高品質インターネット　3－2
(2002. 6. 26)

汎用的なパラメータフィルタの設計と実装

垣内　正年† 森島　直人† 宇多　仁†† 中村　豊† 砂原　秀樹†

† 奈良先端科学技術大学院大学　〒630-0101 奈良県生駒市高山町 8916-5
†† 北陸先端科学技術大学院大学　〒923-1292 石川県能美郡辰口町旭台 1-1
E-mail: †{masato-k,naoto-m}@is.aist-nara.ac.jp, ††zin@jaist.ac.jp, †††yutaka-n@itc.aist-nara.ac.jp, ††††suna@wide.ad.jp

あらまし　通信を制御するには，対象となる通信のフローを識別し，その通信に対する操作を決定しなければならない．例えば Diffserv において，DS ドメインをまたがる End-to-End での通信では，統一的な通信ポリシの管理が要求される．このような複数のパラメータから構成されるポリシを End-to-End 通信に適用するような研究はこれまで行われていない．本研究では，複数のパラメータから構成されるポリシから特定のルールを通信に適用する機構を設計した．本機構の有効性を示すために，MPLS のラベル決定機構に実装した．

キーワード　Diffserv，MPLS，パラメータフィルタ，フロー分類

Design and Implementation of Universal Parameter Filter

Masatoshi KAKIUCHI†, Naoto MORISHIMA†, Satoshi UDA††, Yutaka NAKAMURA†, and Hideki SUNAHARA†

† Nara Institute of Science and Technology　8916-5 Takayama-cho, Ikoma, Nara, 630-0101 Japan
†† Japan Advanced Institute of Science and Technology　1-1 Asahidai, Tatsunokuchi, Nomi, Ishikawa 923-1292 Japan
E-mail: †{masato-k,naoto-m}@is.aist-nara.ac.jp, ††zin@jaist.ac.jp, †††yutaka-n@itc.aist-nara.ac.jp, ††††suna@wide.ad.jp

Abstract　For communication control, we distinguish a target flow of communication and must decide an operation that. For example, in Diffserv technology, we require unification policy management of communication in end-to-end communication extend DS-Domain. In past, a method applying a communication policy consists of many parameters have not researched. In this paper, we designed mechanism for communication to use specific rule based on the policy consists many parameters. We implemented our proposal in the MPLS label decision mechanism to show our validity.

Key words　Diffserv, MPLS, parameter filter, flow classification

1. 背　景

情報基盤としてインターネットが普及し，マルチメディアアプリケーションが広く利用されるようになった．従来のネットワークはベストエフォート型である．しかし，リアルタイム性を要求するネットワークアプリケーションでは，低遅延，低パケットロスといった通信品質が要求される．

Diffserv [1] では，DS ドメイン内のノードにおいて同一のポリシに基づいて IP フォワーディングを制御しなければならない．しかし，従来のインターネットは分散したノードが独立して協調動作するシステムとして設計・運用されている．広域ネットワークにおいてこれらの QoS 保証を実現するには，個々のネットワークノードによる独立分散処理では困難である．また，制御単位を複数ノードへ拡大し，統一した制御を行う必要がある．

Diffserv では，Policy Decision Point (PDP) がポリシを決定し，Common Open Policy Service (COPS) [2] 等のポリシ転送機構により各ルータにポリシを伝達する．ルータは，伝えられたポリシを解釈し，IP フォワーディングを行う．従来のルータでは，複数のポリシが存在した場合，ポリシの解釈方法は実装により異なる．そこで本研究では，複数のポリシを統一的に管理し，IP フォワーディング機構へ伝達する枠組みを設

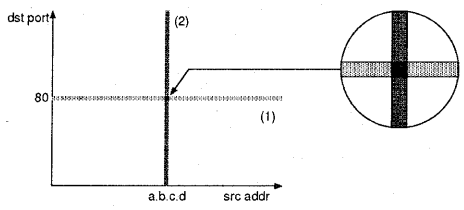

図 1 結果の競合 (1)
Fig. 1 competition of result (1)

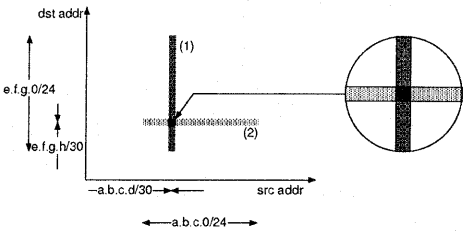

図 2 結果の競合 (2)
Fig. 2 competition of result (2)

計した.

本研究の有効性を示すために，WIDE プロジェクト研究会においてアドホックネットワークを構築し，実証実験を行った.

2. パラメータフィルタ

通信を制御する時，対象となる通信を識別し，その通信に対して適用する処理を決定しなければならない．例えば，Diffserv では制御パラメータに基づいて，トラフィックを IP ヘッダ，TCP ヘッダ，UDP ヘッダなどから分類し，IP データグラムのマーキング，シェーピング等の処理を行う．また，ファイアーウォールでは，フィルタルールに基づいて，サイトのポリシに適合しない通信を判別する．その結果に応じてブロック，ロギングを行う.

このような通信ポリシには，複数の制御パラメータがある．本研究では，あるトラフィックに対して制御パラメータを決定するシステムをパラメータフィルタ (parameter filter) と定義する.

多くの場合，フローに対応する操作を決定するために，インターネット層の始点アドレスと終点アドレス，および，トランスポート層の始点ポートと終点ポート等を利用する．しかし，これらを利用する順序や優先順位によって，結果が変わったり，結果が競合する.

例えば，透過型プロキシと特定ユーザの優先制御を，並行して利用する場合を考える．この場合，次の 2 つのルールが必要である.

(1) 終点ポートが 80 のフローはプロキシへ

(2) 始点アドレスが a.b.c.d のフローは専用線へ

図 1 に，(1), (2) のルールが適用されるフローを表す．縦軸に終点ポートを，横軸に始点アドレスを示す．網掛けの横線は (1) のルールが適用されるフローの集合，網掛けの縦線は (2) のルールが適用されるフローの集合である.

ここで，始点アドレスが a.b.c.d，終点ポートが 80 であるフローに着目する．このフローは図 1 の 2 つの線が交差する部分であり，どちらのルールを適用すれば良いか明確に決定できない.

次の例として，始点アドレスとそのプレフィックス長，および，終点アドレスとそのプレフィックス長で表される 2 つのルールを考える．入力となる IP データグラムに対して，始点アドレスと終点アドレスの最長一致を考える.

(1) 始点アドレスが a.b.c.d/30，終点アドレスが e.f.g.0/24 のフローはルータ A へ

(2) 始点アドレスが a.b.c.0/24，終点アドレスが e.f.g.h/30 のフローはルータ B へ

図 2 に，(1), (2) のルールが適用されるフローを表す．縦軸に終点アドレスを，横軸に始点アドレスを示す．網掛けの横線は (1) のルールが適用されるフローの集合，網掛けの縦線は (2) のルールが適用されるフローの集合である.

ここで，始点アドレスが a.b.c.d，終点アドレスが e.f.g.h であるフローに着目する．このフローは図 2 の 2 つの線が交差する部分である．始点アドレスの最長一致は (1) のルール，終点アドレスの最長一致は (2) のルールである．最長一致の条件だけでは，2 つのルールの内どちらを適用すれば良いか決定できない.

上記の 2 つの例では，複数のルールが競合するため，それぞれの条件だけでは一意に結果を決定できない.

このような問題は，以下のモデルに一般化できる.

- ある対象をキーとして
- ある分類スキーマに基づく分類レコード群から
- 適合レコード（群）を抽出する

ここで，対象，分類スキーマ，分類レコード，適合レコードとは，それぞれ以下の項目を表す．なお，“[”，“]”の中は，上記の 2 つの例に対応する.

対象 (object) 入力となるオブジェクト［IP データグラム］

分類スキーマ (classification schema) 対象が持つ属性のうち，分類レコード群の抽出に利用するもの［始点・終点アドレスおよび始点・終点ポート等］.

分類レコード (classification record) 制御パラメータ．分類スキーマに基づく，実際のパラメータを定めたインスタンスと，それに対応する適用操作集合の組［個々のルール］．通常は，複数のルールが存在し，分類レコード群を形成する.

適合レコード (adaptive record) 分類スキーマに基づいて抽出された分類レコード［結果となるルール］．抽出結果は，複数のレコードから構成される適合レコード群となる.

上記の例では，入力となる IP データグラムを元に適合レコード群が決定され，それに基づく操作が適用される.

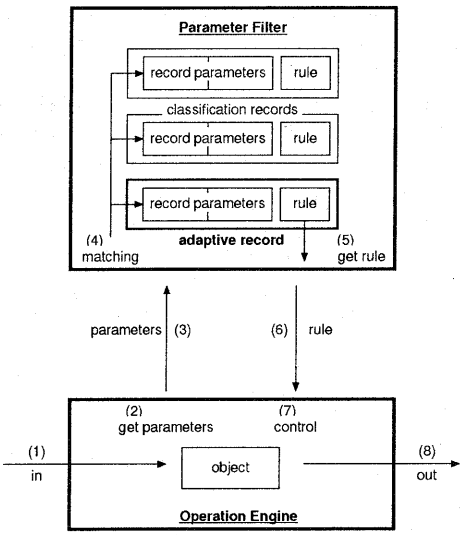

図 3 パラメータフィルタの動作
Fig. 3 parameter filter behavior

3. パラメータフィルタの設計

本章では，複数のパラメータで構成されるポリシを持つ制御パラメータ群から対象に適用するルールを決定するパラメータフィルタの設計について述べる．

3.1 パラメータフィルタの動作

ルータの IP 転送機構，Diffserv のキューイング，シェーピング機構などを操作機構 (operation engine) と呼ぶ．操作機構は，入力された対象を操作して，出力する．Diffserv においては，IP データグラムが入力され，キューイング，シェーピング処理を行い，出力する．キューイングの優先度，シェーピングの帯域などは，パラメータフィルタにより決定される．パラメータフィルタは，制御パラメータを分類レコードとして分類表に保持する．分類レコードは，レコードパラメータをキーとして，操作機構を制御するルールを持つ．

パラメータフィルタと操作機構の動作を以下に示す．(図 3)

(1) 操作機構に制御の対象が入力される．

(2) 対象から分類スキーマに基づいてパラメータ群を取り出す．

(3) (2) で取り出したパラメータ群をパラメータフィルタに送る．

(4) 操作機構から受け取ったパラメータ群と分類レコードを比較し，適合レコードを抽出する．

(5) 適合レコードからルールを取り出す．

(6) (5) で取り出したルールを操作機構に送る．

(7) パラメータフィルタから受け取ったルールに基づいて，対象を操作する．

(8) 操作した対象を出力する．

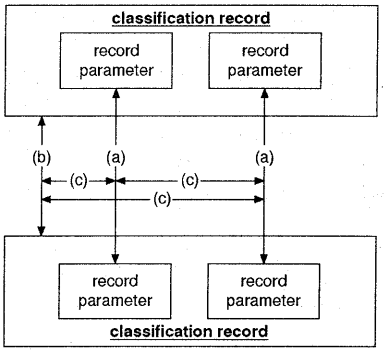

図 4 優先度と適用順序
Fig. 4 priority and order

3.2 競合の解決

始点・終点アドレス，始点・終点ポート等，複数のパラメータを持つ分類レコードは，互いの強弱関係が唯一に決まらない．多くのルータ実装は Access Control List (ACL) を用いて，リスト上位から順にレコードを比較し，最初もしくは最後に一致したレコードを検索することで，レコードを唯一に決める．また，別の実装では複数のパラメータ間に優先度をあらかじめ決めておくことで，レコードを唯一に決める．前者と後者の方法ではレコードの記述方法が異なる．ACL においても，ルータベンダ毎に実装が異なる．このため，異なるルータで同一の通信制御を行うことは困難である．また，ACL は先頭から順にレコードを検索するため，リストの順序によっては検索速度が不利になる．

競合する複数の制御パラメータから，対象に適用する操作を決定するためには，何らかのポリシが必要である．2. 章の透過型プロキシと特定ユーザの優先制御の問題は，「透過型プロキシは特定ユーザの優先制御より優先される」という規則があれば，競合が解決できる．この場合，始点アドレスが a.b.c.d，終点ポートが 80 であるフローは透過型プロキシを利用する．

また，始点アドレスと終点アドレスで最長一致するルールが異なる問題では，「終点アドレスの一致は始点アドレスの一致より先に適用される」いう規則で競合が解決できる．

これらは，以下の概念で表現できる．(図 4)

(a) 比較優先度 (comparision priority)

(b) 分類レコード優先度 (classification record priority)

(c) 適用順序 (application order)

比較優先度により，分類レコードの各パラメータに着目して，より限定的なパラメータを持つレコードを優先する．

分類レコード優先度は，「透過型プロキシは特定ユーザの優先制御より優先される」等の規則を表す．

適用順序は，「終点アドレスの一致は始点アドレスの一致より先に適用される」等の，比較の順序を示す．

4. KUPF の実装

本研究では 3. 章の設計に基づいたパラメータフィルタ，

KUMA's Universal Parameter Filter (KUPF) を実装した．

適合レコード群中の適合レコードは，分類スキーマに基づいて対象に一致する分類レコードの部分集合である．これに着目すると，分類レコード群から適合レコード（群）を抽出する手順は，次の2段階に分類できる．

第一段階 対象に一致する分類レコードをすべて選出

第二段階 第一段階の結果から，抽出ポリシにしたがって適合レコード（群）を抽出

KUPF では図5のように，前処理と，第一段階を行う第1ステージと，第二段階を行う第2ステージに分け，適合レコード（群）を抽出する．

前処理は，対象から分類スキーマにしたがって属性となるパラメータリストを取り出す．対象が IP データグラム，分類スキーマが始点・終点アドレスであれば，前処理の結果は2つのアドレスからなるパラメータリストである．第1ステージは，前処理の結果をキーとして，分類表 (classification table) の分類レコードから一致する分類レコードをすべて選出する．第2ステージは，選出した分類レコードから抽出ポリシにしたがって適合レコード（群）を抽出する．

前処理は，3.1節の (2) に該当する．第1ステージおよび第2ステージは，(4) に該当する．IP 層と TCP, UDP 層の情報によりフローを識別して，Diffserv による品質保証を行う場合，分類スキーマは始点・終点アドレスと始点・終点ポートである．パラメータリストは2つのアドレスと2つのポート番号から構成される．適用レコードにより，キューイング優先度，シェーピング帯域などが決まる．

KUPF は NetBSD 1.5.2 上で C 言語で実装した．KUPF はアプリケーションでの利用，カーネル内での利用が共に考えられる．ユーザランドで用いるライブラリを実装し，ライブラリをカーネル内へ移植した．メモリ管理には，『くまプロジェクト』(注1)の共有ライブラリ KA を利用した．このライブラリをカーネル内に移植することで，ユーザランド，カーネル両方でメモリ管理をほぼ同じ API で扱うことが可能となった．

以下で KUPF のデータ構造，処理内容を説明する．

4.1 パラメータ

対象を分類するために，対象から分類スキーマに基づいて属性を取り出す必要がある．分類レコードは，取り出した属性と比較するための値を持つ．これらの属性，値を表現するために，パラメータのリストを用いる．TCP/IP の場合は，始点・終点のアドレス・ポートの各パラメータからなるパラメータリストとなる．HTTP リクエスト の場合は，URL のパラメータからなるパラメータリストとなる．

4.1.1 構 造 体

パラメータを表現するために以下の構造体を定義した．

パラメータ型構造体 4.1.2 節で説明するパラメータ型 (parameter type) を表現する．パラメータ型ごとの同一性を比較する関数，パラメータの値の領域を複製・解放する関数の参照を格納する．

パラメータ仕様構造体 パラメータ仕様 (parameter specification) を表現する．パラメータの値の長さ，パラメータ型構造体の参照を格納する．

パラメータリスト仕様構造体 パラメータのリストの仕様を表現する．リストに含まれるパラメータの数，リスト内のパラメータごとのパラメータ型構造体の参照を格納する．

パラメータデータ構造体 パラメータの値を表現する．パラメータの値により領域長が変わる．領域長と値をもつ．

パラメータリスト構造体 パラメータのリストを表現する．参照カウンタ，パラメータリスト仕様構造体の参照，リスト内のパラメータごとのパラメータデータ構造体の参照を格納する．

4.1.2 パラメータ型

以下のビットストリーム，オクテットストリーム，整数を表現するパラメータ型を実装した．

- 固定長ビットストリーム（0–128 ビット長）
- 可変長ビットストリーム（0–128 ビット長）
- 符号なし整数 ($0 \leqq N \leqq 2^{64}-1$)
- 符号付き整数 ($-2^{63} \leqq N \leqq 2^{63}-1$)
- 固定長オクテットストリーム（0–1024 オクテット長）
- 可変長オクテットストリーム（0–1024 オクテット長）

固定長ビットストリームはビットの列で，IPv4 アドレス，IPv6 アドレス等を表現する．IPv6 アドレスを表現可能にするため，128 ビット長までを扱えるように実装した．長さは分類スキーマ毎にパラメータ仕様で指定する．

可変長ビットストリームはビット列とそのビット長さを持ち，IPv4 アドレスプレフィックス，IPv6 アドレスプレフィックス等を表現する．固定長ビットストリームと同様に，128 ビット長までを扱えるように実装した．

符号なし整数，符号付き整数は 64 ビット長の整数を扱える．ポート番号などを表現する．

固定長オクテットストリーム，可変長オクテットストリームは 0 から 1024 オクテット長までのオクテット列を扱える．

固定長オクテットストリームは長さをスキーマ毎にパラメータ仕様で，可変長オクテットストリームは長さをパラメータ毎に指定する．文字列等を表現する．

以上の型に属さない特別なパラメータとして，ワイルドカードパラメータを導入した．これは，分類レコードの値として使用することで，あらゆる対象のパラメータと一致することを示す．

4.2 パラメータ比較

対象に一致する分類レコードを得るためには，対象から分類スキーマに基づいて取り出した属性と，分類レコードの値を比較する必要がある．

パラメータ比較のために，パラメータ比較関数を使用する．パラメータ比較関数は，対象の属性のパラメータリストの各パラメータ（以下キーパラメータとする）とキーパラメータのパラメータ仕様，分類レコードの値のパラメータ（以下レコードパラメータとする）と分類レコードのパラメータ仕様を引数にする．キーパラメータとレコードパラメータが一致すれば MATCH を，一致しなければ NOMATCH を返す．

(注1)：http://www.moon-bear.net/

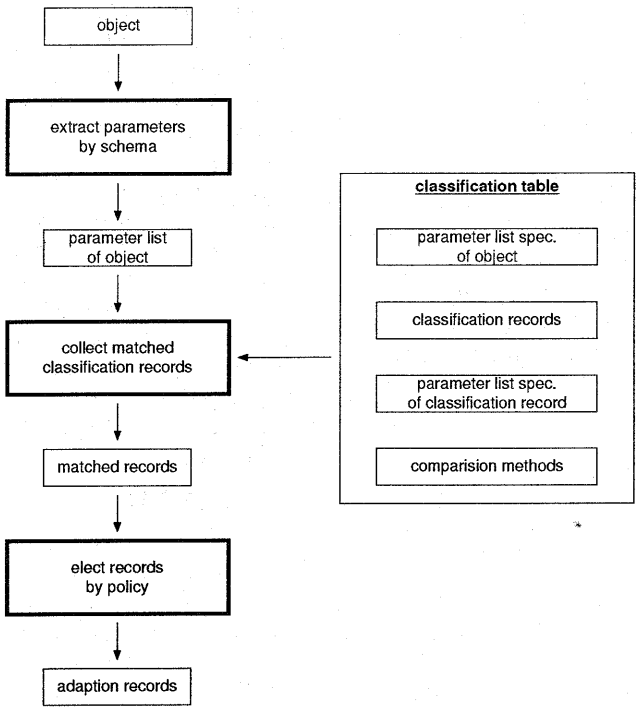

図 5　KUPF の処理の流れ
Fig. 5　processing flow of KUPF

レコードパラメータから比較優先度を求めるために，比較優先度関数を使用する．キーパラメータ k_1，レコードパラメータ p_1, p_2 を引数にし，$p_1 \leqq k_1 \leqq p_2$ が成立した時に MATCH を返す比較関数 range を考える．$k_1 = 5, p_1 = 4, p_2 = 6$ と $k_1 = 5, p_1 = 1, p_2 = 9$ は共に MATCH となる．両者を比べると，前者の方が後者より限定的なレコードパラメータを持つといえる．比較関数 range の比較優先度関数 $P_{range}(p_1, p_2)$ は以下の条件を満たす．

$$p_2 - p_1 = q_2 - q_1 \Rightarrow P_{range}(p_1, p_2) = P_{range}(q_1, q_2) \quad (1)$$

$$p_2 - p_1 < q_2 - q_1 \Rightarrow P_{range}(p_1, p_2) < P_{range}(q_1, q_2) \quad (2)$$

$$p_2 - p_1 > q_2 - q_1 \Rightarrow P_{range}(p_1, p_2) > P_{range}(q_1, q_2) \quad (3)$$

実装としては，次式が一例として考えられる．

$$P_{range}(p_1, p_2) = ((\text{パラメータ最大値}) - (p_2 - p_1)) \quad (4)$$

ワイルドカードパラメータの場合は，常に 0 と定義する．

4.2.1　構　造　体

パラメータを比較するために以下の構造体を定義した．

パラメータ比較方法構造体　パラメータ比較方法を表現する．パラメータ比較関数・比較優先度関数の参照を格納する．さらに，キーパラメータのパラメータ型構造体の参照，必要なレコードパラメータの数，レコードパラメータリストのパラメータごとのパラメータデータ構造体の参照を格納する．

4.2.2　パラメータ比較方法

本実装では，以下のパラメータ比較を実装した．

match

比較関数は同じ型のキーパラメータ k_1 とレコードパラメータ p_1 を引数にする．型が固定長ビットストリーム，可変長ビットストリームの場合，k_1, p_1 のビット長が等しく，全ビットが等しい時は MATCH を，それ以外の時は NOMATCH を返す．型が符号なし整数，符号付き整数の場合，数学的に k_1, p_1 が等しい時は MATCH を，等しくない時は NOMATCH を返す．型が固定長オクテットストリーム，可変長オクテットストリームの場合，k_1, p_1 のオクテット長が等しく，全オクテットが等しい時は MATCH を，それ以外の時は NOMATCH を返す．比較優先度関数は，常に 1 を返す．

mask match

比較関数は固定長ビットストリーム型のキーパラメータ k_1 と 2 つの固定長ビットストリーム型のレコードパラメータ p_1, p_2 を引数にする．k_1, p_1, p_2 のビット長をそれぞれ k_1^{len}, p_1^{len}, p_2^{len} とし，n ビット目をそれぞれ k_{1n}, p_{1n}, p_{2n} とする．以下の式が成立する時は MATCH を，成立しない時は NOMATCH

を返す.

$$k_1^{len} = p_1^{len} = p_2^{len} \quad (5)$$

$$k_{1n} \cdot p_{2n} = p_{1n} \cdot p_{2n} \quad (n = 1, ..., k_1^{len}) \quad (6)$$

比較優先度関数は，$\sum_{n=1}^{k_1^{len}} p_{2n}$ を返す

prefix match

比較関数は，固定長ビットストリーム型のキーパラメータ k_1 と可変長ビットストリーム型のレコードパラメータ p_1 を引数にする．k_1, p_1 のビット長をそれぞれ k_1^{len}, p_1^{len} とし，nビット目をそれぞれ k_{1n}, p_{1n} とする．以下の式が成立する時はMATCHを，成立しない時はNOMATCHを返す．

$$k_1^{len} \geqq p_1^{len} \quad (7)$$

$$k_{1n} = p_{1n} \quad (n = 1, ..., p_1^{len}) \quad (8)$$

比較優先度関数は，p_1^{len} を返す．

また，これら以外のパラメータ比較の追加できるように，実装には柔軟性を持たせている．

4.3 分類表

分類表，分類レコードを表現するために以下の構造体を定義した．

分類レコード構造体 分類レコードを表現する．参照カウンタ，使用カウンタ，パラメータリスト仕様構造体の参照，リスト内のパラメータごとのパラメータデータ構造体の参照を格納する．さらに，適用操作の参照とその領域を解放する関数の参照を格納する．また，4.2.2節で説明した比較優先度関数の結果は，レコードパラメータと比較方法により定まるので，値を構造体に格納する．

分類表ノード構造体 分類表内で分類レコードのリストを管理するノード．リスト管理の情報と，分類レコード構造体自体を格納する．

分類表統計構造体 分類表ごとの，分類レコードの選出回数カウンタ，キャッシュヒット・ミスヒットカウンタ，分類レコード追加・削除カウンタを格納する．

分類表構造体 分類表を表現する．対象の属性のパラメータリストと分類レコードのパラメータリストのパラメータリスト仕様構造体の参照を格納する．さらに，キーパラメータリストのパラメータごとのパラメータ比較方法構造体の参照，分類表リストの参照，4.4節で説明するキャッシュを格納する．

分類レコードは，分類表ノード構造体をもちいて線形リストで保持する．

4.4 第1ステージ

第1ステージは，分類表の分類レコードから対象のパラメータリストに一致する分類レコードをすべて選出する．分類表は，対象のパラメータリスト仕様，分類レコードリスト，分類レコードのパラメータリスト仕様，パラメータごとの比較方法リストを保持する．本実装では，分類レコードを線形リストで保持し，選出の都度全パラメータを比較関数をもちいて比較する．

高速化のために，選出結果のキャッシュを実装した．選出が終わるたびに，選出に使った対象のパラメータリスト，選出結果の分類レコードのリストをキャッシュに格納する．次回の選出の時，対象のパラメータリストが前回と同じであれば，前回の選出結果を返す．分類レコードの追加・削除の際はキャッシュを破棄する．

4.5 第2ステージ

第2ステージは，選出した分類レコードから抽出ポリシにしたがって適合レコード（群）を抽出する．

本実装では，ベストマッチを実装した．分類レコードのマッチの度合いは，分類レコード内の各レコードパラメータの比較優先度値により決まる．比較優先度の等しい分類レコードを区別するために，分類レコード優先度をもちいる．そのため，複数の優先度の尺度が存在する．これらの優先度の適用順序を変えることでポリシを反映する．適用順序は，適合レコードの抽出の都度指定する．

5. 実証実験

WIDE Project(注2) 2002年3月の4日間の研究会において，アドホックネットワークを構築し，実証実験を行った．研究会には273人が参加し，ATM回線1.5 Mbpsと衛星回線上り0.5 Mbps，下り1.5 Mbpsの対外線により IPv4/IPv6 のインターネット接続を参加者に提供した．このネットワークでは，『あやめプロジェクト』(注3) および『くまプロジェクト』による，ユーザから動的に帯域要求を行なえるネットワークの運用・実験を行った．

あやめプロジェクトは，次世代インターネットの基盤構造に対する考察をもとに，新しいMPLS[3]の実装と研究を行っている．今回の実験では，転送層としてあやめプロジェクトのAYAME MPLS実装を利用した．また，MPLSのシグナリングプロトコルとしては，トラヒックエンジニアリングに適したCR-LDP[4]を利用した．くまプロジェクトは，多様なポリシを多様な方法で実現するためのアーキテクチャ，コンポーネント間のインタフェイス，さまざまなポリシを調停する多目的フィルタなど，広域ネットワーク制御を実現するためのフレームワークの研究を行っている．今回の実験では，制御層としてくまプロジェクトの制御パラメータトランスポート，細分化意思決定機構，細分化意思統合機構[5]を利用した．

図6にネットワーク外略図を示す．ユーザセグメント収容ルータからインターネットとの接続点までのルータは，衛星回線の両端を除きすべてAYAME LSRで構成した．MPLSドメイン内での通信品質（帯域）保証のために，MPLS/Diffservの技術を用いた．実験ネットワークでは，帯域におけるボトルネックは対外線のみであった．このため，LSRエッジルータにおいてLSP毎のポリシングとマーキングを，対外線収容ルータにおいてマークに基づいた優先処理を行った．KUPFは，LSRエッジルータにおいて，制御パラメータにもとづいてフローを

(注2)：http://www.wide.ad.jp/
(注3)：http://www.ayame.org/

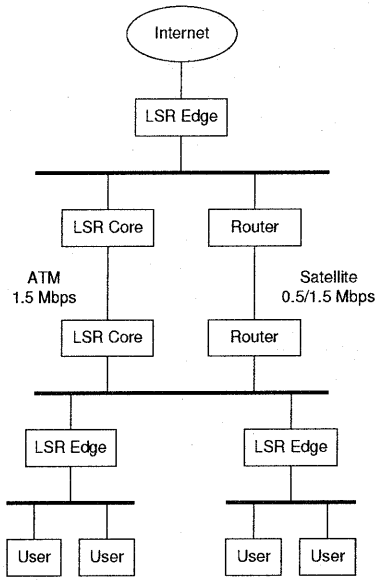

図 6　ネットワーク簡略図
Fig. 6　network outline

表 1　IPv4 分類スキーマ
Table 1　classification schema for IPv4

属性	型（長さ）
プロトコル	符号なし整数
TOS	固定長ビット列（8 ビット）
終点アドレス	固定長ビット列（32 ビット）
始点アドレス	固定長ビット列（32 ビット）
終点ポート	符号なし整数
始点ポート	符号なし整数
IPsec 情報	符号なし整数

識別し，MPLS のラベルを決定するために用いた．デフォルトでは通信は遅延の大きい衛星回線を経由する．ユーザが WWW インターフェースにより回線を予約すると予約したユーザの通信のみ遅延の小さい ATM 回線を利用できる．

LSR ルータは NetBSD 1.5.2 が動作する PC を用いた．フロー識別のために，KUPF を LSR エッジルータのカーネル内に組み込んだ．

IPv4 フローに用いた分類スキーマを表 1，分類レコードパラメータ仕様を表 2 に示す．IPv6 フローに用いた分類スキーマを表 3，分類レコードパラメータ仕様を表 4 に示す．IPv4, IPv6 共にプロトコル，終点ポート，始点ポート，IPsec 情報は符号なし整数で表し，比較には match を用いた．TOS およびトラフィッククラスは固定長ビット列で表し，比較には mask match を用いた．終点アドレス，始点アドレスは固定ビット列で表し，比較には prefix match を用いた．これは，最長一致則にしたがう．以上のパラメータにしたがって，フローを識別した．

表 3　IPv6 分類スキーマ
Table 3　classification schema for IPv6

属性	型（長さ）
プロトコル	符号なし整数
トラフィッククラス	固定長ビット列（8 ビット）
フローラベル	符号なし整数
終点アドレス	固定長ビット列（128 ビット）
始点アドレス	固定長ビット列（128 ビット）
終点ポート	符号なし整数
始点ポート	符号なし整数
IPsec 情報	符号なし整数

表 5　キャッシュヒット率
Table 5　cache hit rate

属性	要求回数	ヒット数	ヒット率
LSR5	4,774,389	1,772,166	37.1%
LSR6	3,151,356	1,273,165	35.3%

今回の実験では，以下のルールが競合した．

(1)　始点アドレスが予約ユーザのフローは ATM 回線を経由してインターネットへ

(2)　終点アドレスがユーザセグメントの場合は，対外線を経由せずにユーザセグメント収容 LSR エッジルータへ

予約ユーザからユーザセグメントへのフローは，(1), (2) の両方に一致する．しかし，(1) のルールが適用されると ATM 回線を経由してインターネットへ出た後戻って来ることになり，対外線を 2 度通ることになる．したがって，終点アドレスがユーザセグメントである IP データグラムは，予約の有無に関わらず，ユーザセグメント収容 LSR エッジルータへ転送されることが要求される．KUPF では，第 2 ステージで適用順序を 1. 始点アドレス，2. 終点アドレスとするとで，この問題を避けた．

適用順序があらかじめ決められている実装では，意図しないルールの適用を避けるためには，レコードの追加が必要である．本実装では，優先度，適用順序により柔軟に対応できた．

研究会期間中，いくつかの軽微な障害は発生したものの，サービス全体としては大きな障害もなく，問題なくサービス提供できた．

表 5 に研究会 3 日目における KUPF 第 1 ステージのキャッシュヒット率を示す．LSR5, LSR6 はユーザセグメント収容 LSR エッジルータである．この結果から，第 1 ステージのキャッシュヒット率が 3 割から 4 割程度であり，キャッシュが有効に働いていることが分かった．

6.　今後の課題

今回の実装は，モデルに基づいた設計に忠実にしたがった．そのため，速度の高速化は，第 1 ステージの単純なキャッシュのみである．さらなる高速化のためには，アルゴリズムの改善，ステージ間の協調，ハードウェア処理が必要である．

今回の実証実験では，IP データグラムの第 3 層，第 4 層のパラメータを用いたフローの識別を行なった．提案システムの

表 2　IPv4 分類レコードパラメータ仕様
Table 2　classification record parameters specification for IPv4

属性	型（長さ）	比較方法
プロトコル	符号なし整数	match
TOS	固定長ビット列（8 ビット），固定長ビット列（8 ビット）	mask match
終点アドレス	可変長ビットストリーム	prefix match
始点アドレス	可変長ビットストリーム	prefix match
終点ポート	符号なし整数	match
始点ポート	符号なし整数	match
IPsec 情報	符号なし整数	match

表 4　IPv6 分類レコードパラメータ仕様
Table 4　classification record parameters specification for IPv6

属性	型（長さ）	比較方法
プロトコル	符号なし整数	match
トラフィッククラス	固定長ビット列（8 ビット），固定長ビット列（8 ビット）	mask match
終点アドレス	可変長ビットストリーム	prefix match
始点アドレス	可変長ビットストリーム	prefix match
終点ポート	符号なし整数	match
始点ポート	符号なし整数	match
IPsec 情報	符号なし整数	match

汎用性を示すためには，IP データグラムのフロー識別以外での実証実験が必要である．

7. ま と め

本研究では，パラメータフィルタの要求事項を明らかにし，一般化したモデルを構築した．このモデルに基づき，パラメータフィルタの設計および実装を行い，実証実験を行った．実証実験により，モデルの有効性を示した．また，本実装においてもキャッシュ機構が有効に働くことが判明した．今後の課題としては，高速化，適応範囲の拡大が挙げられる．

文　献

[1] S. Blake and D. Black and M. Carlson and E. Davies and Z. Wang and W. Weiss, "An Architecture for Differentiated Service," RFC 2475, Dec. 1998.

[2] J. Boyle and R. Cohen and D. Durham and S. Herzog and R. Rajan and A. Sastry, "The COPS (Common Open Policy Service) Protocol," RFC 2748, Jan. 2000.

[3] E. Rosen and A. Viswanathan and R. Callon, "Multiprotocol Label Switching Architecture," RFC 3031, Jan. 2001.

[4] L. Andersson and P. Doolan and N. Feldman and A. Fredette and B. Thom as, "LDP Specification," RFC 3036, Jan. 2001

[5] Naoto Morishima and Akimichi Ogawa and Hiroshi Esaki and Osamu Nakamura and Suguru Yamaguchi and Jun Murai, "Preliminary Field-Trial for QoS Routing and Dynamic SLA," IEICE transactions on communications, Vol.84, No.8, pp2039–2047, Jan. 2001