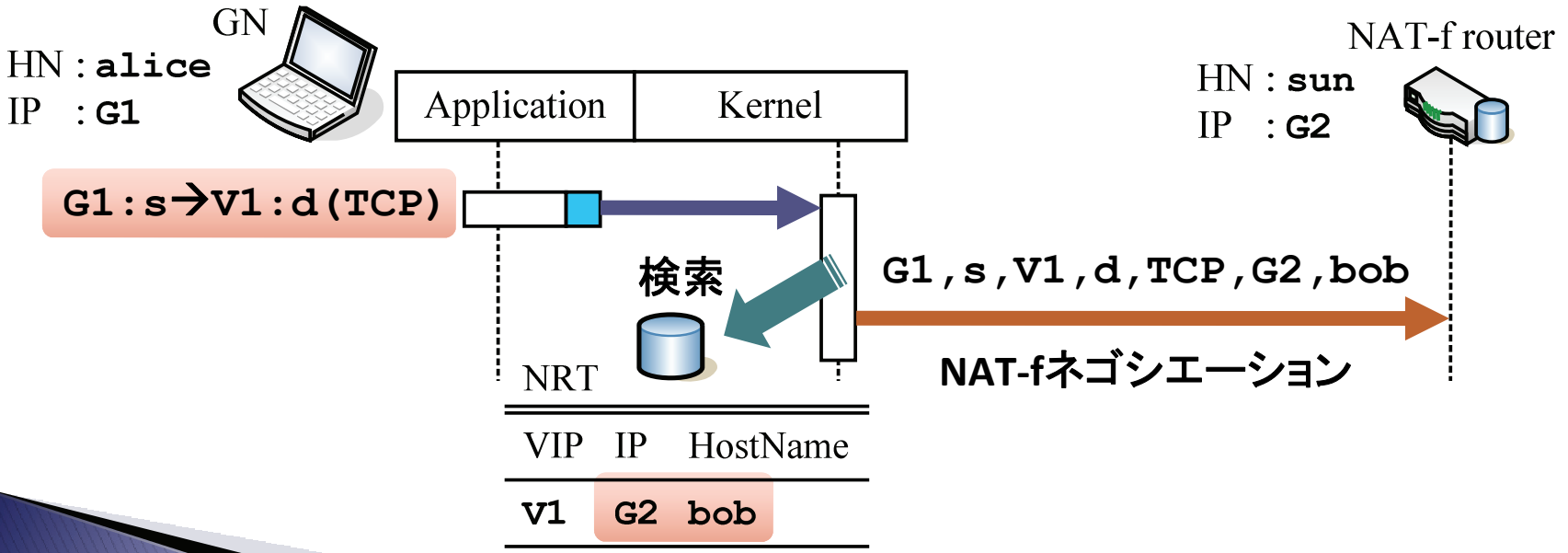
Ph.1　DNS名前解決処理

- GNは取得IPアドレスを仮想アドレスへ書換え
  - ➔名前関連テーブルNRT（Name Relation Table）に保存
    - 仮想アドレス，取得したIPアドレス，プライベートホスト名



Ph.2-1　NAT-fネゴシエーション処理

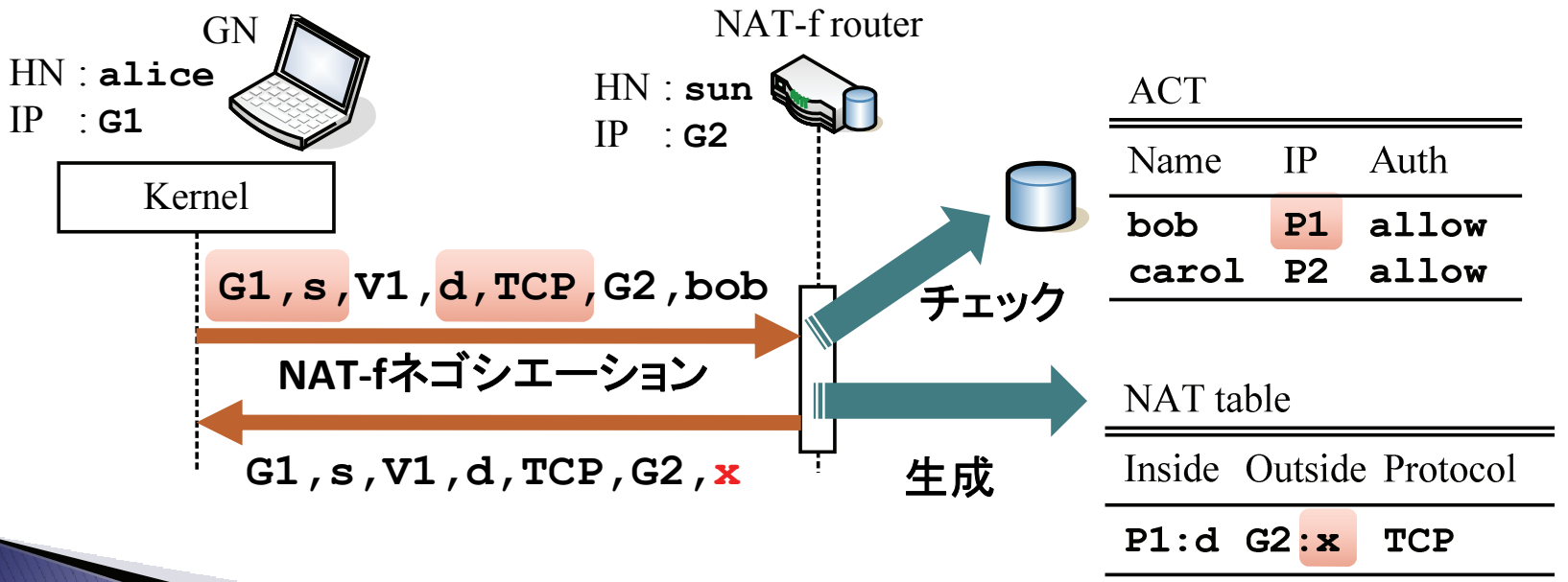
- GNは最初の送信パケットをトリガーにして
  - NAT-fルータに情報を通知
    - 送信パケットのIPアドレス，ポート番号，プロトコルタイプ
    - 仮想アドレスに対応するプライベートホスト名



Ph.2-2　NAT-fネゴシエーション処理

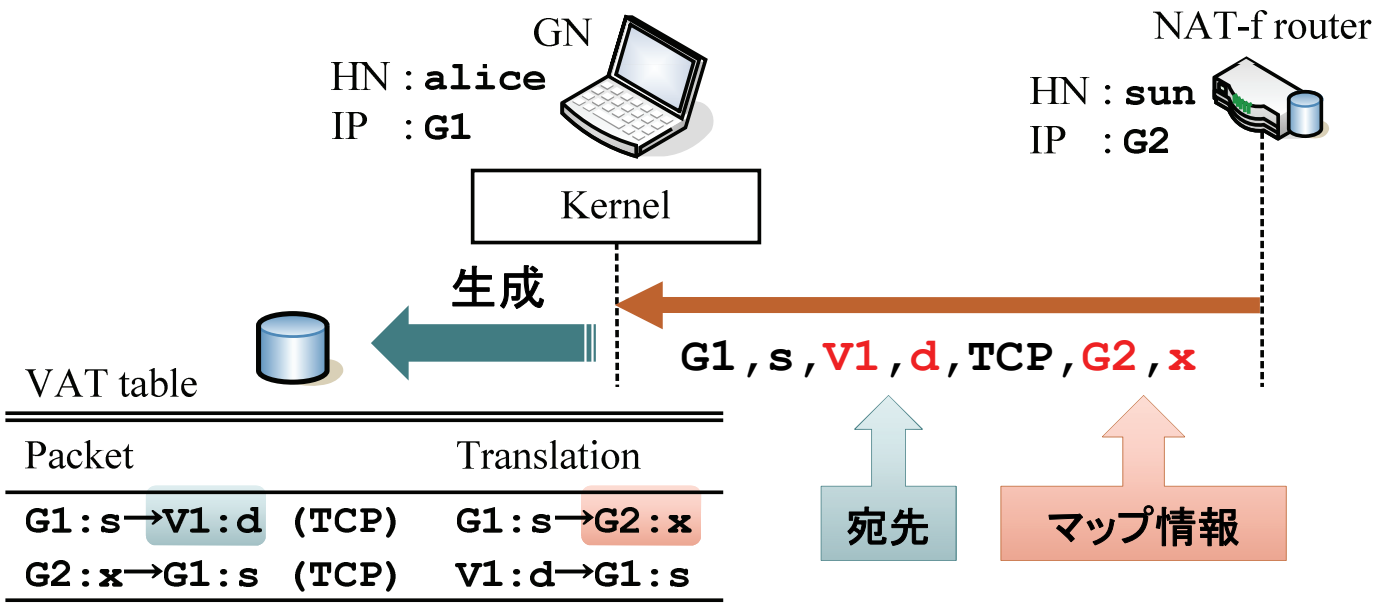
- NAT-fルータはGNからの通知に対して
  - 通知された名前よりACTをチェック

  ➔アクセスが許可されていたらNATテーブルを生成
  - 通知された情報とマップされたポート番号を応答



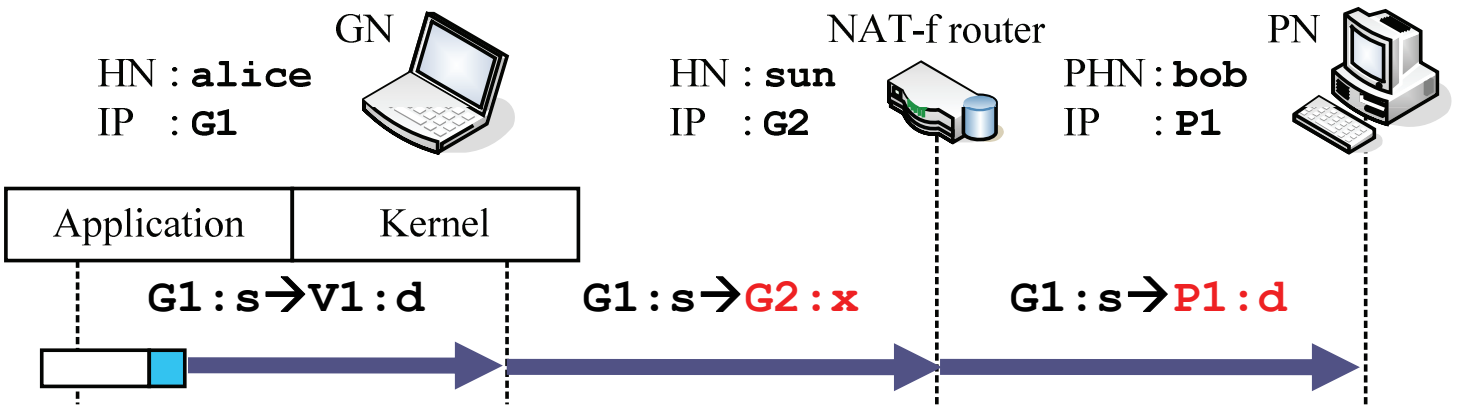
Ph.2-3　NAT-fネゴシエーション処理

- GNはNAT-fルータからの応答に対して
  - 仮想アドレス変換テーブルを生成
    = VAT（Virtual Address Translation）
  - TCP/UDP通信を再開



Ph.3　仮想アドレス変換処理

- GNは送信パケットに対してVAT処理
  - NAT-fルータにマップされたIPアドレス・ポートに変換
- NAT-fルータは通常のNAT処理によりPNへ転送



Packet	Translation
G1:s→V1:d (TCP)	G1:s→G2:x
G2:x→G1:s (TCP)	V1:d→G1:s

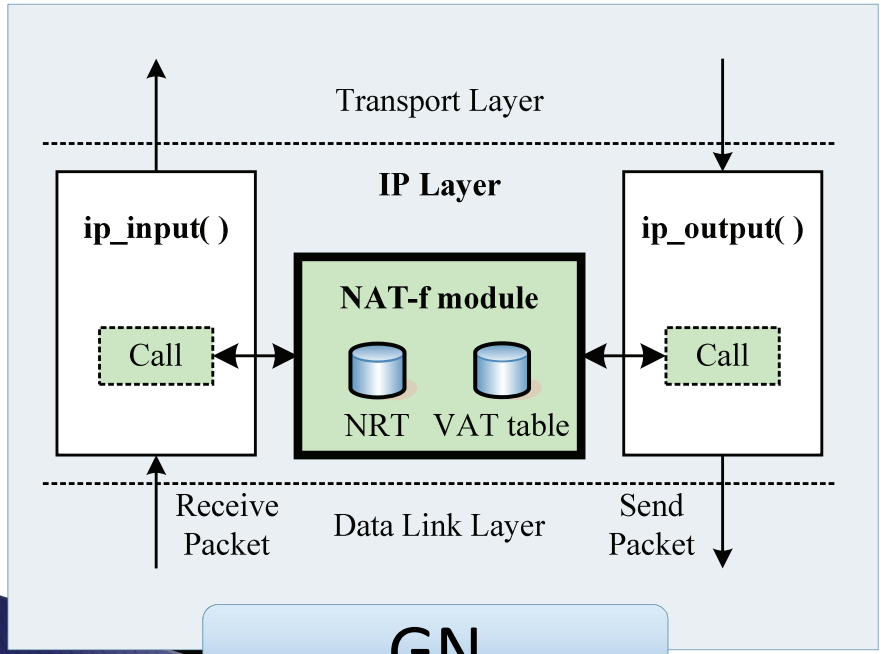
Inside	Outside	Protocol
P1:d	G2:x	TCP

NAT-fの実装

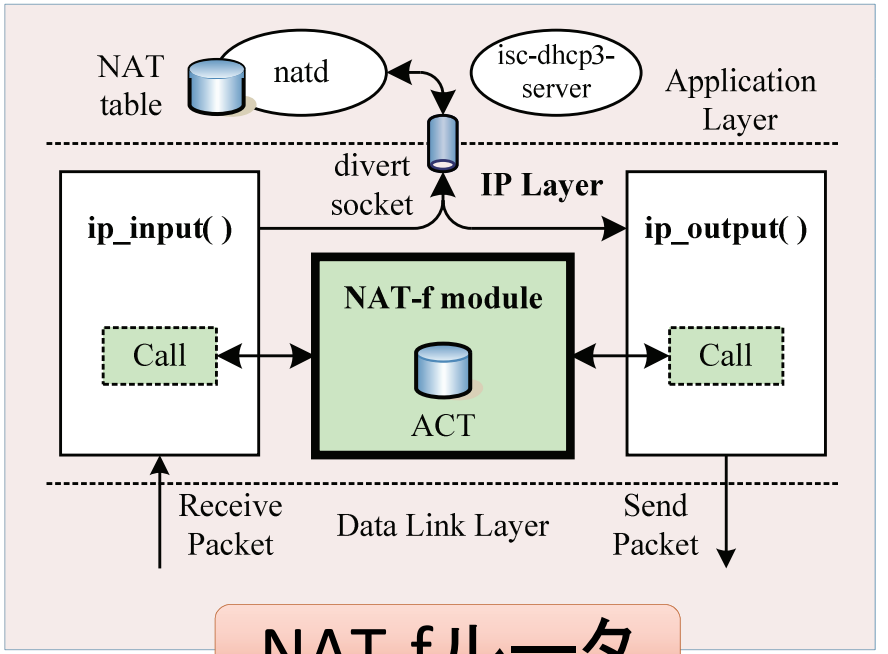
- FreeBSDのIP層に実装
  - **ip_input/ip_output**からモジュールをcall

➔既存のIP層の処理に影響を与えずに機能追加可能

  - NATプログラムはそのまま利用



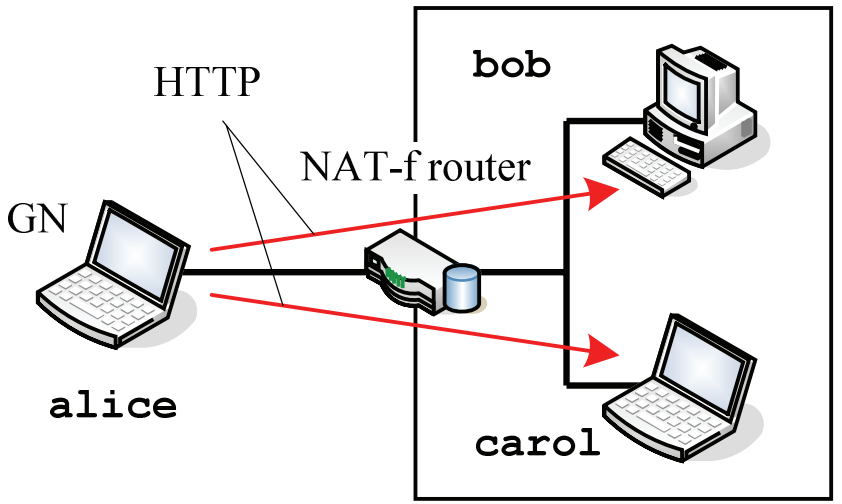
GN

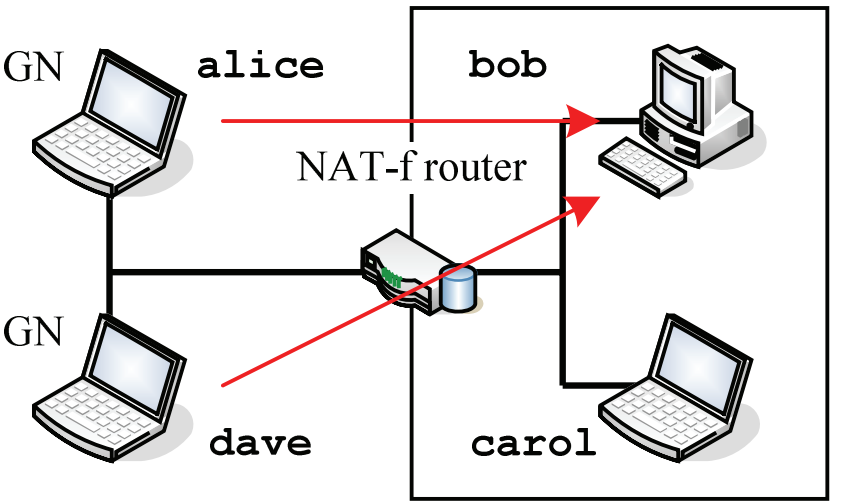


NAT-fルータ

動作検証

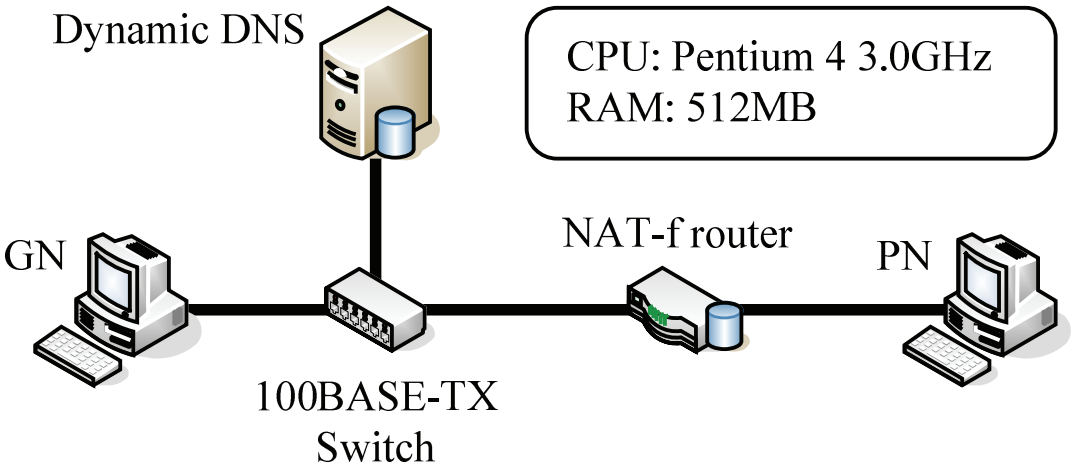
- 複数のPNへ同一ポート(HTTP)の通信が可能
- 複数のGNからPNへの同時通信も可能





性能評価

- 通信開始時のオーバヘッド
  - Etherealを使用
- スループットの測定
  - GNのVAT処理による影響（ポートフォワードと比較）
  - Netperfを使用



測定結果

- 通信開始時のオーバヘッド （μsec）

NAT-fネゴシエーション時間	347.5

- スループット （Mbps）

NAT-f実装時 （VAT処理あり）	93.21
ポートフォワード（VAT処理なし）	93.23

NAT-fを導入しても実際の通信に影響を与えない

まとめ

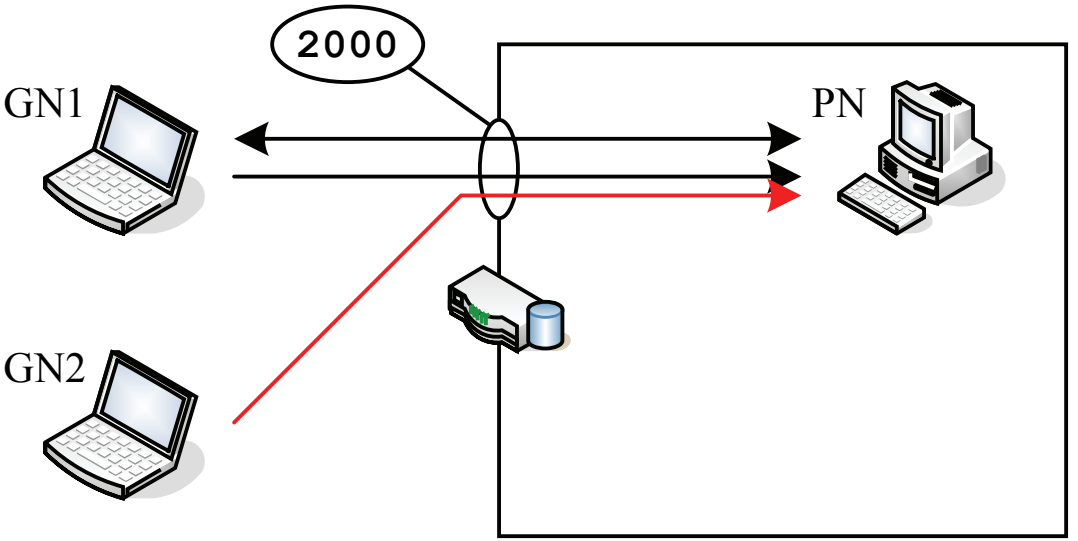
- アドレス空間透過性を実現するNAT-f
  - アプリケーションに依存せず，P2PでNAT越えを実現
  - NATの外部からテーブルを動的に生成
  - 通信性能に与える影響はほとんどない

- 今後の展開
  - 移動透過通信への応用
    - PNがグローバルアドレス空間，プライベートアドレス空間を跨って移動

付録

Full Cone NAT

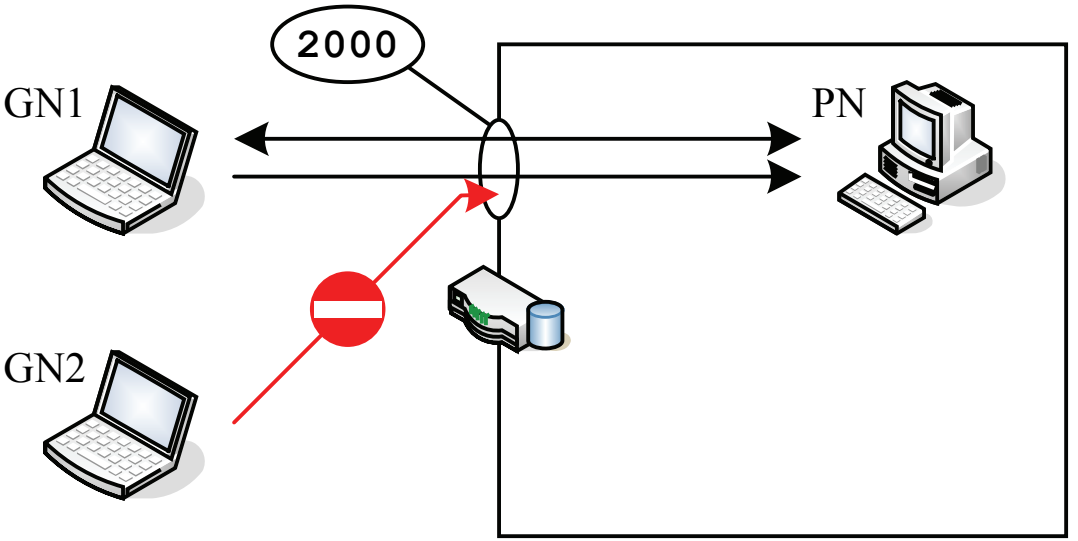
- 任意の外部端末からマップされた内部端末へ送信可能



外部	NATルータ	内部
:	NAT:2000	PN:1000

Restricted Cone NAT

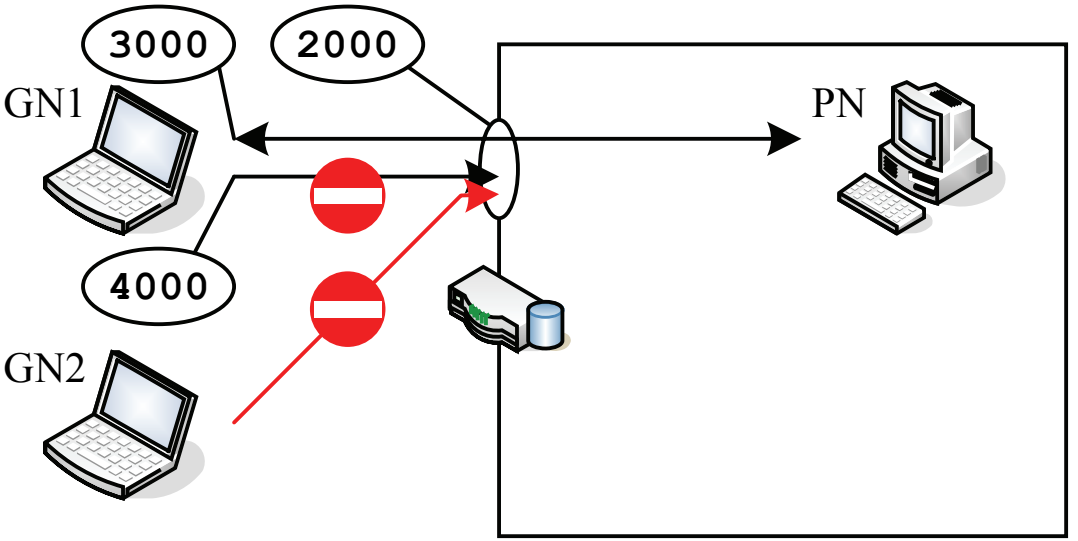
- 送信したことがあるIPアドレスからのパケットのみ内部端末へ送信可能



外部	NATルータ	内部
GN1:*	NAT:2000	PN:1000

Port Restricted Cone NAT

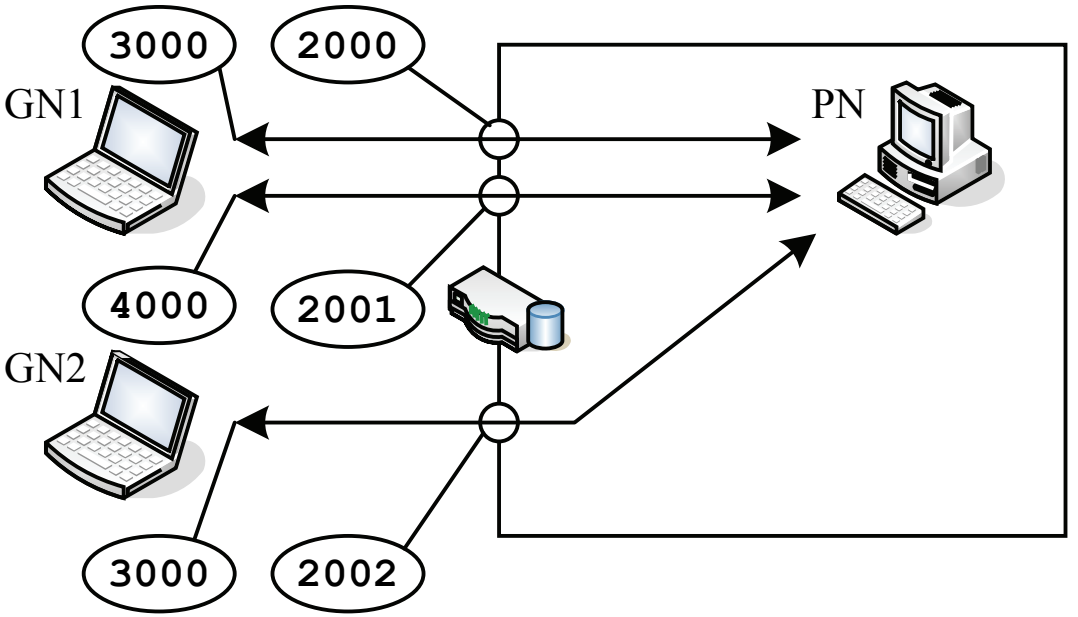
- 送信したことがあるIPアドレス&ポート番号からのパケットのみ内部端末へ送信可能



外部	NATルータ	内部
GN1:3000	NAT:2000	PN:1000

Symmetric NAT

- 宛先ごとに異なるポート番号がマップされる



外部	NATルータ	内部
GN1:3000	NAT:2000	PN:1000
GN1:4000	NAT:2001	PN:1000
GN2:3000	NAT:2002	PN:1000

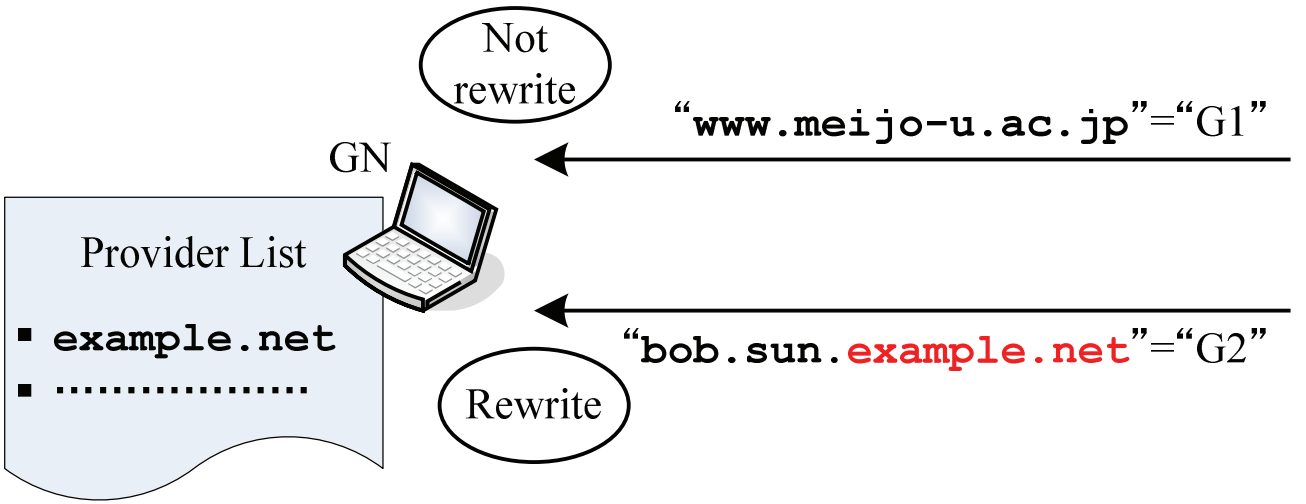
NAT越え技術

Application Level	設置・変更箇所	特徴
STUN	STUNサーバ	Full Corn NATにのみ対応
TURN	TURNサーバ	TURNサーバを中継. TCP，Symmetric NAT対応.
ICE	STUN/TURNサーバ	STUN，TURNの連携．SIPベースで複雑
Teredo	Teredoサーバ	IPv6 over UDP(IPv4)トンネリング.
UPnP	UPnP対応NATルータ	PNが動的にNATテーブルを生成

Network Level	設置・変更箇所	特徴
NATS	DNS，NATルータ，GN	サブアドレス体系を導入
AVES	DNS，NATルータ， waypointサーバ	PNへの通信は全てwaypoint中継. 三角形路.
IPv4+4	GN，PNを含めた通信 経路上の全機器	IPヘッダを拡張し，2つのアドレスを連結. IPv4アドレスを入れ替えてルーティング
IPNL	GN，PNを含めた通信 経路上の全機器	L3,L4の間に新しい層を追加. 独自のアドレスによりルーティング.

DNS応答の書換え判断

- DNS応答に記載されているドメイン名で判断
  - Exp)　NAT-fサービスプロバイダリストを保持する

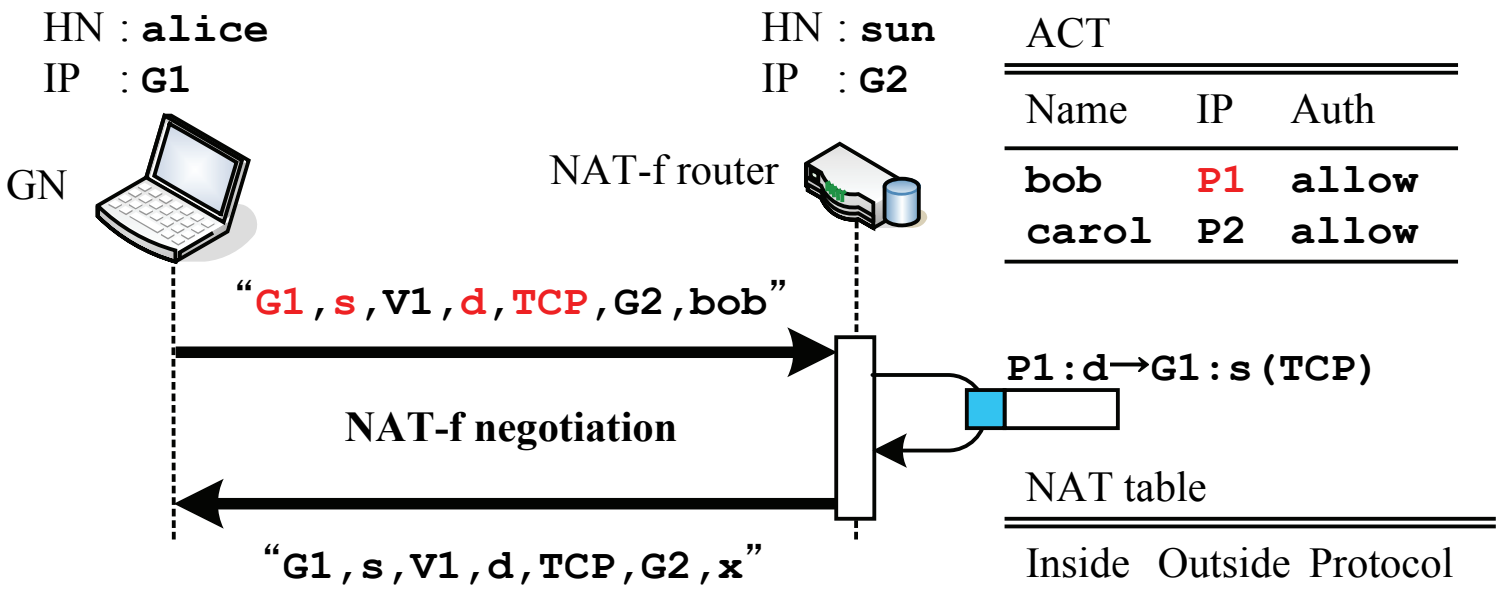


同時通信

- 仮想アドレスへ書き換えているため，送信パケットがNAT-fルータ配下の誰に送信したいのか特定できる
  ➔「○○のNATテーブルを生成」と通知できる
- NATテーブルの作り方はPNからGNの通信時と同じ方法
  ➔PNが複数のGNと，複数のPNが同一GNと通信できることと同じ原理

NATテーブル生成手法

- 通知情報とACT情報から疑似パケットを生成
  - PNからGNへの送信パケットに見せかけたもの
  - NAT-fルータの内側インタフェースで受信した際の処理を実行



ACT

Name	IP	Auth
bob	P1	allow
carol	P2	allow

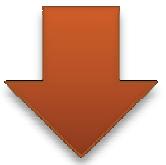
NAT table

Inside	Outside	Protocol
P1:d	G2:x	TCP

NAT-fだけで対応できないケース

- IPアドレスやポートを制御するアプリケーション
  - FTP
  - SIP

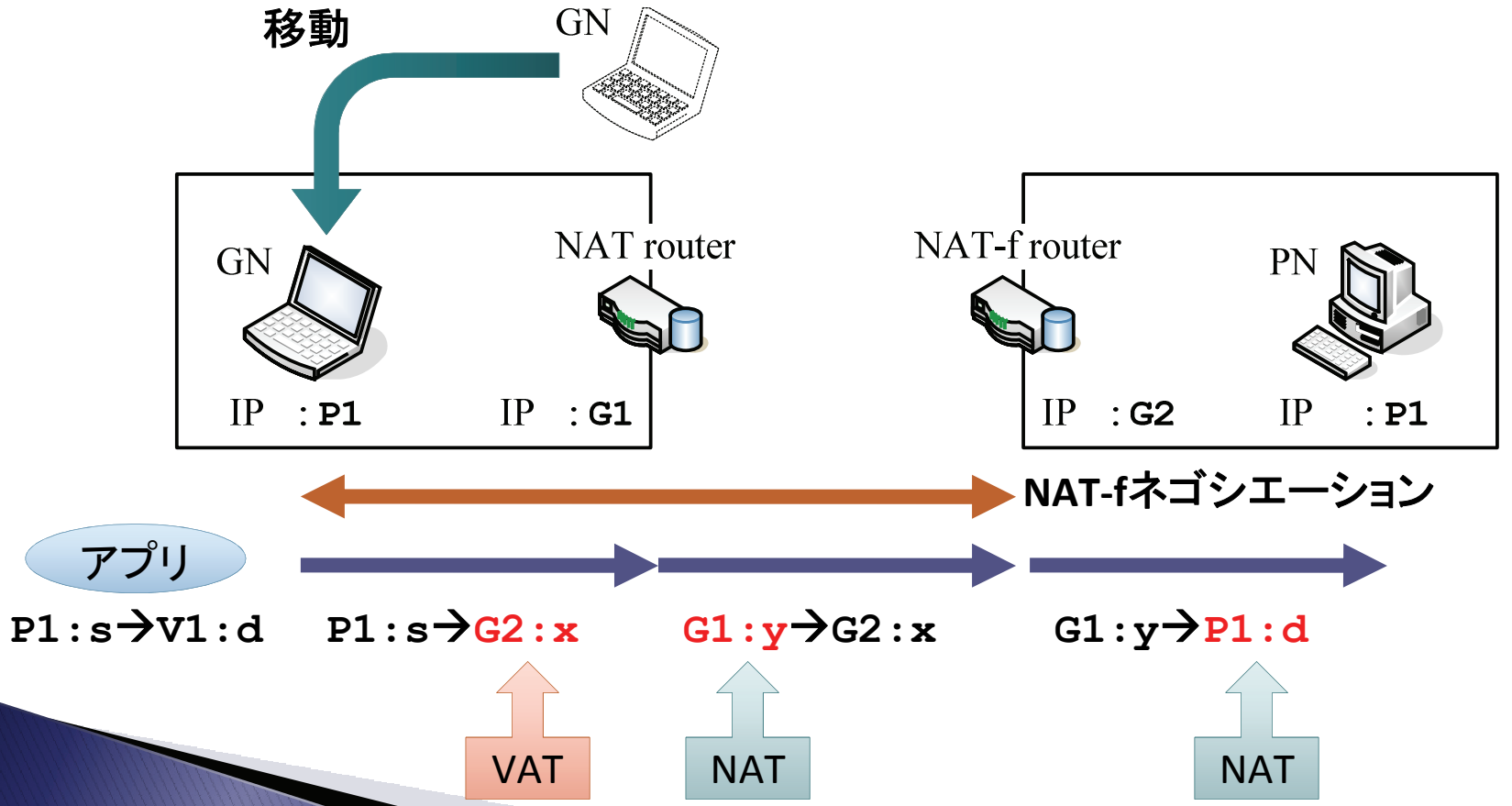
➔IPペイロード内にIPアドレス・ポートの情報が記載されており，NAT通過時に変換されない



ALG（Application Layer Gateway）を
NAT-fルータに実装して解決

GNがNAT配下にいる場合

- NAT-fルータのNATの種類がFull Cone NATとRestricted Cone NATなら対応



NAT-fの応用例

- 異なるプライベートアドレス空間同士の通信
  - GNの機能をNAT-fルータに搭載

➔NAT-fルータ間でネゴシエーション