第20部

大規模な仮設ネットワークテストベッドの設計・構築とその運用

第 1 章　合宿ネットワークの概要

本年度の WIDE 合宿は、9 月 6 日–9 日に静岡県浜松市舘山寺温泉で、3 月 14 日–17 日に山梨県石和温泉で行なわれた。

9 月合宿では、機器トラブルや準備不足が原因で、動作が不安定な部分が多かった。一番の反省点は、各実験が相互依存する構成になっていたため、ある実験で問題が起こると他の実験にも大きく影響がでて、思うような実験結果が得られなかったことである。

3 月合宿では、その反省に基づき、各実験の相互依存を減らすような構成がとられた。また、各実験担当者も 9 月合宿での問題を解決して 3 月合宿に臨んだため、3 月合宿では比較的順調に実験が行なわれた。

1.1 9 月合宿ネットワーク

1.1.1 実験概要

【OSPF6】 OSPF の IPv6 サポートの検証するため、合宿内に 9 個のユーザセグメントを作って、OSPF で経路制御を行なった。

【トラブルチケット・システム】 トラブルおよびその対応を記録するシステムの導入。

【対外線】 3 本のリンクを束ねて 1 本にするマルチリンク技術の実験と、バルク通信を衛星回線へ回す首振り機能の検証。

【IPv6 対応ダイアルアップ・ルータ】 ヤマハ RTA50i に実装した IPv6 サポートの動作確認を対外線を使って行なった。

【diffserv】 COPS 実装とトラフィックコントロールを連動させ、DV ストリームを流す実験を行なった。

【IPv6-IPv4 トランスレータ】 IPv6 と IPv4 を変換するトランスレータの動作検証。

【vDSL】 BOF 部屋が離れているため、ホテルの構内電話回線を使った vDSL による接続を行なった。

図 1.1 にネットワーク構成を示す。

1.1.2 対外接続

128Kbps の臨時専用線を 3 本と 2Mbps の衛星回線を利用した。接続先はいずれも藤沢 NOC である。3 本の専用線は、複数のリンクを束ねて仮想的に 1 本のリンクを実現するマルチリンク技術の検証に用いられた他、うち 1 本をダイアルアップ・ルータの IPv6 対応の検証にも使った。

1.2 3 月合宿ネットワーク

1.2.1 実験概要

【1.5M 臨時専用線による対外接続】 JGN 山梨 NOC に 1.5M ATM over T1 で接続することにより、高速な対外接続線を実現。

【ターミナルルームの設置】 プレナリルームの隣室にターミナルルームを設置。ターミナルルームでは DHCP により WIDE のグローバル・アドレスを取得可能とした。

【OSPF6 のマルチホーム環境】 IPv6 の経路制御には OSPF6 を利用した。IPv6 は藤沢 NOC および奈良 NOC と接続され、camp-net が OSPFv6 的なループの一部となる構成を取った。

【VLAN 技術による接続性の提供】 プレナリルームおよび各 BOF 部屋に敷設されるネットワークでは、IPv4 プライベートアドレス空間が利用され、NAT を経由して外部に接続する形態をとった。NAT と合宿プライベートセグメントを接続する部分には、物理リンクを複数の仮想リンクに分割するデータリンク抽象化システムを配置して実験を行なった。

【対外線優先制御サービス】 対外線の 1.5M はひとつの (src, dst) の組にたいして 64Kbps の帯域を予約することを可能とした。また、参加者に予約を行っていない場合と予約した場合の違いを体験してもらう目的で、時折対外線に負荷をかける実験を実施した。

【IPv6 DNS 逆引登録サービス】 IPv6 の DNS の逆引を登録するシステムを運用した。

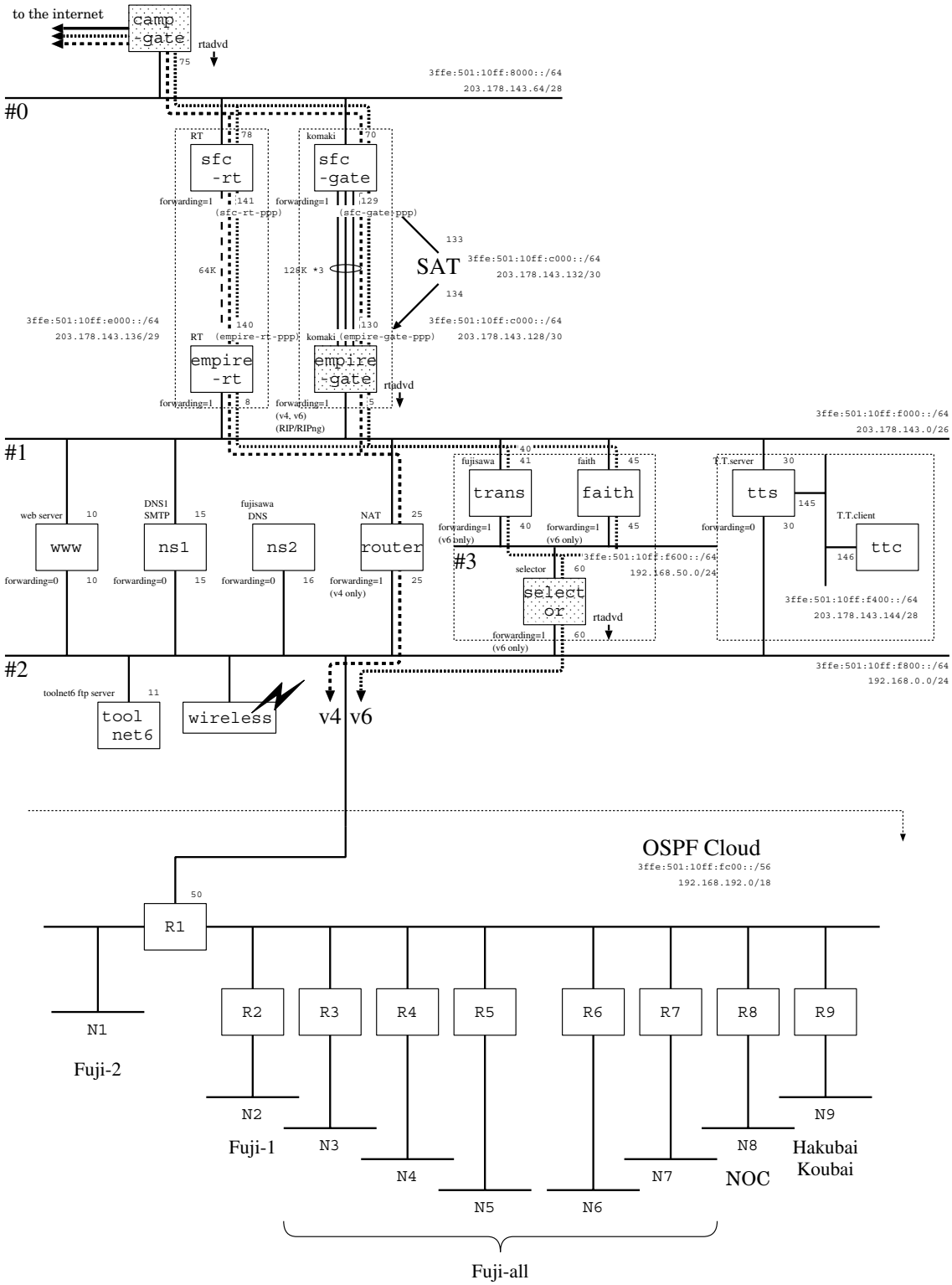


図 1.1　9 月合宿トポロジ

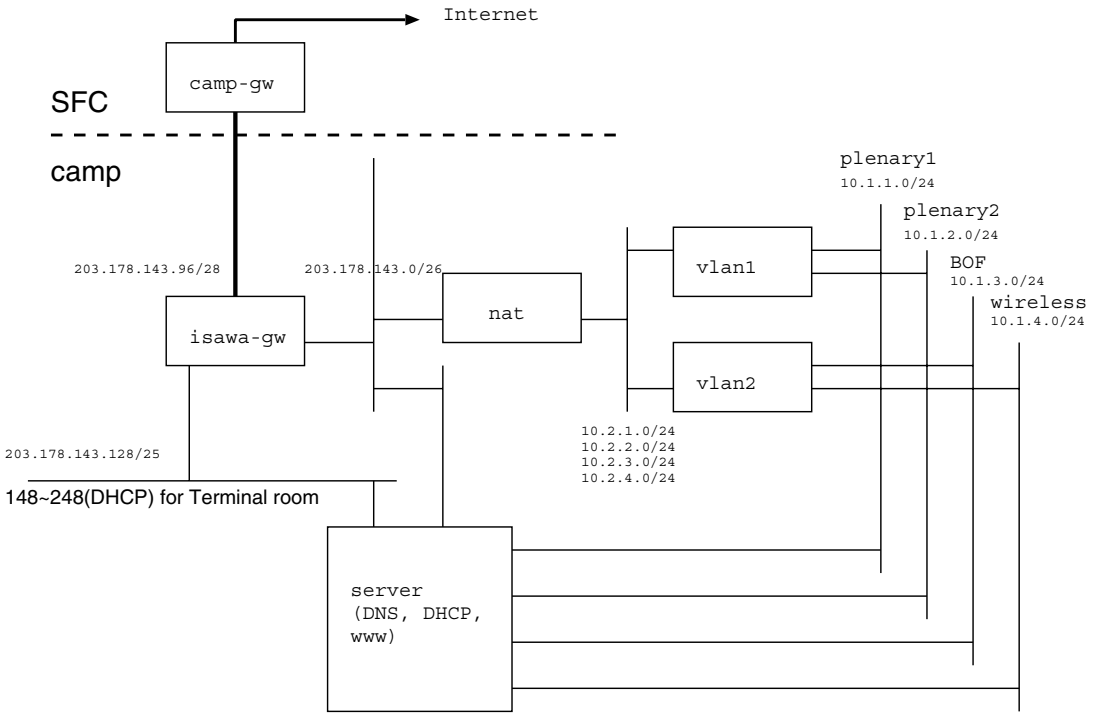


図 1.2 3 月合宿 IPv4 トポロジ

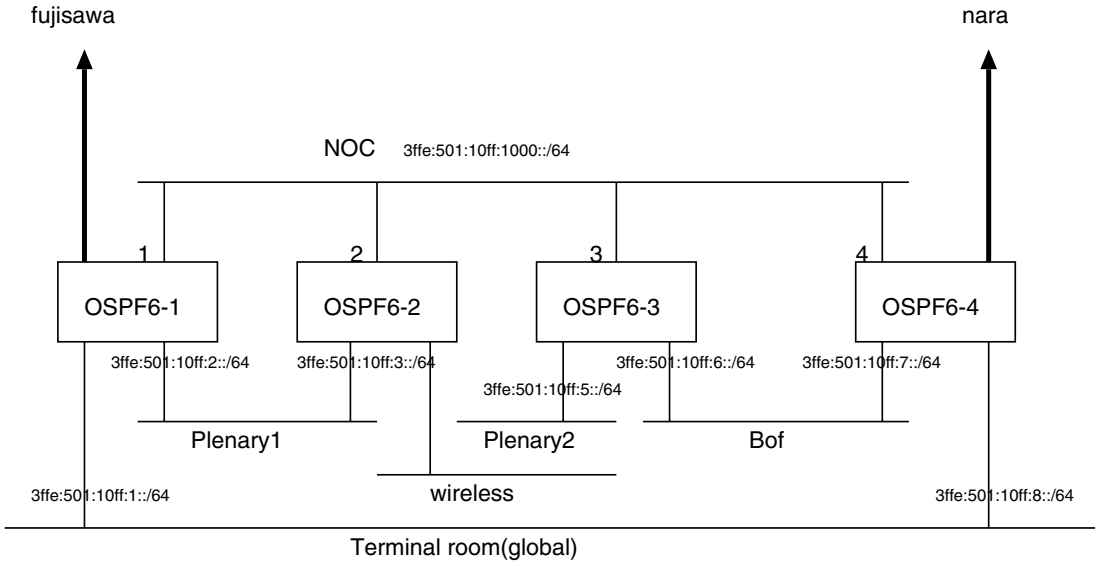


図 1.3 3 月合宿 IPv6 トポロジ

【ネットワーク監視】 トラブルチケット・システム、ネットワーク監視システム、トラヒック情報収集システムといったネットワーク安定運用のためのシステムを運用し、トラブルの早期発見、担当者への適切なディスパッチおよび迅速なトラブルシューティングを試みた。

図 1.2 に IPv4 の、また、図 1.3 に IPv6 のネットワーク構成を示す。

1.2.2 対外接続

3 月合宿では、JGN のご好意により JGN 山梨 NOC を経由した ATM 接続が実現した。対外線接続に ATM が利用できたためトポロジ構成の自由度が高まり、その結果、物理的な接続、2 層の ATM 接

続、3 層の IPv4 の接続、3 層の IPv6 の接続が異なるトポロジ設計となった。

物理的な接続は、T1 臨時専用線によって合宿地と JGN 山梨 NOC を接続した。ATM 的には 3 本の VC を使って、IPv4 用の VC を WIDE 藤沢 NOC に、IPv6 用の VC 2 本を WIDE 藤沢 NOC と WIDE 奈良 NOC に接続した。これらの VC は、合宿地と JGN 山梨 NOC 間は T1 上に ATM を通し、JGN 山梨 NOC と WIDE 東京 NOC までを JGN の ATM 回線を利用して接続、その後、藤沢と奈良に VC を引き込む構成とした。また、ATM レベルでのシェーピングを、VC 3 本まとめてボトルネックの 1.5Mbps に合わせて行なう必要があった。そこで、奈良へ接続した IPv6 用 VC も一度藤沢に引き込み、藤沢と合宿地において VC 3 本まとめてシェーピングする構成をとった。したがって、3 層からみると、IPv4 的には藤沢との 1 対 1 接続、IPv6 的には藤沢と奈良の 2 点に独立した接続を持つようになっており、かつ、IPv4 と IPv6 は完全に独立している。

第 2 章　合宿ネットワーク実験内容

本章では、2 回の合宿で行われた実験の具体的な内容について報告する。

2.1 データリンクの抽象化に関する実験

2.1.1 背景

IP に仮想リンクを提供する技術として、現在 Multi Link や VLAN、首ふりの技術が知られている。Multi Link とは複数の物理リンクを束ねて仮想的に 1 本のリンクを構築する技術である。Multi Link により物理リンクの存在は隠蔽され、IP は束ねられた仮想リンクのみを認識する。VLAN は Multi Link とは逆に、1 本の物理リンクを仮想的に分割し、複数の仮想リンクを構築する。VLAN により 1 本の物理リンクの存在は隠蔽され、IP は分割された仮想リンクのみを認識する。また首ふりは、パケットの属するフローによって、そのパケットの次ホップを自由に設定できる技術である。従来 IP では、パケットの宛先アドレス (dst addr) のみによって経路制御を行なっていたが、首ふりでは (src addr、dst addr、src port、dst port、proto) の組によって経路制御を行なう。首ふりは、たとえば図 2 のように専用線と衛星線がリンクとして利用できる場合に、バルクデータは衛星線、インタラクティブデータは専用線といったような経路分けをすることができる。ここで、首ふりを図 2 のように Point-to-Point で行なう場合に限定して考えると、これは一種の Multi Link と見ることができる。IP は束ねられた 1 本の仮想リンクに対してパケットを転送し、仮想リンクの中で首ふりを行なうのである。

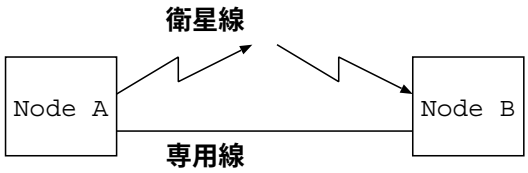


図 2.1　Point-to-Point での首ふり機構

これらの技術の実装例としては、Multi Link は PPP、首ふりでは mpath[217] などが挙げられる。既存の実装では、独自に IP に手を加えたり、デバイスドライバの部分で対応するなどの方法をとっている。たとえば、既存の首ふり技術である mpath では、IP 層で複数の次ホップを保持するようにし、パケットの処理中にどの次ホップに転送するかを選択する機能を追加することで首ふりを実現している。また既存の Multi Link 技術を実現している PPP では、デバイスドライバのレベルで複数のリンクを束ねた仮想リンク用のデバイスを用意し、デバイスレベルで Multi Link を実現している。

2.1.2 本実験の目的

前節で述べたように、IP に仮想リンクを提供する既存の技術では、各技術毎に IP やデバイスドライバに独自の変更を加えているのが現状である。これは、従来の TCP/IP の実装モデル (図 2.2) では IP とデバイスドライバが隣接していることに起因している。IP とデバイスドライバが隣接していることにより、IP に仮想リンクを提供するためには、IP やデバイスドライバのレベルで各技術が独自に実装をするしかないからである。

しかしこのような手法を採用しているということは、IP に仮想リンクを提供する新たな技術が提案されるたびに、IP やデバイスドライバをさらに変更し

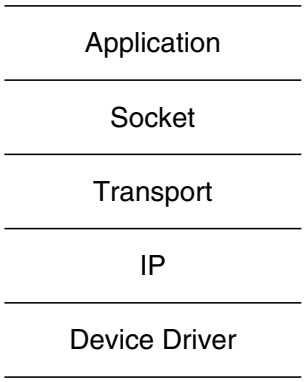


図 2.2 従来の TCP/IP 実装モデル

ていくことになる。これは、IP に仮想リンクを提供する枠組としては汎用性に欠ける。

そこで本実験では、IP とデバイスドライバの間に、データリンクを抽象化し仮想リンクを上位に提供する層を設ける。この層を ADL(Abstract DataLink) 層と名付ける (図 2.3)。ADL 層は、IP に仮想リンクを提供する技術を実現するべき適切な層といえる。IP に仮想リンクを提供する技術は全て ADL 層で閉じられ、新たな技術が登場しても ADL 層で実現することにより、IP やデバイスドライバには全く手を加えない。

ADL 層では下位の物理デバイスを IP から隠蔽し、IP に対して抽象化された仮想リンクを提供する。

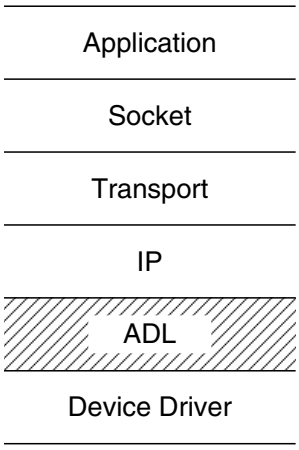


図 2.3 本実験で提案する実装モデル

2.1.3 1999 年秋合宿における実験

1999 年秋合宿における実験では、次のような実験を行なった。

1. ネットワーク構成

秋合宿での対外線トポロジを図 2.4 に示す。

今回、対外線は 128K の専用線を 3 本と ISDN 回線を 1 本、さらに SFC から合宿地への 2M の衛星線を用意した。ISDN 回線はヤマハの IPv6 対応ルータ (RT) 実験で使用した。これを RT 線と名付ける。本実験では 128K の専用線 3 本と衛星線を利用し、それらの線を接続する一対のルータ (sfc-gate、camp-gate) を実験マシンとして使用した。以降では、本実験で使用した 128K の専用線および 2M の衛星線を対外線と呼び、それらを接続するルータを対外線ルータと呼ぶ。

合宿地の IPv4 ホストは NAT ルータである nat のプライベートセグメント内に位置し、IPv4 トラフィックは全て nat を経由して RT 線および対外線を通る。IPv6 ホストにはグローバルアドレスが割り当てられ、IPv6 トラフィックは全て selector を経由して RT 線、対外線を通る。selector は IPv6 首ふり機構を備えており、外向きの IPv6 トラフィックは selector においてプロトコルによって trans、faith それぞれに経路を振り分けられる。

トラフィックが RT 線と対外線のどちらの線を通るかは静的経路で設定される。その経路は、内向きのトラフィックは camp-gate、外向きのトラフィックは nat や faith、trans において設定される。

2. 実験内容

本実験では、SFCNOC と合宿地にそれぞれ設置した対外線ルータにおいて、ADL 層を実装する。本実験は、複数の物理リンクを ADL 層で隠蔽し、IP に対して仮想リンクを提供することを目的とする。そのため、合宿で利用する対外線は複数の異なる物理リンクが望ましく、128K の専用線 3 本と 2M の衛星線という対外線構成になった。128K の専用線 3 本と衛星線は ADL 層で抽象化し、IP には 1 本の仮想リンクを提供する。ADL 層で対外線を抽象化する際、128K の専用線 3 本を Multi Link によって束ね、その束ねた 384K の仮想リンクと衛星線をさらに Multi Link する。この 384K の仮想リンクと衛星線の間では図 2 のように首ふりを行ない、IP

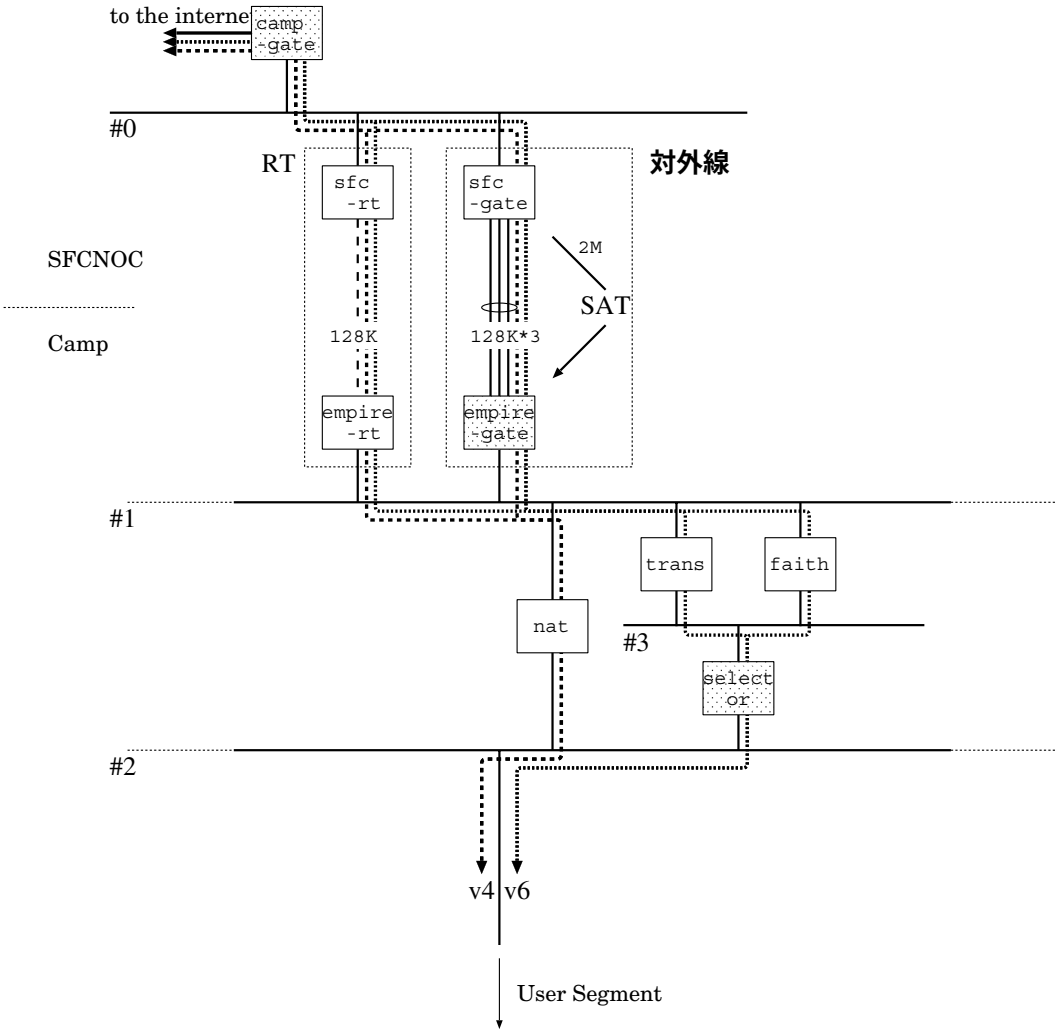


図 2.4　対外線トポロジ

からは単に 1 本のリンクが認識される。

また本実験では、ADL 層は userland ppp を用いて実現する。userland ppp を使用する理由は以下の通りである。

- PPP は概念的に IP とデバイスドライバの間に位置する
- PPP では multi link が既に実現されている
- userland での実装であるため実装、デバッグが容易である

具体的には、userland ppp に多段 Multi Link や首ふりの技術を追加し、userland ppp が使用するトンネルデバイスを仮想リンクとして IP に提供する。

3. 結果

今回の実験は結果としてはうまくいかなかった。以下にその理由を示す。

(a) TA 問題

3 本の専用線を接続するためには計 6 台の TA が必要であり、6 台の TA は機材募集により確保はできていた。しかし、128K での専用線接続に対応した TA を前もって調べておかなかったため、6 台の TA の中に 128K での専用線接続に対応していない TA がいくつかあった。それに気づいたのが合宿直前で、camp-net チームを混乱させ、新たな TA を確保するのに手間取ってしまった。代替策として、TA をルータとして機能させることで 128K 接続を実現できるため、その上で PPP over TCP

の技術を利用することにより TA の問題は解決した。

結局、合宿での TA の構成は専用線 3 本のうち専用線接続が 2 本、ルータ機能を使用して PPP over TCP での接続が 1 本で行なうこととなった。

(b) 接続問題

合宿での対外線状態は良好ではなかった。まず衛星回線が機材の故障により合宿を通して使用できなかった。また、3 本の地上線の使用状況以下の通りだった。

- 1 本目 (PPP over TCP) ··· 前泊時に接続

  1 本目の TA に関しては、SFCNOC に設置してきた TA はシリアルケーブルで合宿地のルータと接続し、合宿地からリモートで設定できたため、前泊日につながった。

- 2 本目 (専用線接続) ··· 合宿 1 泊目の昼に接続

  2 本目、3 本目の地上線の使用開始が遅れたのは、専用線を接続する両端の TA の設定に手間取っていたからである。2 本目、3 本目の TA に関しては、設定の変更を反映させるためには本体の電源を ON、OFF させる必要があった。そのため、SFCNOC に設置してきた TA の設定を変更する場合に物理的に電源を操作する必要があり、リモートで設定するのに非常に手間取った。

  2 本目の地上線に関しては、合宿当日に SFC にいた人に電話を通じて電源を操作してもらいながら設定し、合宿 1 日目の昼にアップした。

- 3 本目 (専用線接続 → & PPP over TCP) ··· 合宿 3 日目の午前に接続

  3 本目の地上線は、合宿地側に接続する TA と PC を接続するシリアルケーブルを忘れてしまい、それをどうやって接続するかで非常に手間取った。この TA のシリアルケーブルは特殊な配線のため、合宿地でケーブルを作成してみるなどいろいろ試したが、結局接続できなかった。最終的にはこの TA の使用をあきらめた。

  2 日目の夜に camp-gate が SFCNOC の電源問題によりダウンしたため、問題解決のために SFCNOC に戻った camp-net チームのメンバに SFC 側の TA を交換してもらった。この際、代替 TA はルータ機能を用いて 128K 接続する機種であったので、3 本目の地上線には PPP over TCP を利用した。3 本目の地上線は合宿 3 日目の午前に接続した。

結局、合宿中の対外線の帯域は、以下のようになった。

- 合宿 1 日目〜合宿 3 日目午前 : 対外線 256K + RT 線 128K
- 合宿 3 日目午後〜 : 対外線 384K + RT 線 128K

(c) PPP 問題

PPP はリンクレベルでのネゴシエーションを行なうため、ネゴシエーションパケットは往復で同一インタフェースを使用する必要がある。しかしこれは、PPP over TCP のトンネルや片方向の衛星線では問題があった。なぜなら、片方向の衛星線では復路のパケットが衛星線を通らず、PPP over TCP では往復のパケットが同じ専用線を通らない場合があるからである。この問題を解決するには、インタフェースに対するエイリアスアドレスの設定や、別経路の設定が必要である。3 本目の地上線を接続する際にこの問題に直面し、この解決法を見つけ設定を行なうまでに相当量の時間を要した。

(d) 接続性問題

対外線は接続された後も良好とは言えず、非常に混雑していた。その理由を以下に示す。

- 合宿中の対外線は、衛星線が利用不可能だったために地上線に全トラフィックが集中してしまい、非常に混雑した
- userland ppp であったためにカーネル内での処理に比べて余計に負荷がかかった
- PC 側のシリアル回線にオンボードのシリアル端子を利用したため、115.2K しか出なかった
- PPP 上で圧縮をかけなかった

当初は衛星線はあるものと考えており、バーストトラフィックは全て衛星線を通るという前提があった。そこで地上線に対する混雑度は低いと予想したため、userland ppp の使用を決定した。しかし衛星線が機材の故障により使用不可能となったため、そのトラフィックが全て地上線に集中し、このような事態になってしまった。

4. トラフィックログ

合宿中のトラフィックを

トラフィック 1

2 日目 14 時 から 3 日目 10 時

トラフィック 2

3 日目 11 時 から 3 日目 20 時

の 2 回、empire-gate の#1 へのインタフェースに対して tcpdump により測定した。この結果に対して、合宿地のアドレスを用いてフィルタをかけることにより、対外線を通ったトラフィックを抽出した。なお、#1 は実際にはスイッチであり、empire-gate は 100Base-TX、RT は 10Base-T で接続していたために empire-gate から RT へのトラフィックは測定されなかった。よって、ここで測定したトラフィックは対外線を流れたパケットのみである。

トラフィック 1、トラフィック 2 をそれぞれ以下のように分類した。

(a) IPv4 と IPv6 の様子
(b) IPv4 と IPv6 の外向き、内向きのそれぞれの様子
(c) IPv4 外向き TCP トラフィック中の各アプリケーションの様子
(d) IPv4 内向き TCP トラフィック中の各アプリケーションの様子
(e) IPv6 外向き TCP トラフィック中の各アプリケーションの様子
(f) IPv6 内向き TCP トラフィック中の各アプリケーションの様子

ここで、上記の 1〜6 の分類に対して、トラフィック 1 の各グラフを図 2.5、図 2.6、図 2.8、図 2.7、図 2.10、図 2.9 に示す。同様にトラフィック 2 の各グラフを図 2.11、図 2.12、図 2.14、図 2.13、図 2.16、図 2.15 に示す。

合宿中、主に IPv4 トラフィックは対外線へ、IPv6 トラフィックは RT 線へ振り分けていた。つまり、このグラフに現れる IPv6 トラフィックの原因としては以下のものが考えられる。

- #1 の各ホストが静的にデフォルト経路を empire-gate に向けていた
- tans を経由せずに他のルータを通ってきた
- その時だけ IPv6 トラフィックを RT 線に向けずに対外線に向けていた

合宿地ではトラフィックのモニタを行なっていなかったため、合宿中にこの問題には気づかなかった。もしモニタを行なっていればその場で確認できたはずであり、これは次回への反省点である。

5. 反省点

今回の合宿には様々な反省点がある。以下にそれを述べる。

- ホットステージ等の事前の準備において、合宿のトポロジをシミュレーションしたテストが不十分であった。配線等も含めて、より正確なテストが必要である。
- 非常事態にも対処可能なように、予備のパスを用意できる PC を使用すべきである。
- 対外線トラフィックを合宿地において常にモニタし、トラフィックの変化や誤りの検出、トラブルシューティングに用いるべきである。

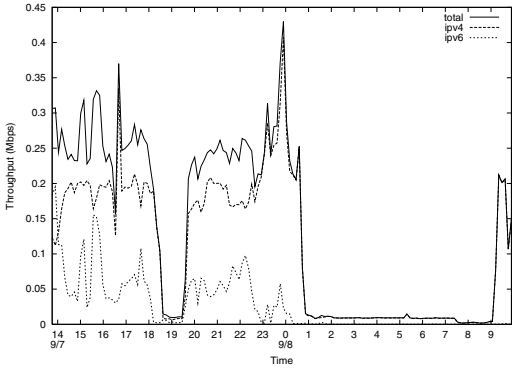


図 2.5　トラフィック 1 における IPv4 と IPv6 の様子

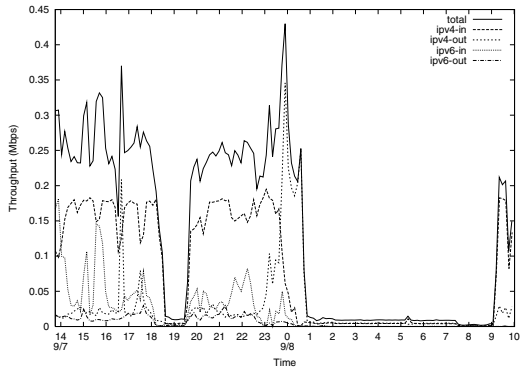


図 2.6 トラフィック 1 における IPv4 と IPv6(外向き、内向き) の様子

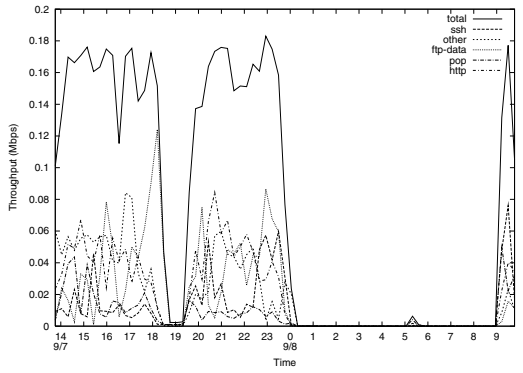


図 2.7 トラフィック 1 における IPv4 内向きの各アプリケーションの割合

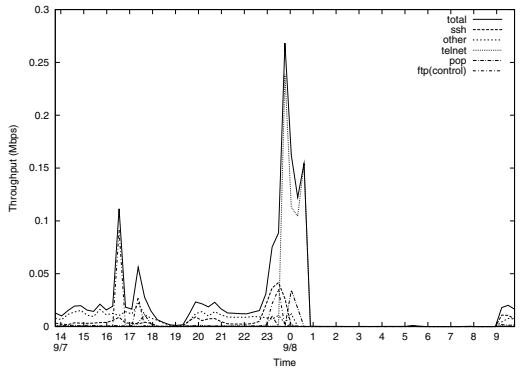


図 2.8 トラフィック 1 における IPv4 外向きの各アプリケーションの割合

2.1.4 2000 年春合宿における実験

2000 年春合宿における実験では、次のような実験を行なった。

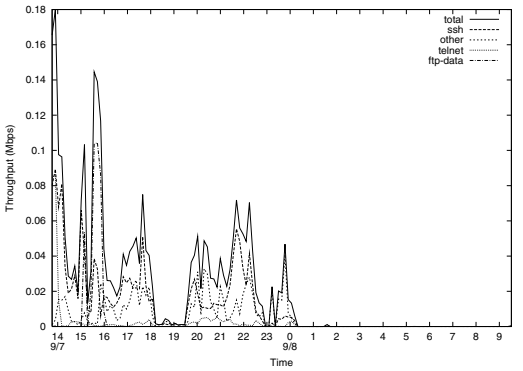


図 2.9 トラフィック 1 における IPv6 内向きの各アプリケーションの割合

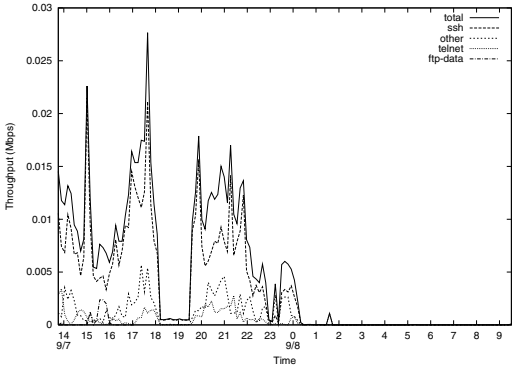


図 2.10 トラフィック 1 における IPv6 外向きの各アプリケーションの割合

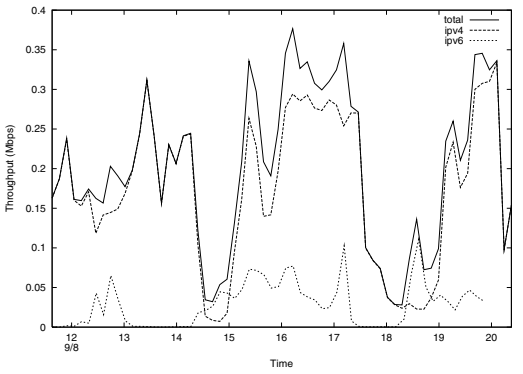


図 2.11 トラフィック 2 における IPv4 と IPv6 の様子

1. 実験内容

合宿地内では、IPv4 ホストは NAT ルータである nat のプライベートセグメント内に位置し、IPv4 トラフィックは全て nat を経由して外部との通信を行なう。IPv6 ホストにはグローバルアドレスが割り当てられ、IPv6 トラフィックは全て OSPFv6 ルータを経由して通信を行なう。

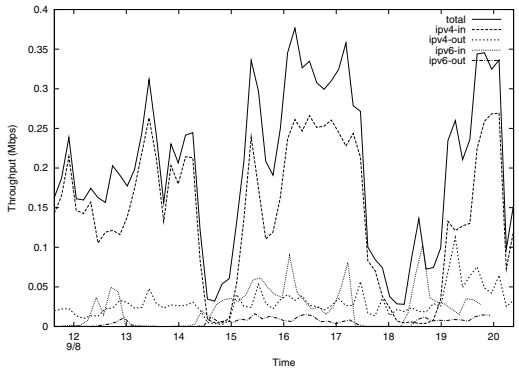


図 2.12　トラフィック 2 における IPv4 と IPv6(外向き、内向き) の様子

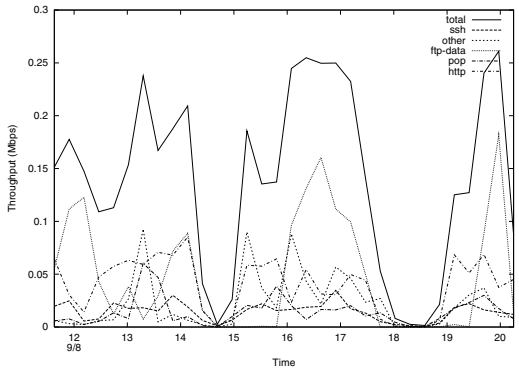


図 2.13　トラフィック 2 における IPv4 内向きの各アプリケーションの割合

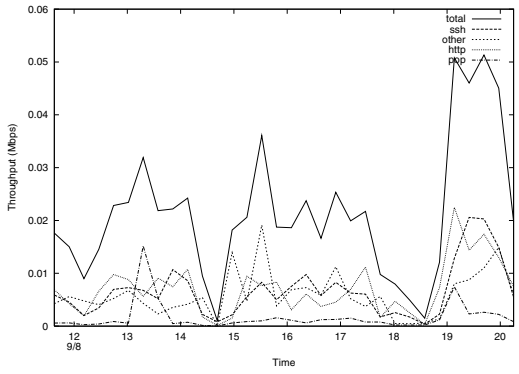


図 2.14　トラフィック 2 における IPv4 外向きの各アプリケーションの割合

本実験のトポロジを図 2.17 に示す。本実験では、ADL 層をカーネル内で実装したシステムの実験を行なう。nat 配下の合宿地内のネットワークにおいて、nat、vlan1、vlan2 の間で VLAN セグメントを構築し、ADL 層によって 1 本の物理リンクを隠蔽して 4 本の仮想リンクに分割

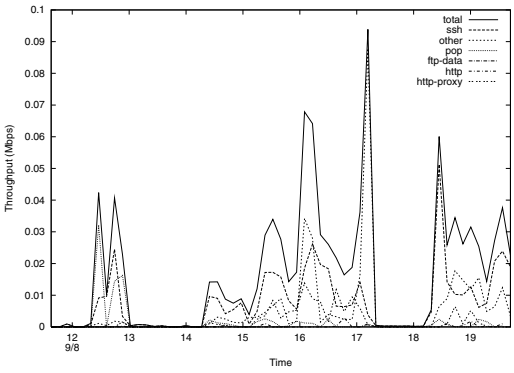


図 2.15　トラフィック 2 における IPv6 内向きの各アプリケーションの割合

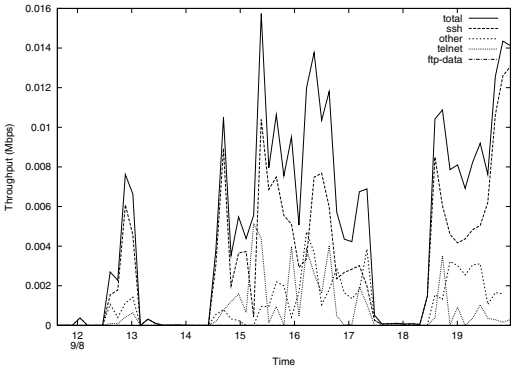


図 2.16　トラフィック 2 における IPv6 外向きの各アプリケーションの割合

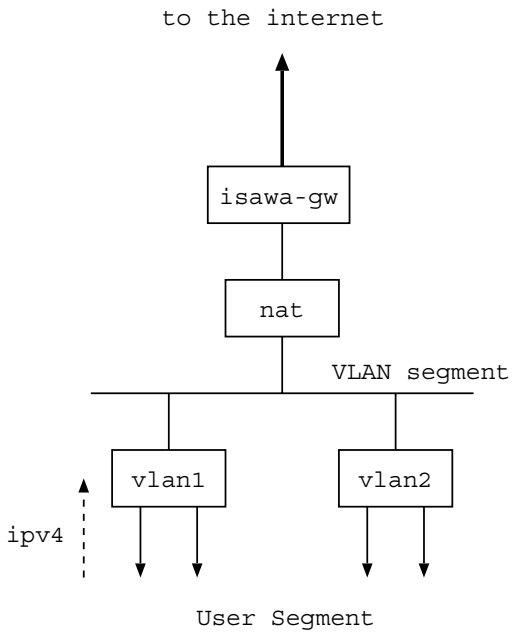


図 2.17　実験トポロジ

する (図 2.18)。nat を経由するトラフィックは IPv4 のみであるため、VLAN システムを利用するのは IPv4 トラフィックのみである。

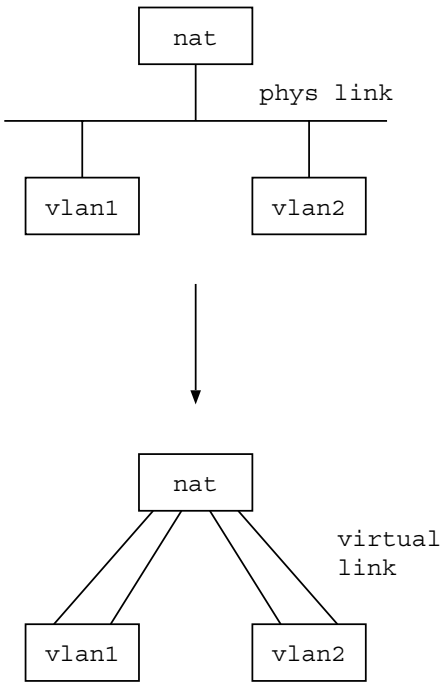


図 2.18 VLAN

2. 結果

今回の合宿では、前回の 1999 年秋合宿に比べると格段に安定したネットワークを供給できた。これは、以下のような理由が挙げられる。

- userland での実装ではなくカーネル内実装であったために負荷に対してある程度頑強であった
- 対外線での実験でなく合宿地内での実験であったために機材の問題もなく、事前の合宿トポロジのシミュレーションを容易に行なうことができた。
- 合宿前のホットステージにおいて、ほぼ完全に合宿トポロジを構築し、検証を行なうことができた。この時点で問題なく動作していた。

3. 今後の課題

今回の実験では、ADL 層において VLAN を実現する実験であったが、今後は VLAN のみでなく、Multi Link や首ふりなども組み合わせたシステムも必要となってくる。また、今回の実験中、一度仮想リンクの IP アドレスが変更されたり、ある特定のフラグメント化されたパケットの通信において不具合が生じるなどの問題があった。これは、nat での natd との連係も含めて、より突き詰めていかなければならない。

2.2 IPv6 に対応した ISDN ルータによる対外線接続

2.2.1 実験の目的

家庭や小規模オフィスで広く用いられている ISDN ルータに，IPv6 に対応したファームウェアを試験的に実装する．また，この ISDN ルータを用いて合宿ネットワークの対外接続環境を構成し，実装の安定性を検証する．

2.2.2 実験の概要

IPv6 に関する基本的な機能として，IPv6 ヘッダの処理，経路制御，近隣探索，および IPV6CP を選択し，これらを ISDN ルータのファームウェアに実装する．ISDN ルータが本来持っている機能については，全てを残すように設計し，IPv4 と IPv6 の両プロトコルを処理できるようにする．

経路制御と IPV6CP の実装は未完成であるため，利用できる機能は限定されている．まず，経路制御では，RIPng のような経路制御プロトコルは実装されておらず，静的な経路のみを扱う．また，IPV6CP では，IPv6 パケットをカプセル化する機能のみが実装されており，IPV6CP の設定を交渉する機能は実装されていない．

本実験では，以上の実装を搭載した ISDN ルータを用いて，合宿ネットワークと慶応大 SFC を ISDN 回線で接続する．これにより，ISDN ルータを用いた対外接続環境が良好に動作することを確認する．

2.2.3 実験環境

ISDN ルータとして 2 台の RTA50i を用意した．これらは，製品と全く同じハードウェアを持っており，特別な改造などは行われていない．これらのうち，1 台を慶応大 SFC に，もう一台を合宿ネットワークに接続した．2 台の間は，ISDN 回線で接続されてお

り，トラヒックの存在するときのみ回線を接続するように設定された．また，Multi-link PPP により，トラヒックの負荷に応じて 64kbit/s と 128kbit/s のいずれかの回線速度で接続するように設定された．

経路制御については，静的経路のみを用い，近隣のルータにおける経路の設定を定期的に変更する方法を取った．これにより，処理するプロトコルやトラヒックの方向などの条件を変化させて評価することができる．また，合宿ネットワークに接続された RTA50i は，SNMP や syslog によって情報を取得できるように設定された．

2.2.4 結果

3 日間連続して運用し，障害による停止が 2 回発生したが，おおよそ良好に動作した．障害はルータの実装に起因するものであり，ルータの記憶領域が解放されないために発生した．この問題は実装を見直すことで解消できた．定期的に経路の設定を変更し，プロトコルやトラヒックの方向を変化させて動作を監視した．最終日には，連続して 24 時間以上の運用を行うことができた．

なお，3 日間を通じて，ルータが処理した IPv6 のトラヒックは ISDN 回線の帯域を埋めるほど多くはなかったため，より継続的に高負荷の環境で評価することが必要であると思われる．

2.2.5 今後の課題

今後の課題としては，まず，実装の追加が挙げられる．特に，機能に制限のある IPV6CP については，実装を追加する必要があり，経路制御に関しては，RIPng を実装する必要がある．また，より定常的に評価と実装の安定化が不可欠である．特に，高負荷における動作など，評価が十分でないものについては，今後も評価を実施し，実装の安定化を図る必要がある．

2.2.6 関連情報

- 白砂哲，木本雅彦，大野浩之，木村俊洋，小規模 IPv6 ネットワーク用 INS ルータの実装，情報処理学会第 58 回 (平成 11 年前期) 全国大会論文集．

- 白砂哲，野田明生，木本雅彦，木村俊洋，大野浩之，IPv6 対応 INS ルータを用いた小規模ネットワークの構築，平成 11 年 マルチメディア通信と分散処理 95-18．

2.2.7 VDSL

実験の目的

ホテル既設の構内回線による vDSL リンクの運用実験および通信特性を評価する。具体的には、今回合宿地である遠鉄ホテルエンパイアの大会議室と紅梅／白梅間を接続し、その上で実験評価する。

実験の概要

1. ネットワークを運用しながら特性を調べる。
2. ケーブル長を延長した場合のパケット落ちを観測する。
3. スループットや遅延を計測する。

実験環境

遠鉄ホテルエンパイアの大会議室と 紅梅／白梅の BOF 部屋を、ホテル既設の構内線を利用して接続する。当初、各部屋に引かれてある構内線はホテル内交換機室の E350 交換機に収容されていた。各部屋に引いてある線を、この交換機から外して MDF の上に直接つないだ。

各部屋では、それぞれ RJ-11 のラインよりスプリッター、ルーター、モデムの順番に接続した。このセグメントは９つある利用者用の運用セグメントの１つである。

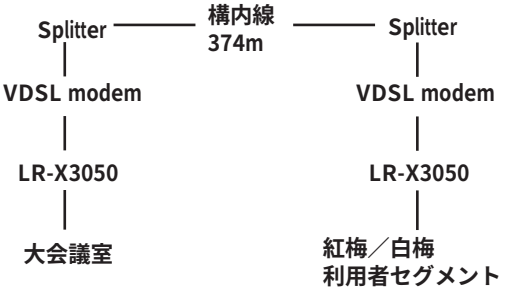


図 2.19　VDSL 運用／実験環境

LR-X3050:www.fujitsu.co.jp/hypertext/Products/telcom/lan/lrx3050.html

VDSL:www.orckit.com/orckit_products.html

結果

1. 距離による測定

まず、大会議室と紅白梅までの構内線の長さをFLUKE で測定した。構内線は、ホテル一階にある交換機室を経由して、もう一方の部屋に繋がる。

表 2.1 FLUKE による構内線の特性

モード	Twin-AX: RJ45 4,5 pin
距離	374.2 m
インピーダンス	94 ohm
反射ピーク	86 % (388.6m)

２つの部屋を結ぶ構内線に、あらかじめ準備した 2 芯線を 200 m 単位 (最長 1km) に繋いで、富士から紅梅方向に ping -f によるパケット落ち (単位 %) を調べた。９月７日午前２時頃に、vDSL モデムの２種類の転送モード (12.5, 25.0 Mbps) で試した。

表 2.2 伝送距離に対するパケット落ち

距離 [m]	12.5 Mbps [%]	25.0 Mbps [%]
374	0	0
574	0	0
774	0	0
974	0	10
1174	0	100
1374	0	100

準備した線は巻き取りの状態で測定したので、伝送条件は悪いと考えられるが、この試験結果から 12.5Mbps では少なくとも 1374 m まで利用することができ、25.0 Mbps の場合には、774 m まではパケット落ちなく利用できることが分かった。

2. 時間帯による性能特性

合宿初日夜から２日目午前１０時位まで、下りの伝送速度を 25.0 Mbps で運用した。しかし、２日目午前９時位に IP レベルでは 30-50 % 落ちることが判明したので、この時点より 12.5 Mbps モードに切替えて運用した。

その後、25.0 Mbps モードでのパケット落ちのない時間帯を見つけるため、確認する時だけ 25.0 Mbps モードに切り替えて、パケット落ちを確認したら 12.5 Mbps モードに戻す作業を 1 時間ごとに繰り返し行なった。その結果、夜間 (午後１０時〜午前９時位) にはパケット落ちがなくなることが分かった。

表 2.3 時間帯ごとの通信状態

	昼	夜
12.5 Mbps	◎	◎
25.0 Mbps	×	○

昼間になるとパケットが落ちが増加する理由は、ホテルのアナログ電話が昼間、頻繁に利用されていて、電話線を通過する複数のアナログ信号が vDSL のアナログ信号に影響したものと考えた。

実際、交換機室のダクトから垂れ下がる構内線は、数百回線分を束ねて扱われており、MDF パネルの直前で一本づつ分けてパネルに付いている。今回の構内線もこの内の１本であり、交換機室までは他の構内線同様に束ねて扱われている。

3. VDSL の性能評価

vDSL ネットワークの TCP/IP の性能を評価するために、PC 機 (BSD/OS 3.1, 100M イーサネット) から netperf ツールを用いてスループットを測定した。測定には、netperf 添付のストリーム用スクリプトを利用した。

TCP: Send, Rcv Socket Size=57344 byte, Message size=4096 byte

UDP: Send Socket Size=57344 byte, Message size=1472 byte

遅延 (大会議室と紅白梅の間)

ping の RTT は 27-30 msec 程度であった。

まとめ

ホテル構内線に vDSL 機器を接続して、合宿の運用セグメントの一部を実現し、紅白梅の利用者にネットワーク環境を提供した。

vDSL モデム 25.0 Mbps のモードでは、昼間にパケットロスが発生してしまうことが分かったため、パケットロスのない 12.5 Mbps のモードで運用を行うことで、合宿期間中安定したネットワーク環境を実現した。

表 2.4　伝送距離に対するスループット

	TCP 25Mbps mode	12.5 Mbps mode
構内線 374[m] 下り	14.12 Mbps	11.76 Mbps
上り	1.76 Mbps	1.75 Mbps
LR 直結 下り	20.08 Mbps	20.08 Mbps
VDSL 200[m] 下り	13.82 Mbps	11.74 Mbps
VDSL 400[m] 下り	13.82 Mbps	11.69 Mbps
VDSL 200[m] 下り	13.82 Mbps	11.74 Mbps
VDSL 200[m] 上下	95.71 Mbps UDP	95.69 Mbps UDP

2.3 Translator

2.3.1 実験の目的

IPv6 から IPv4 への translator を 実際の使用環境に近い環境で動作させる。

2.3.2 評価

99 年 夏合宿の実験は、translator をうまく動作させることができなかったため、失敗した。

Translator を動作させることに失敗した原因は特定できなかったが、trouble shoot に失敗したことについては 以下のように考えられる。

- Kernel debug に頼りすぎたため、kernel debug のできない環境で動作させると、正常に動作しない場合に 何が原因で動作しないのかの特定ができなかった。

2.3.3 対策

Kernel debug できない環境で translator を動作させることを考慮し、以下のような改良を施した。

- Log 出力の整備 Command とは別に translator からの出力を拾う daemon を動かし、この daemon 経由で kernel からの出力を表示する。 default での log 出力は 標準出力に行くが、daemon 立ち上がり時の指定で syslog 経由にすることもできる。

- Dump point の整備 Translator 内部に 複数の dump point と置き、command line からの指示で IP packet あるいは mbuf の dump を取ることができるようにする。 Dump point は最大 32 箇所、個々の dump point について on/off が可能である。 Dump point からの出力は 上記した daemon によって解釈され、人に読み易い形で出力される。

この効果としては 以下のようなものがある。

- Dump point を置き、実行時に command line から指定することで 必要な処での処理が目にみえるようになった。

- 指定された dump point での出力がなされない時には、program 上でこちらの意図した流れと異る流れを packet が通ってきているということなので、この点からも 問題点の発見が早くできるようになった。

- これは trouble shoot とは直接の関係はないが、Log 出力を常時出し続けていくことで、translator の内部の動きが ある程度見えるようになった。

2.4 Differentiated Services

2.4.1 実験の目的

Differentiated Services(Diffserv) [218, 219] は優先制御によってサービスの統計的区分化および通信品質を統計的に保証するための枠組みであり、IETF の Diffserv WG によって検討されている。この枠組みでは、複数のフローを統合することによってスケーラビリティを実現している。また、様々なキュー制御アルゴリズムとアドミッション制御 (ポリシの適用、資源配分機構等) を組み合わせることにより、サー

ビス提供者が特色あるサービスクラスを定義し、品質保証サービスを構築・提供することができる。

Diffserv WG ではサービスモデルに関する議論が繰り返されてきたが、IETF の『ポリシとメカニズムの分離』という方針のもとに多様なポリシを実現するメカニズムだけが規格に盛り込まれている。このため、ポリシが多分に影響すると思われる『DS ドメイン間の関係』や『DS ドメインの種類』等については議論されていない。

我々は、DS ドメイン間の関係が階層的なものであるモデルを考えている。このモデルでは、末端の DS ドメイン (各組織) においては個々のユーザが直接ネットワークに対して資源予約を要求する。しかし、このような動的資源予約に関する実験はほとんど行われておらず、ユーザの挙動やシステムに必要な機構、動的資源予約の問題点等に関してはほとんど明らかになっていない。

今回の合宿では、上記の点を明らかにするためにユーザからの動的な要求に対するネットワークの資源予約システムを構築し、運用実験を行った。

2.4.2 実験の概要

実験は、慶應義塾大学・奈良先端科学技術大学院大学・大阪大学の Diffserv 研究チームが合同で行った。

予約する資源としては、ATM による 1.5Mbps の対外線を対象とした。今回の合宿ネットワークは IPv4 および IPv6 の対外接続を提供するため、制御しなければいけないルータが多数にのぼる。資源予約の運用負荷を下げるため、PVC ブリッジと呼ばれる技術を導入した。これにより、IPv4 ルータは IPv6 の PVC ブリッジとしても機能し、トラヒック制御を単一の機器で行うことが可能となった。

また、ネットワーク資源の永続的な占有を避けるため、合宿でのみ通用する仮想的な通貨を用意し、資源予約に対して課金を行うこととした。さらに、現在の予約状況および仮想通貨の残高を確認できるシステムを Web によって提供した。

ユーザには資源予約のためのクライアントを用意した。これは C 言語および Java によって提供され、ほとんどのプラットフォームから利用できるようにした。また、これらのクライアントは IPv4 および IPv6 の両方に対応した。

2.4.3 実験環境

図 2.20 に合宿ネットワークの第 2 層の構成を示す。

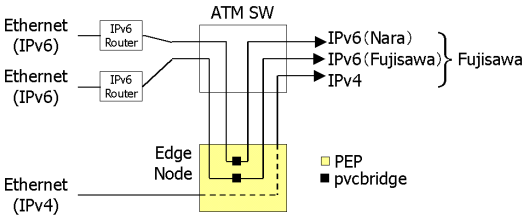


図 2.20 合宿ネットワーク (石和、第 2 層)

ATM による 1.5Mbps の対外線を ATM スイッチで収容し、さらに PC ルータ (isawa-gw) に接続されている。IPv4 の接続性は isawa-gw を経由して提供されている。また、ATM スイッチには 2 台の IPv6 ルータ (ospf1、ospf4) が接続されており、IPv6 の接続性はこの 2 台を経由して提供されている。IPv6 用の VC は ATM スイッチから一度 isawa-gw へ向かい、PVC ブリッジを経由して藤沢および奈良へ向かう。これによって対外線を第 2 層的にすべて isawa-gw に集めることができ、トラヒック制御を集中的に行うことができる。

トラヒック制御のために isawa-gw では ALTQ[220] をもちいている。トラヒックの制御は入力キューにおいて TRTCM[221] でマーキングし、出力キューにおいて HFSC[222]/RIO[223] で制御した。

奈良への VC は一度藤沢の IPv4/IPv6 ルータの PVC ブリッジを経由し、同様の構成をとっている。

この実験では対外線の 1.5Mbps を 19 ブロックに分割し、そのうち 18 ブロックは予約用、残りの 1 ブロックを通常のトラヒック用とした。予約用の 18 ブロックはそれぞれが 64Kbps の帯域を持つ。

合宿の参加者には、それぞれに 2000WU(WIDE Unit、仮想通貨) を配布した。それぞれの参加者は通信において (src, dst) の組で予約することができる。予約は 1 分 10WU とした。また、予約用には PEPe と呼ばれるクライアントを C 言語および Java で配布し、多くのプラットフォームで利用可能となるようにした。

実験中には予約を促進させるために人工的な輻輳を断続的に発生させた。特に 3 月 16 日の夜はかなり激しい輻輳を発生させた。

2.4.4 結果

図 2.21 に 1 分あたりの予約要求の数を示す。

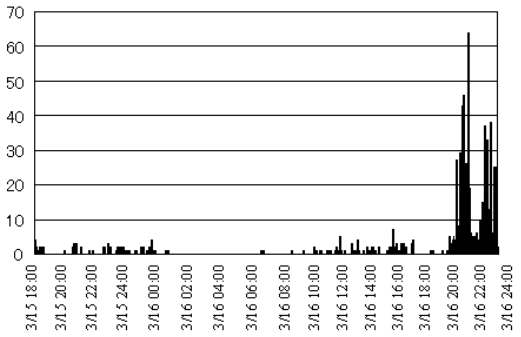


図 2.21　1 分あたりの予約要求の数

このグラフより 3 日目の夜の予約要求が特に多い。これは次の 2 つの原因が考えられる。1) ネットワークの輻輳しているときには、参加者が帯域に予約するだけの価値を見いだしている。この結果、輻輳時に予約が集中し、特に 3 日目の夜に激しい人工的輻輳を発生させたため、予約が集中したものと思われる。逆に、輻輳していないときには帯域の価値は低く見積もられ、それほど予約されていない。2) ほとんどの参加者にとって、帯域を予約するということははじめての体験であった。したがって、最初は帯域を予約することの効果や価値に懐疑的であり、なんらかの躊躇があったと思われる。しかし、その効果に気が付き、帯域を予約するという行為に慣れるにしたがって頻繁に予約するようになったと考えられる。このため、合宿が進むにつれて予約数が多くなり、3 日目の夜にもっとも多くなったものと考えられる。

以上のことから、ある状況においては帯域予約がユーザにとって重要な役割を果たすことがわかる。

2.4.5 まとめ

今回の合宿では、ユーザからの要求により帯域を予約するというシステムを運用し、実験を行った。対外線のみの予約という限られた環境にも関わらず、その有用性はユーザにも徐々に理解されつつある。今後はサービスクラスの充実および広域網における資源予約システムの設計と開発、および運用ネットワークへ適用していく必要がある。

2.5 OSPFv3 の実験運用

合宿ネットワークでは、OSPFv3 を用いて IPv6 の経路制御を行った。この実験を通じて、実装、運用それぞれにおいて様々な問題が発見された。

2.5.1 実験環境

合宿の IPv6 ネットワークは奈良 NOC と藤沢 NOC に接続された。さらに、合宿のネットワークは内部でループを形成するトポロジとした。このため、OSPFv3 により経路制御されるネットワークは、多数のループを形成するトポロジとなっている。合宿における IPv6 ネットワークトポロジを図 2.22 に示す。

OSPFv3 の実装には、Zebra の ospf6d を用いた。それぞれのルータには、FreeBSD3.4、KAME、Zebra をインストールした。

2.5.2 発見された問題

発見された問題には、実装に関するものと運用に関するものの二通り挙げられる。

- 実装に関するもの

  1. LSA の premature aging がされない
     LSA の premature aging による経路の削除が的確に行われていなかった。この問題は、合宿中に実装を改善することによって解決した。
  2. 他プロトコルからの経路をフィルタする機能の欠如
     他の経路制御プロトコルからの経路をフィルタする機能が欠如していた。この問題は、合宿中に実装を追加することによって解決した。

- 運用に関するもの

  1. 他の経路制御プロトコルとの連係
     他の経路制御プロトコルとの連係がうまくとれていなかった。具体的には、経路情報のループが発生し、AS-External-LSA が増加し続けるという現象が発生した。この問題は、合宿中に解決されていない。

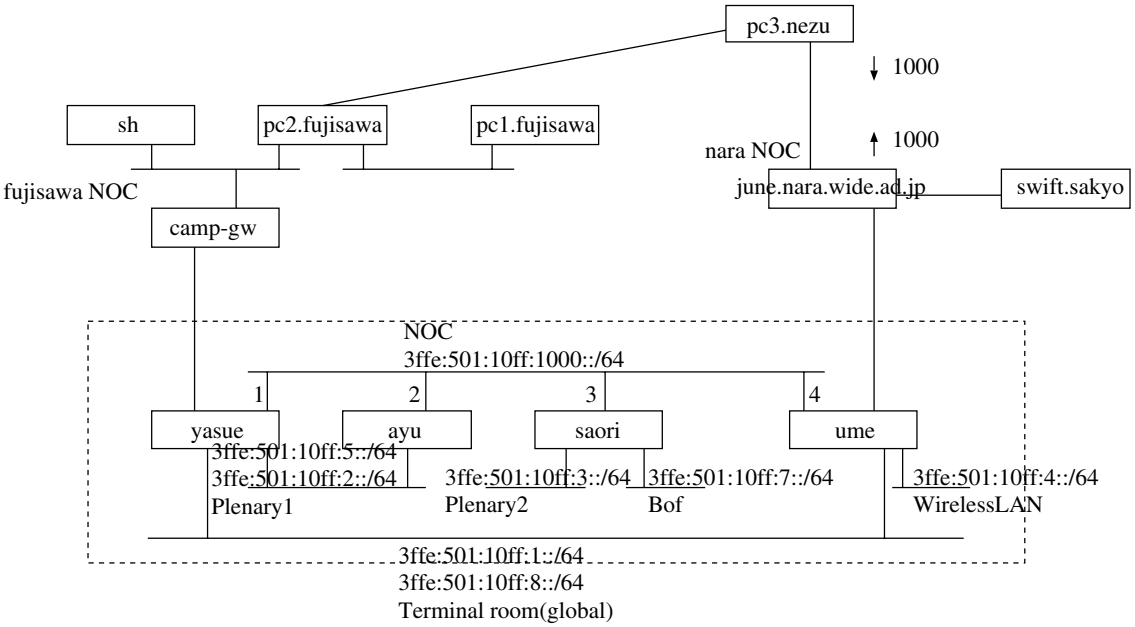


図 2.22 合宿時の WIDE IPv6 ネットワーク

2.5.3 まとめ

OSPFv3 で経路制御されるネットワーク内に存在する終点への経路は安定した。しかし、OSPFv3 で経路制御されるネットワーク外に存在する終点への経路は、合宿を通じて不安定なままであった。これは、他の経路制御プロトコルとの連係が的確になされないことが原因である。

OSPFv3 の実装である Zebra ospf6d には、実際に運用するうえで、様々な機能が欠如していることがわかった。

WIDE 6Bone を用いた OSPFv3 の実験運用を続けることにより、実装と運用手法の改善を続ける。

2.6 会議室インターネットライブ中継

本節では，2000 年春の WIDE 合宿において行った，会議室インターネットライブ中継実験について述べる．

2.6.1 実験の概要

今回の合宿ネットワークにおいて，会議室のインターネット中継を行った．この中継により，どの程度のアクセスが発生し，サーバの負荷が上昇するかの計測を行った．

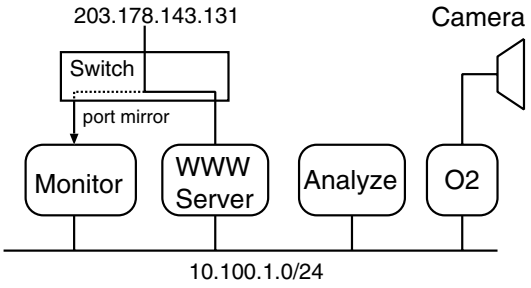


図 2.23 ネットワーク構成

2.6.2 実験環境

図 2.23 に今回の実験を行うためのネットワーク構成を示す．図 2.23 に示す O2 により，会議室内の画像を取り込み，WWW サーバへ画像を転送した．

到着したリクエストを計測するために，WWW サーバの公開アドレスをスイッチのポートミラーにより，モニタホストへ転送した．解析ホストは，リアルタイムに結果を出力するために設置した．

2.6.3 結果および考察

図 2.24 にインターネット中継のメインページの概観を示す．ページの左に中継されている会議室の画面が 4 つ 3 秒間隔で表示される．全体の画像更新間隔は 6 秒である．

今回の中継用 WWW サーバには，ほとんどアク

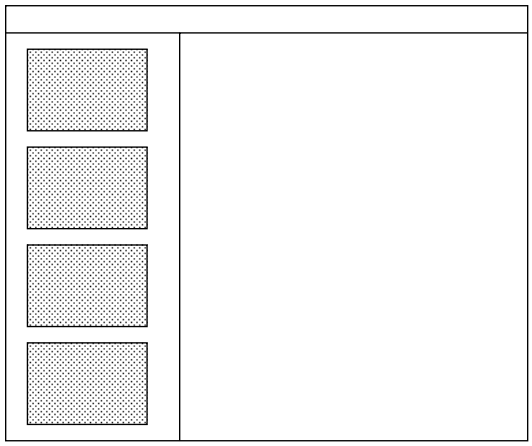
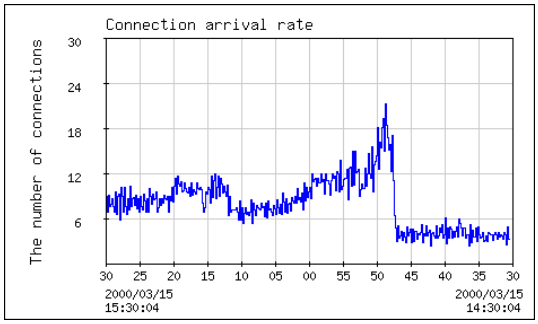


図 2.24　メインページ



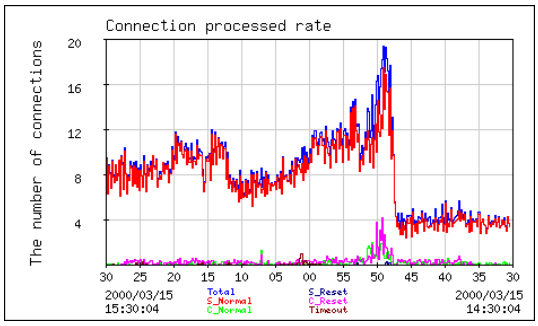


図 2.25　計測結果

セスは集中しなかった．最もアクセスの集中した時間帯に置いても，1 秒間に約 20 リクエストであった (図 2.25)．

WIDE 合宿におけるアンケートの結果から，

- コンテンツがわかりにくい．
- 無意味なリロードが多い．

と言った回答が得られた．まず，「コンテンツがわかりにくい」という回答については，「公開している画像の大きさが小さかった」という事が考えられる．これは，画像内部のスライドの文字が読めないほど小さかったからであると考えられる．次に，「無意味なリロードが多い」という回答については，「公開している画像にほとんど動きがなかった」という事が考えられる．会議は野球などと異なり，画像 (スライド) の動きがほとんどない．したがって，頻繁にリロードを行ったとしても，前のスライドと同様の場合が多い．これにより，ユーザは同じ画面を何度もリロードしていると感じたのではないかと考えられる．

今回の中継では，会議という性質上，

- コンテンツの動きが少ない．
- 中継画面が VGA プロジェクタで固定していたため，OHP を使うと何をしているのかわからなくなる．
- 発表者の声がないため，何を言っているのかわからない．

と言った事が考えられる．これを解決するためには，

- ログをリアルタイムに入力する．
- 発表を VGA プロジェクタのみで行い，それを直接中継サーバに入力する．
- 無意味なリロードを避け，スライドが切り替わった時に，画面の更新を行う．

といった事が考えられる．

2.7 DNS ダイナミックアップデートによる名前の登録

2.7.1 目的

DNS は計算機の IP アドレスと、その計算機につけられた名前の対応をとるためのシステムである。特に、IPv6 ではアドレス空間が大きくなっていることから、名前と IP アドレスの対応は人間の記憶力を遥かに凌駕しており、DNS の重要性は大きくなっている。

IP アドレスから名前を求めることを逆引きと言う。IP アドレスからの逆引きが可能であるということは、そのアドレスが DNS に登録されており、ある程度の管理がされていると判断することも可能で、

これを簡単な認証機構として用いられている場合がある。例えば、一部の FTP サイト等には IP アドレスの逆引きが存在しない場合には接続を受け付けないこともある。

WIDE 合宿において提供されるネットワークは IPv4 と IPv6 の両方を利用している。DHCP で配布される IPv4 アドレスに関しては、あらかじめ配布するアドレスの逆引きを DNS に登録しておく。しかし、ステートレス自動設定 [224] される IPv6 アドレスに関しては、予測不能である、だからと言って、最初から 2^{64} 個のアドレスを DNS に登録しておくことは現実的ではない。そこで、IPv6 のアドレスの逆引きをユーザ自身が登録できる仕組みを提供することでこの問題を解決させることが出来ると考えた。

2.7.2 実験

実験では、Web のインターフェースから、ユーザに登録を希望する名前と IPv6 アドレスを登録させ、DNS ダイナミックアップデート [133] による名前の逆引きを登録を行なう。

1. 名前空間の集約
   合宿ネットワークはいくつかのサブネットで構成されており、しかもそれぞれのネットワークが複数のネットワークプレフィクスを持っている。このため、単純に名前の逆引きを登録していくと、複数のプレフィクスを登録する必要が生じ、新たなプレフィクスが増加したときや、リナンバの際にはその全てを登録しなおすことが必要になってしまう。
   そこで、DNAME Resource Record [225] という新しい仕組みを使って、逆引きのゾーンを単一ゾーンに集約し、登録の仕組みを単純にした。具体的には、全ての逆引きのゾーンを v6.camp.wide.ad.jp というゾーンに集約し、プレフィクスに関わらず、アドレスの下位 64 ビットだけを見て名前を解決することにした。この DNAME の機構を利用するため、最新の BIND version 9 beta1 を利用した。
2. 実験ネットワークの構成
   BIND9 の動作がまだ不安定だったため、DNS は 2 台用意することにした。1 台目は通常のサービスを行なうための DNS で、安定性を重視し、BIND8 に KAME Project のパッチを当てて、v6 ready にしたものを利用する。2 台目は BIND9 で構成され、v6 アドレスの逆引きゾーンの Primary Master として動かす。ユーザが参照するのは常に 1 台目の BIND8 ベースの DNS にする。
   Web サーバから CGI で専用プログラムを呼出し、2 台目の DNS に対してダイナミックアップデートをかける。2 台目の DNS は、アップデートメッセージの始点アドレスが Web サーバだった場合のみアップデートを受け付ける設定にした。
3. サービスの提供
   登録のための Web ページを用意し、CGI によるデータの入力を行なわせる。簡単な認証が必要だと考えたことと、登録する数に制限を持たせたかったことから、Diffserv の実験チームが利用していた課金及び認証のシステムを利用させてもらうことにした。これにより、認証したユーザの情報や、登録された名前を関係データベースで管理することができるようになった。

2.7.3 結果と課題

システムは安定して動作し、不慮のエラーなどは出なかった。しかしながら、いくつかの細かな問題はあった。

- 登録インターフェースとなる Web ページへのリンクが切れていた時間帯があったため、その時間帯に利用を試みたユーザは登録ができなかった。
- アップデートがかかってから、それが実際に反映されるまでの時間が、BIND8 における通常のアップデートと比べて非常に遅く、そのことをユーザ側に通知していなかったため、登録されなかったのではないかと思われることも多かった。

最終的に、このシステムで IPv6 の名前の逆引きを登録した人数は 27 人だった。あまりよい結果とはいえないが、IPv6 ネットワークの、対外接続は不安定な時間が多く、逆引きの登録が意味をなさない場合が多かったことも利用人数の少なさの原因かもしれない。

今後、同様なサービスを行なう場合に徹底することは、まずユーザインターフェースとなる Web などの部分は何度も動作を確認することが大切であると思われる。リンクが切れるといった、初歩的なミスでサービス全体に悪影響を与える結果を招いてしまうことは避けたい。そして、ユーザへのシステムの利用方法の通知を徹底することの重要性も痛感した。具体的にどのようなシステムでどのような操作をすると、どのような効果があるのかを事前に通知しておけば、もっと利用者は増えたかもしれない。

今回の実験を総括すると、システム的には不満もなく、正しい動作をさせることができたが、ユーザへの配慮が足りなかったと思う。

2.8 無線 LAN 環境の運用

2.8.1 目的

合宿期間中における無線 LAN 環境の構築・運用を行なう。

2.8.2 概要

合宿期間中のネットワークサービスのひとつとして、無線 LAN 環境を構築・運用した。

2000 年春合宿では、それ以前の合宿で用いられてきた NetWave のアクセスポイントのほかに WaveLAN など IEEE 802.11 対応のアクセスポイントも導入し、ひとりでも多くの合宿参加者がネットワークに接続できるよう試みた。

2.8.3 環境

NetWave,WaveLAN 各アクセスポイントを、プレナリルーム、ターミナルルーム、BOF 部屋にそれぞれ 1,1,2 の数で設置した。BOF 部屋については大部屋をパーティーションで 3 部屋に区切ってあったため、BOF 部屋全体をカバーするよう、両端の部屋にアクセスポイントを設置した。

ネットワークサービスについては、無線 LAN 専用のセグメントを 1 つ用意し、DHCP による IP アドレスの取得など、有線と同様のサービスが得られるようにした。

2.8.4 結果

無線 LAN 環境の運用中、DHCP による IP アドレスのリース時間が長かったためか、クライアントによっては IP アドレスが取得できないことが時々生じていたが、全体的には特に大きな支障が出ず、安定した運用ができた。

2000 年春合宿では、無線 LAN 環境の利用状況を把握するため、合宿終了間近に行なわれる合宿に関するアンケートの中に無線 LAN 環境の利用について質問した。その結果、合宿参加者の約 2 割が無線 LAN 環境を利用していたことがわかった。

これは、合宿開始前に無線 LAN 環境が利用できることのアナウンスが弱かったこと、WaveLAN による無線 LAN 環境の運用が遅れたこと、日本製の NIC によっては使えなかったものがあったなどから、無線 LAN 環境の利用が少なかったと思われる。

しかしながら、利用者の中では、ケーブルレスで利用できて良かった、ローミングできるようになっていたため便利だったなど、無線 LAN 環境は好評であった。

2.8.5 まとめ

現在、802.11 対応無線 LAN カードが容易に手に入れることができるため、2000 年秋以降の合宿でも無線 LAN 環境を構築・運用することは必須であると思われる。そのためには、1 つのアクセスポイントに対して、同時にいくつまでクライアントが接続できるか、どこにアクセスポイントを配置するか、DHCP による IP アドレスのリース時間をどうするかなど、効率良く運用するためのノウハウを蓄積していく必要があると思われる。

2.9 衛星

2.9.1 衛星回線運用技術の蓄積

衛星回線は、地上回線にない様々な特性を持つ。WIDE 合宿では、回線設定の柔軟性や地理不偏性などを活かし、衛星回線がインターネットへの接続性を提供してきた。

地球局設備としては超小型地球局 VSAT(Very Small Aperture Terminal) を用いている。VSAT はパラボラアンテナ、屋外装置 ODU(Out-Door Unit)、屋内装置 IDU(In-Door Unit) から構成される。データの通信速度は最大 2Mbps である。データ送受信のため、ルータと IDU はシリアルケーブルで接続す

る。衛星ルータは、高速シリアルカード Wanic400 を装着した PC である。これを図 2.26 に示す。

最大 2Mbps の通信速度は電話回線や ISDN 回線を用いたインターネット接続よりも高速である。そのため 99 年度秋の WIDE 合宿では、合宿期間中これを用いてキャンプネットを WIDE バックボーンへ接続した。しかし、2000 年度春の WIDE 合宿では WIDE バックボーンへの接続に高速な地上回線を用いた。そのため、衛星回線は運用技術の集積を目的に設置された。衛星回線を災害時のバックボーンとして利用したり、必要な時に即座に設置するためには、このように衛星回線の運用技術を常に集積し続ける必要がある。これまでの VSAT を用いた運用経験は、近々文書化し公開する予定である。

2.9.2 UDLR 実験

WISH-WG は、片方向の衛星を含むネットワークにおける経路制御の手法として、片方向の衛星回線上に仮想的ブロードキャストリンクを構築することを考案した。これは Internet-Draft(draft-ietf-udlr-lltunnel-03.txt) が発表され、標準化が進められている。2000 年度春の WIDE 合宿では、慶応大学湘南藤沢キャンパスの実装を用いてこの手法の有効性を検証した。

片方向の衛星回線を含むネットワークでは、経路制御上、次の問題が生じる。

- 衛星回線の受信局から送信局へ経路情報が流れない
- 衛星回線を用いて、受信局と送信局が双方向の通信を行なえない

上記の問題は、経路制御プロトコルが通信路の双方向性を前提に設計されているために生じる。また、下記の問題は、インターネットにおいて隣接するノード同士が通信する場合、それらを直接接続している通信路を用いるために生じる。

仮想ブロードキャストリンクを用いる手法は、これらの問題を次のように解決する。

1. 受信局が衛星回線に送信しようとするデータリンクパケットを、別の IP パケットにカプセル化し、送信局の双方向通信路の IP アドレス宛に送信する。(トンネリング)
2. 送信局では、受信局から受けとったパケットを脱カプセル化し、取り出したデータリンクパケットをそのまま衛星回線に送信する。(ブロードキャストエミュレーション)
3. 以上により、片方向の衛星回線上に仮想的なブロードキャストリンクを構築し、上位層からは通常のブロードキャストネットワークに接続しているように見せる。

実験トポロジを図 2.27 に示す。

受信局は Camp-net に接続し、送信局は WIDE バックボーンに接続した。衛星ネットワークにはプライベート IP アドレス (10.0.0.0/24) を割り振った。受信局から送信局の地上回線インタフェース宛にトンネルを設定した。

また、今回の実験ではデータリンクプロトコルとして”JC” を用いた。“JC” はシリアルインタフェースを用いてマルチアクセスネットワークを構築する。“JC” では、Ethernet と同じデータリンクヘッダを用いる。宛先データリンクアドレスはネクストホップ IP アドレスである。送信元データリンクアドレスは、そのパケットを送信したインタフェースの IP アドレスである。

この結果、受信局 (10.0.0.2) から送信局 (10.0.0.1) への到達性が確認された。また、受信局からのブロードキャストパケットや受信局から 10.0.0.0/24 の他のノードへのユニキャストパケットが衛星回線上に流れていることを tcpdump により確認した。

2.10 Server の監視

2000 年 3 月の合宿では、サービスの安定をはかるため、各サーバーのプロセスを監視し、必要に応じて再立ち上げまで行なうような機構を導入した。この節では、サーバー内のプロセスの監視と、その実装時に考慮した部分について述べる。

2.10.1 設計

設計段階では、以下に挙げる内容を考慮した。

- プロセスの状態を監視すること
- 機能しないプロセスが発生したら再立ち上げを行なうこと
- 再立ち上げをしてもプロセスが立ち上がらない場合には、管理者に連絡すること

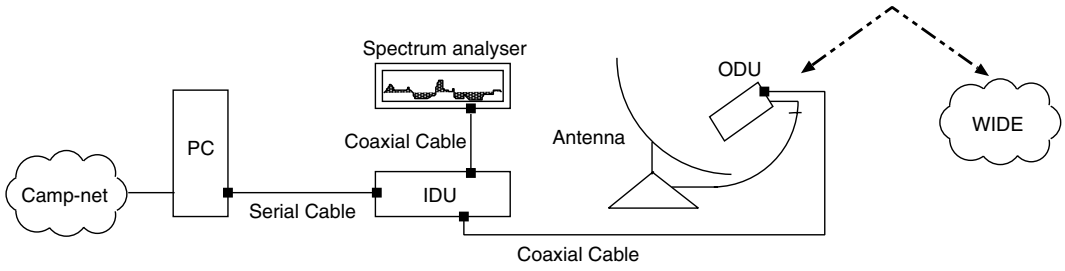


図 2.26　VSAT 構成図

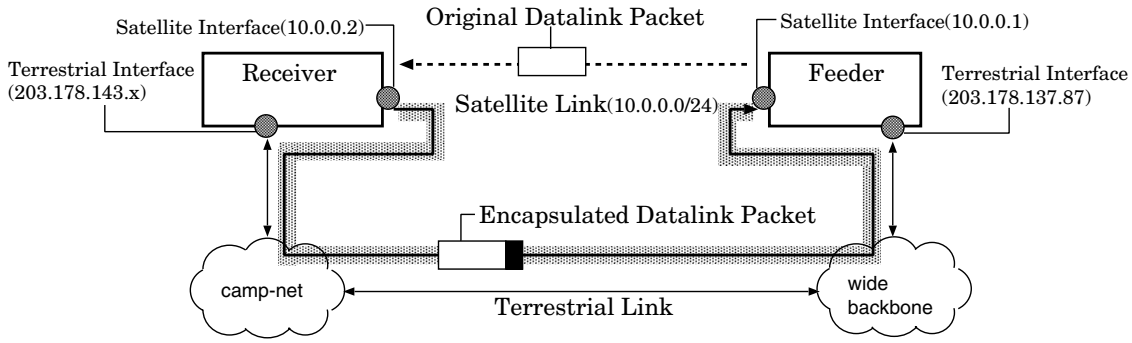


図 2.27　UDLR 実験トポロジ図

今回の設計では、サービスの停止状態を以下のように定義した。

1. プロセスが process table に無い
2. プロセスが process table 上にあるにも関わらず、サービスが受けられない

process table の確認には ps コマンドを利用し、その出力を確認すると言う方法を取った。この方法で、サービスを提供するサーバープログラムが process table 上にないという状態は検出できる。しかし、process table 上にサーバープロセスがありながらサービスが受けられないと言ういわゆるハングアップの検出は難しい。サービスが受けられないという状態になる原因として、

- プロセスがハングアップしている
- 設定が間違っているため、サービスを拒否している

ということが想定できるが、設定間違いは、自動的に検出することが困難なためである。プロセスのハングアップだけなら、サービスを受けてみて返事があるかどうかを検出すれば良いと言うことになる。そこで今回の実験では「設定は正しい」と言うことを前提とした。

2.10.2 実装

実装には、今後さまざまな場所で利用できるように perl, ps, sendmail のみを利用している。sendmail は、管理者への連絡のために用いているので、実際の監視作業は全て perl と ps で行なった。

基本的には、

1. 定時に (例えば 5 分毎とか 10 分毎など) 監視スクリプトを起動する
2. 監視スクリプトは process table を確認する
3. process table にサービスプロセスが無い場合、再起動する
4. サービスプロセスにサービスの要求を行なう
5. 正しくサービスを行なっている場合、終了
6. サービスが行なわれていない場合、再起動する
7. 3 回再起動してどうしても立ち上がらない場合、管理者にメールで連絡する

という作業を繰り返すように実装した。

2.10.3 運用

合宿では、重要なサービスをしているサーバーに

対して監視システムを仕掛けておいた。合宿中において、再起動に失敗して管理者に連絡が行くという状況は発生しなかったが、サービスが停止したためプロセスを再起動するという状態は数回あった。

実際にこの監視システムを導入することで、サービス停止時間は非常に少なくできたと思う。

2.10.4 課題

今回の実装では、監視できるサービスの種類が多かったとはいえない。今後は、監視できるサービスの充実と、さらなる設定の容易さをはかっていくつもりである。

問題点としては、この手の監視システムがサービスの監視のために実施する確認が、DoS に類する攻撃の手法とほぼ変わらないことが挙げられる。従って、公開や運用には最新の注意を払う必要があると言える。

2.11 トラブルチケットシステム

2.11.1 目的

WIDE 合宿で利用されるネットワークは、二つの目的がある。一つは、合宿参加者へのインターネットへの接続性の提供である。もう一つは、実験ネットワークとしての合宿ネットワークであり、様々な運用実験である。この二つの目的は、ネットワーク運用に対する安定性に対して異なるポリシーが存在する。

このため、合宿ネットワークの運用時はユーザへのサービスの提供時に様々なトラブルが発生する。また、ユーザにとってトラブルと考えられる状況が実験担当者による意図的な状況の場合もある。

トラブルチケットシステムは、ネットワークの状況を利用者からの情報を運用者へ速やかに流し、合宿で発生するトラブルの効率的な解決を支援する仕組みを構築することを目的とする。

2.11.2 システム概要

ユーザインターフェースは、http での web form を用いる。Web を用いることにより、トラブルに関する情報を広く共有することができる。

ユーザは、状況をカテゴライズし、トラブルの状況を web 経由でトラブルチケットシステムに伝えチケットを発行する。ユーザの入力する項目を表 2.5 に示す。

表 2.5 入力項目

項目	選択肢・内容
Type	Trouble
	Information
	FAQ
Priority	Urgent
	High Priority
	Middle Priority
	Low Priority
	Dust
Category	作成されたカテゴリ
記述欄	Subject
	トラブルの詳細

トラブルの内容によってチケットをカテゴライズは、解決作業の分担を明確化する。カテゴリは実験を行う実験担当者がカテゴリを作成し、解決作業を行うメンバーなどを指定する。解決作業を行うメンバーは発行されたチケットに対応し、その対応内容の情報をチケットに付加する。また、チケットは状況に応じ作業担当者が作業担当者の変更 (Transfer)、作業終了 (Close) 等のチケットのステータスを変更する。これにより発行されたチケットは、トラブルの発生から解決までの過程を記録する。

システムのユーザインターフェースは、ユーザ登録とパスワード認証を行う。これにより、ユーザごとに関係するチケットを自動的に表示することが可能となる。

2.11.3 実装と運用

情報を保持するシステムのバックエンドは PostgreSQL を用いた RDBMS を利用する。ユーザインターフェースは、perl スクリプトでの CGI を用いている。

2000 春合宿でのトラブルチケットシステムが稼働するホストは合宿ネットワークの 7 つの各セグメントの全てに接続するインターフェースを持つ。これによって、各セグメント上のユーザからのトラブルチケットシステムまでの接続性はローカルセグメントでの 2 層以下でのトラブルでない限り接続性が維持される。

2.11.4 運用結果

運用時に置いてはいくつかの問題が発生した。

各セグメントのユーザは、ローカルセグメントでの接続性が維持される限り、トラブルチケットシステムを利用することが可能である。しかしながら、提供する Web Server に対する DNS の名前解決は、各セグメントからそのセグメント毎の異なるアドレスで名前解決をする必要がある。また、チケットの発行はユーザに対する啓蒙が不十分であり、一部のユーザからの利用に限定され少なかった。

2.12 合宿ネットワークのトラヒック情報収集・公開

2.12.1 実験目的

広範囲・広帯域・マルチプロトコル (IPv4/v6) 環境でのネットワーク情報収集とその公開、およびそれらの手法の確立

2.12.2 実験概要

ネットワーク内要所の観測点からネットワーク管理標準プロトコル SNMP により定期的にトラヒック情報を取得する。取得したデータは随時 Web サーバを通して公開する。また後日、そのデータを用いて統計処理等を行ない、その結果を公開する。

これらを行なうためのシステムとして、NetSkate というシステムを構築し使用した。

2.12.3 ネットワーク情報収集・公開システム (NetSkate) について

NetSkate は大きく分けて DataCollector モジュール (情報取得) と NetSkate モジュール (GUI) から構成され、ネットワーク情報を取得し、ネットワークのユーザにわかり易い形にして公開するためのシステムである。

DataCollector モジュールは Perl で記述され、SNMP を用いてデータを取得・蓄積する。

NetSkate モジュールは Java 2 で記述され、以下に挙げる GUI ベースでプラットフォーム非依存のデータブラウジング環境を提供する。

ネットワークマップを図 2.28 に示す。ただの「絵」ではなく、ネットワーク情報を持ち随時アップデートされる。

トラヒック量を図 2.29 に示す。取得したデータをグラフにして表示する。データの形式は「<タイムスタンプ> <カウンタ>」の対であり、SNMP により取得される。

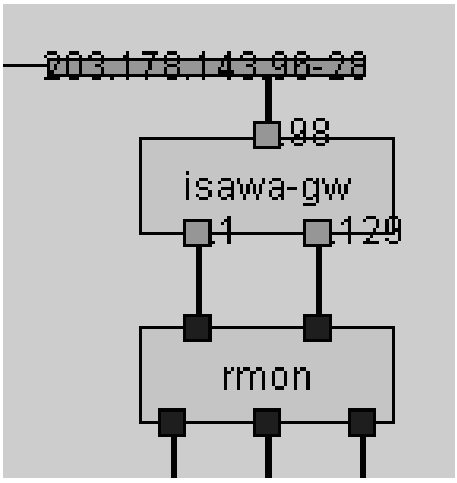


図 2.30　インターフェース監視

インターフェース監視の様子を図 2.30 に示す。正常動作しているインタフェースが緑、ダウンしているものは赤、監視対象外のものは青でそれぞれマップ上に表示される。SNMP を用いて MIB-II の ifDescr / ifOperStatus を監視することにより実現されている。

図 2.31　アプリケーション監視

アプリケーション監視の様子を図 2.31 に示す。監視対象のプロセスが存在すれば緑色、存在しなければ赤色で示す。監視対象は SNMP エージェントの設定ファイルにより指定し、その情報を SNMP を用いて取得する。

リソース監視の様子を図 2.32 に示す。監視対象のリソースが正常な状態ならば緑色、危険であれば赤

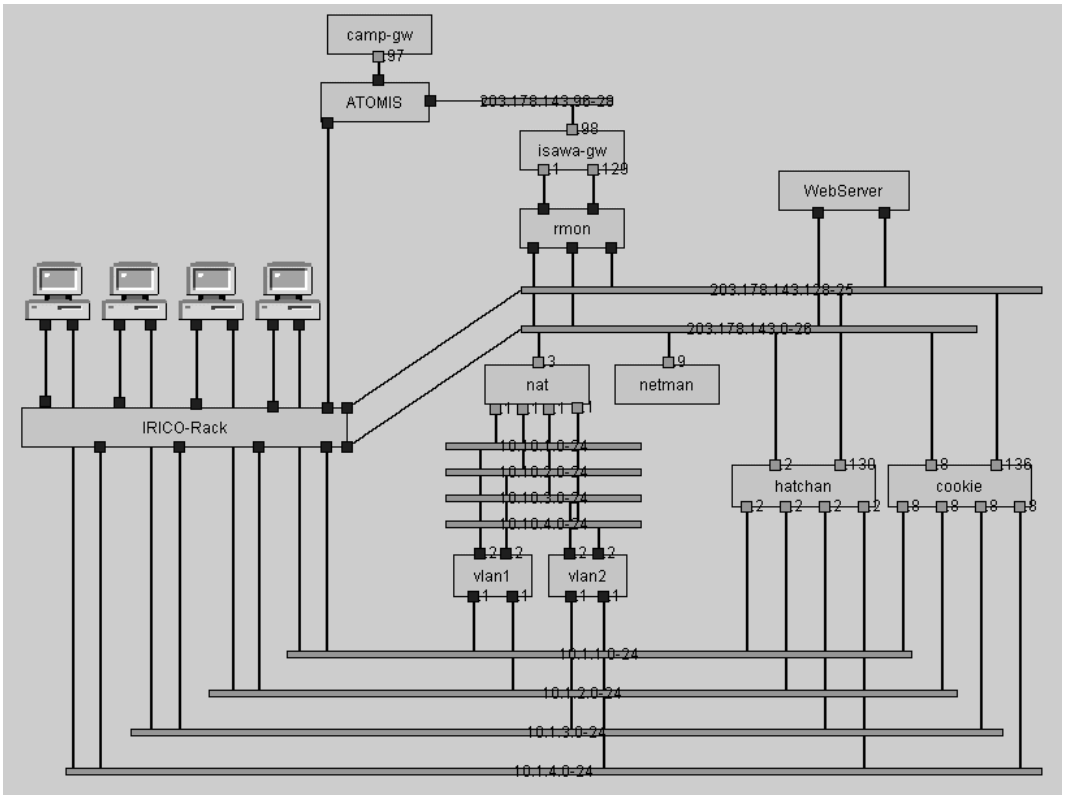


図 2.28 ネットワークマップ

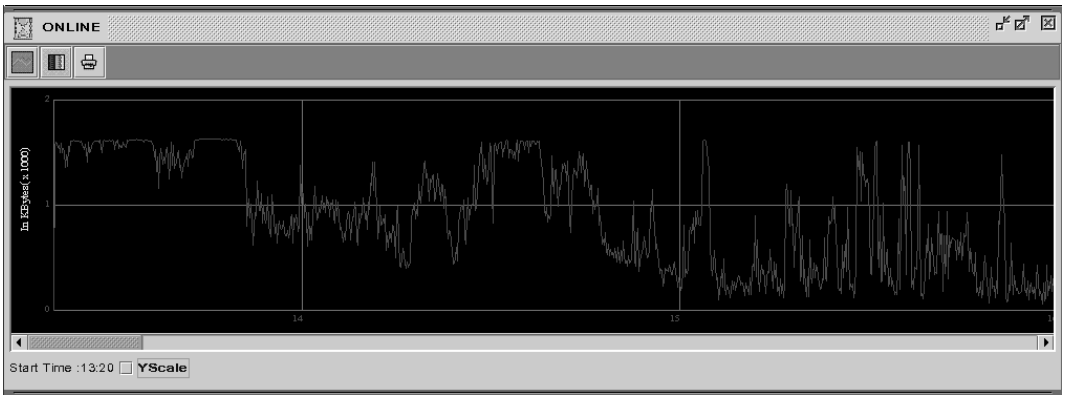


図 2.29 トラヒック量

色で示す。監視対象は SNMP エージェントの設定ファイルにより指定し、その情報を SNMP を用いて取得する。

2.12.4 実験環境

今回の実験のために cookie /netman という二台のマシンを用意した。cookie 上で DataCollector を動作させ、netman 上で NetSkate を動作させた。NFS を用いデータの受け渡しを行なった。

監視対象マシンには SNMP エージェントとして UCD-SNMP 4.1.1 + IPv6 patch を導入し、さらにアプリケーション監視やリソース監視のための設定を行なった。

2.12.5 実験結果

- DataCollector 運用結果

図 2.32　リソース監視

期間:	2000/3/10 - 3/11
	2000/3/14 - 3/17
総データサイズ:	21221 KBytes
総データ数:	2487 個

- NetSkate 運用結果
  以下の URL で合宿期間中どこからでもアクセスできた。
  `<URL:http://netman.camp.wide.ad.jp/~netskate/>`

2.12.6 解析結果

グラフは付録を参照のこと。

どの観測点に於いても上りと下りのトラヒック量には差があることがあるが，パケット数はほぼ同数であることがわかる。このことから逆に上りと下りのパケット数に大きな差がある時は何らかの障害が起きていてネットワークが不調であったのではないかと考えられる。また，nat を介したトラヒック量を見ると他のものに比べ圧倒的に下りのトラヒックの割合が高くなっていることが特徴的である。

2.12.7 問題点

データ取得に関して

- SNMP トラヒック量の測定をしなかったため、このシステムがネットワークに与える影響というものを測定できなかった。
- IPv6 に関係するマシン群への SNMP アクセスの設定をしなかったため、IPv6 上での SNMP アクセスを実験できなかった。
- データ取得プログラムを手動で起動することにしており、チェック機能を設けなかったため、16 日の 21 時から 17 日の 9 時までという長期間にわたりデータをとり損なうという事故が発生した。

データ公開に関して

- Java 2 の環境を持っていない人は全くこのシステムを利用できなかった。このため netman をターミナルルームに向け解放したが、効果が少なかった。
- 操作方法などの説明をする時間がなく、わかりにくい印象を与えた。

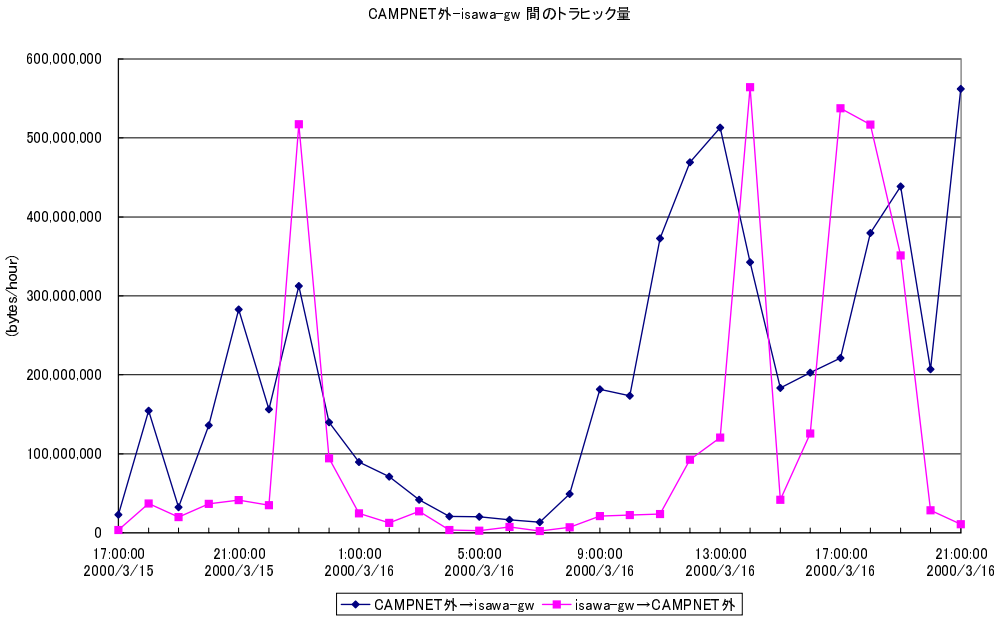
2.12.8 今後の課題

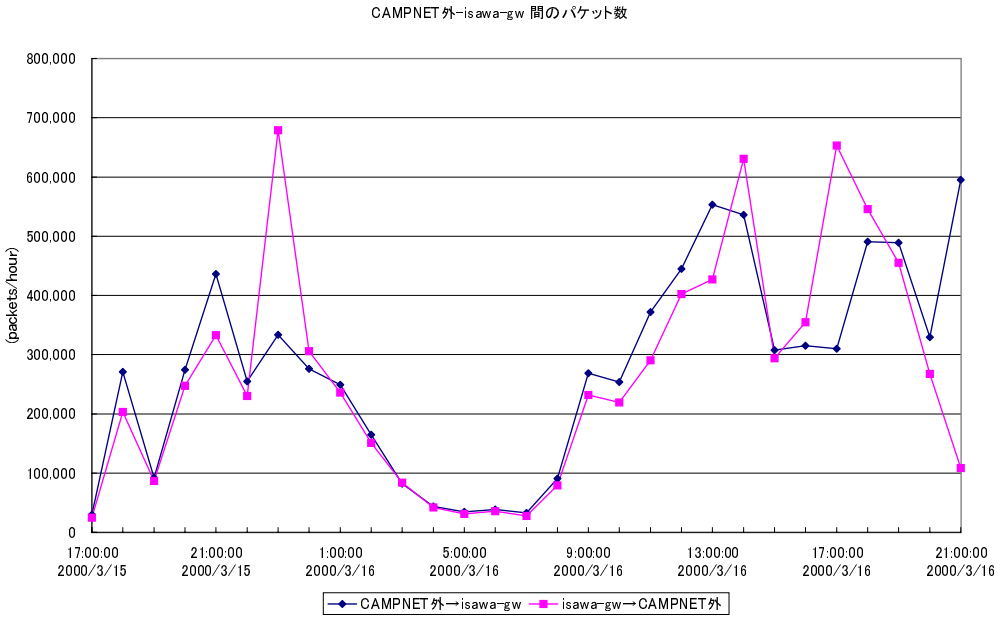
今後の課題として，

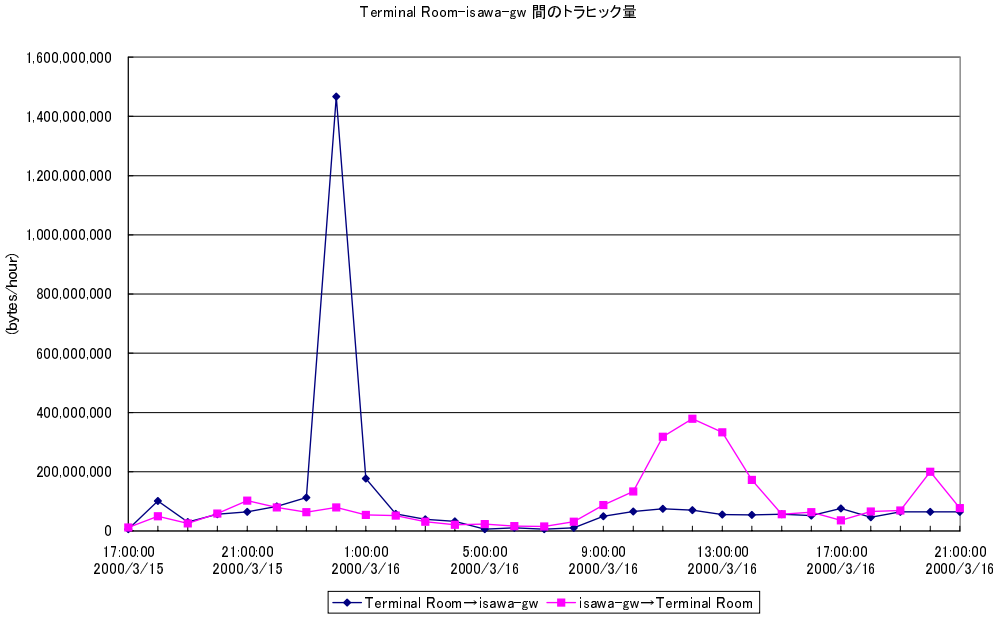
- RMON の活用やエラーの測定等今回の結果を補完する物を導入する。
- システムの与える影響の測定
- IPv6/SNMPv3 等への対応
- 公開インタフェースの改良

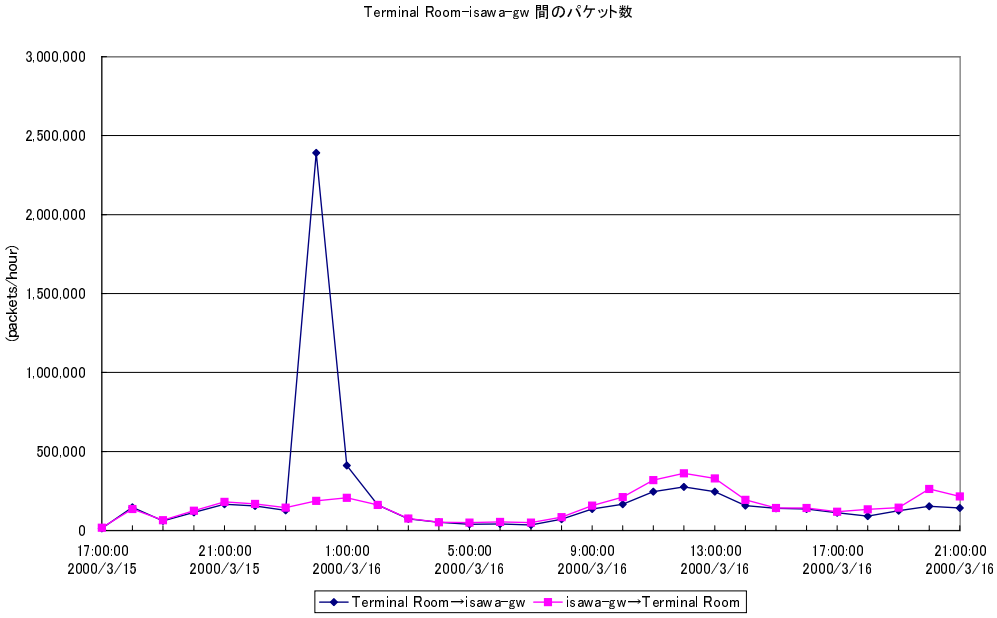
を考えていきたい。

2.12.9 付録データ

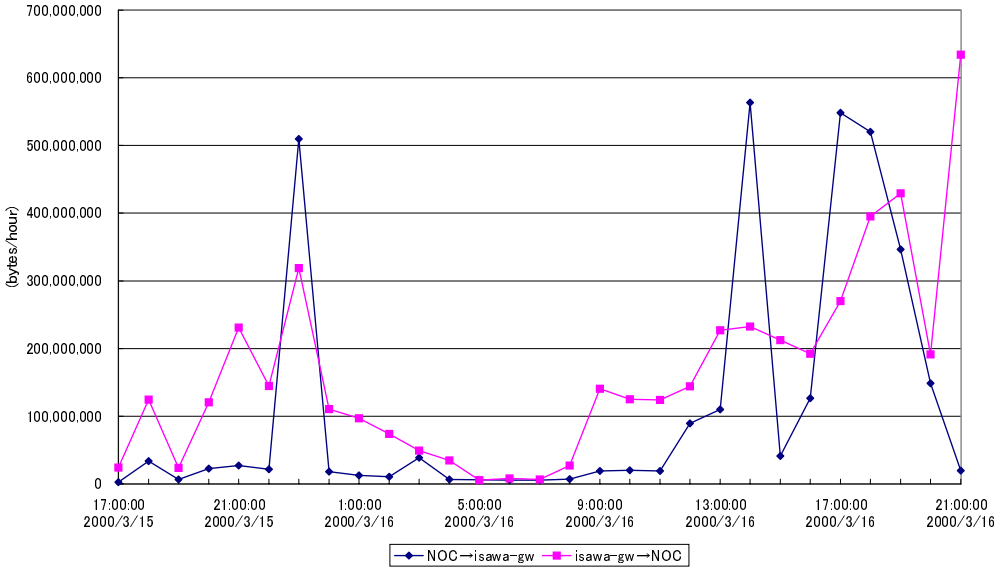

CAMPNET外−isawa−gw 間のトラヒック量
600,000,000
500,000,000
400,000,000
300,000,000
200,000,000
100,000,000
0
(bytes/hour)
17:00:00 2000/3/15
21:00:00 2000/3/15
1:00:00 2000/3/16
5:00:00 2000/3/16
9:00:00 2000/3/16
13:00:00 2000/3/16
17:00:00 2000/3/16
21:00:00 2000/3/16
CAMPNET外→isawa−gw
isawa−gw→CAMPNET外


CAMPNET外−isawa−gw 間のパケット数
800,000
700,000
600,000
500,000
400,000
300,000
200,000
100,000
0
(packets/hour)
17:00:00 2000/3/15
21:00:00 2000/3/15
1:00:00 2000/3/16
5:00:00 2000/3/16
9:00:00 2000/3/16
13:00:00 2000/3/16
17:00:00 2000/3/16
21:00:00 2000/3/16
CAMPNET外→isawa−gw
isawa−gw→CAMPNET外

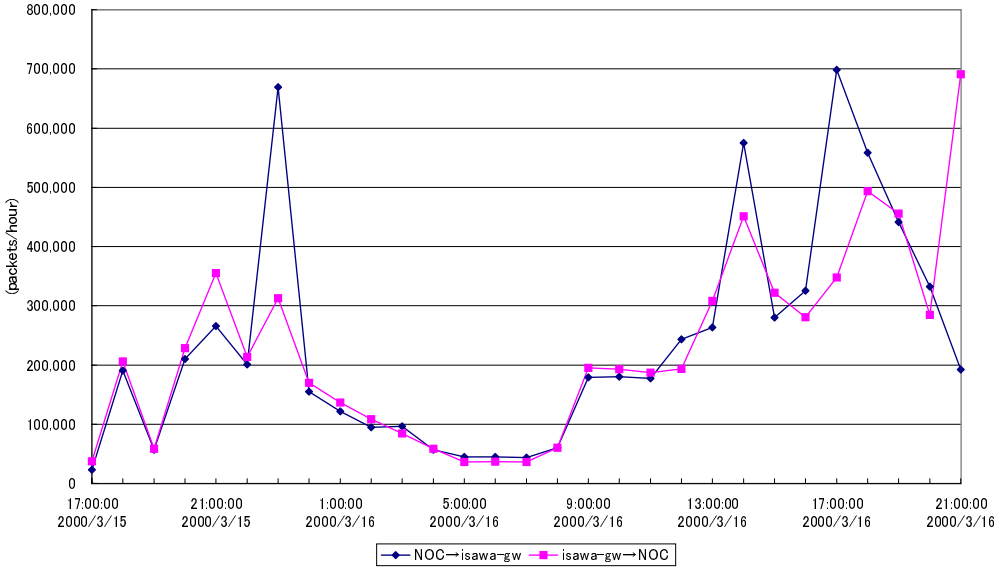

Terminal Room-isawa-gw 間のトラヒック量
1,600,000,000
1,400,000,000
1,200,000,000
1,000,000,000
800,000,000
600,000,000
400,000,000
200,000,000
0
(bytes/hour)
17:00:00 2000/3/15
21:00:00 2000/3/15
1:00:00 2000/3/16
5:00:00 2000/3/16
9:00:00 2000/3/16
13:00:00 2000/3/16
17:00:00 2000/3/16
21:00:00 2000/3/16
Terminal Room→isawa-gw
isawa-gw→Terminal Room


Terminal Room-isawa-gw 間のパケット数
3,000,000
2,500,000
2,000,000
1,500,000
1,000,000
500,000
0
(packets/hour)
17:00:00 2000/3/15
21:00:00 2000/3/15
1:00:00 2000/3/16
5:00:00 2000/3/16
9:00:00 2000/3/16
13:00:00 2000/3/16
17:00:00 2000/3/16
21:00:00 2000/3/16
Terminal Room→isawa-gw
isawa-gw→Terminal Room

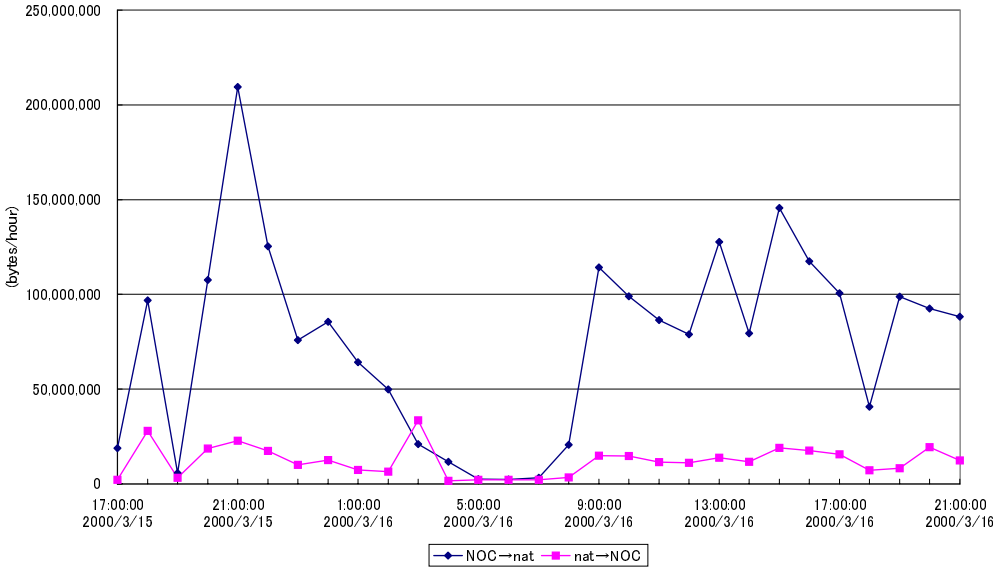
NOC−isawa−gw 間のトラヒック量



NOC−isawa−gw 間のパケット数



NOC−nat 間のトラヒック量



NOC−nat 間のパケット数