Internet Week98
チュートリアル

QoS技術: Int-servとDiff-serv

長 健二朗
ソニーコンピュータサイエンス研究所
kjc@csl.sony.co.jp

チュートリアルの内容

- トラフィック管理とQoS
- Int-servとRSVP
- Diff-serv

トラフィック管理とQoS

インターネットにおけるトラフィック管理

□目的
- ○多様なサービスへの要求を満足すること
- ○資源を効率的に配分すること

□トラフィック管理
- ○ポリシーとメカニズムの集合

□チャレンジ
- ○ユーザの要求が多様、変化が激しい
- ○効率的な実現
  - ▹スケール、多様性、変動の速度、変化の予測が困難

経済原理

□豊富なサービス品目
- ○多様なユーザの要求を満たす

□サービス・インフラの統合
- ○e.g., voice & data

□集約
- ○規模が大きくなるほど単価は低くなる
  - ▹10M/10人より100M/100人
- ○トラフィックの統計多重効果
  - ▹ピーク値計算 --> 平均値計算
- ○（耐故障性の問題）

トラフィック・モデル

□電話網トラフィックモデル
- ○発呼到着、通話時間ともにポアソン分布で近似
  - ▹相関がない（独立事象）
  - ▹統計多重効果
- ○キューイング理論
  - ▹本当はデータ通信から始まった
- ○６０年代に確立、長い間成功してきた
  - ▹FAX、データ通信の出現で変化

インターネット・トラフィックモデル

- 電話網モデルとの違い
  - ネットワークごとに特徴が異なる
  - 急速な質的変化 （新しいアプリケーションの出現）
  - 自己相似性： 従来のモデルに疑問
- モデルの種類
  - LAN/WAN/アクセス網
  - 単一フローモデル vs 複合フローモデル
  - 実測モデル vs 合成モデル（数式モデル）

トラフィックの自己相似性

- トラフィックはどの時間スケールで見てもバースト的
  - 時間的、空間的変動が大きい
  - 自己相似 = フラクタル
- 多重化しても滑らかにならない → ポアソンモデルに反する
- heavy tailな分布： 無限大の分散
  - 原因の可能性
    - ファイルサイズ、WWWページサイズ、
    - CPUタイム、フィードバック系
    - 集約すると自己相似性が出現

キューイング理論

□ARPANETの時代からデータ通信と共に発展した
理論体系

□システムのスループットを計算する
- ○確率関数で表される入力レート、サービス時間を仮定

□基本：M/M/1 モデル
- ○入力レート：ポアソン到着
- ○サービス時間：指数分布
- ○サーバー数：1

キューイング理論 (2)

□リトルの法則

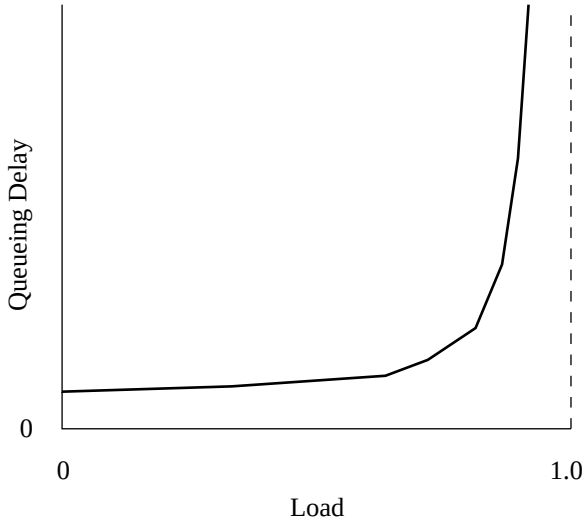
- ○N = λT
  - ▹N: システム内の平均プロセス数
  - ▹λ: 平均到着レート
  - ▹T: 平均サービス時間

□キューイング・システム内の平均プロセス数
は、平均到着レートと平均サービス時間の積に
等しい

キューイング理論 (3)

□キューイング・システムの挙動

○負荷があるポイントを越えると急速に効率悪化



キューイング理論の応用

□システム性能計算、ボトルネックの発見

○コンポーネントをキューに置き換えてブラックボックス化

○粒度を変えて解析 （ボトルネック解析）

▹例： web server

□キューイング理論の限界

○マクロな統計的解析

○システムエンジニアリングでは例外時のミクロな挙動が重要

○キャパシティを越える入力があるときは解析不能

○フィードバック系、相関がある入力の解析が困難

トラフィック管理の時間粒度とメカニズム

- Long-term:
  - 容量設計
- Day:
  - ピーク時間料金制
- Session:
  - サービス別料金制、アドミション制御、ルーティング
- Round Trip Time:
  - エンド・エンド・フロー制御
- Packet Time:
  - キューイング方式
- それ以下:
  - データリンク層依存

Best-effort サービス

- インターネットの基本概念
  - 簡潔さと自由の精神で成功
  - ネットワーク
    - 最善を尽くすが、何の保証もしない
    - （基本的に）すべてのパケットを公平に扱う
  - アプリケーション
    - いつでも、いくらでも勝手にパケットを送ることができる（自由）
    - ネットワークの状態に適応することが要求される （モラル）

- ユーザの要求がBest-effortサービスでカバーしきれなくなってきた

QoS技術

□Best-effortと完全保証の間の様々なサービス品質

□目的
- ○期待されるアプリケーション性能を満足する
- ○ネットワーク資源の分配を制御する
- ○プロバイダ(ISP)による幅広いサービスを実現する

なぜQoSが必要か？

□アプリケーションの変化
- ○リアルタイム性の要求 （コンティニュアス・メディア）
- ○プレミアムサービス／ディスカウントサービスの要求

□資源の有効分配
- ○多様なアプリケーション、多様なネットワークの出現
  - ▷トラフィック変動の速度が速い、幅が広い
- ○従来のエンド・エンド制御の限界
  - ▷光速の限界

QoSとCoS

- QoS (Quality of Service) （狭義のQoS)
  - 定量的に表すことのできる絶対的性能
    - Int-serv model
- CoS (Class of Service)
  - クラス間の相対的性能差
    - Diff-serv model

QoSとは何か

- アプリケーションから見えるネットワーク性能
  - 帯域、遅延、ジッタ、パケット損失率
- QoSの制御
  - ネットワーク上の資源配分を制御
    - パケット・スケジューリング
    - バッファ
    - CPUパワー

キューイング遅延

□遅延要因のなかでキューイング遅延が圧倒的に大きい

○コントロールすると効果大

Time required to send a 1KB packet

Link (bps)	64K	512K	1.5M	10M	45M	100M
time (usec)	125,000	15,625	5,208	800	178	80

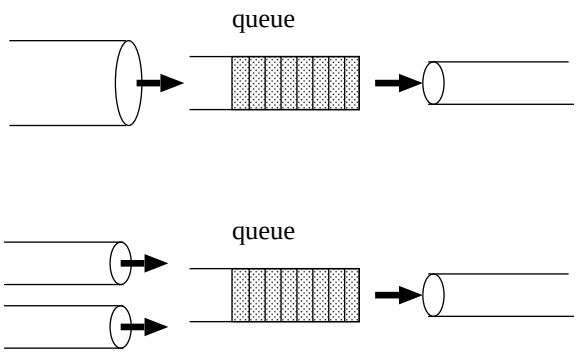
QoS制御の有効なポイント

□ボトルネック

○キューが増大、溢れる

□トラフィックの集中

○複数の流入口から特定の出口へ



QoS vs Over-provisioning

- 回線が速ければ輻輳はおこらない？
  - QoS
    - トラフィックをコントロールする
  - Over-Provisioning（過剰投資）
    - 需要に対して十分な資源を用意する

  - 小規模ではover-provisioningである程度いける
  - 広域ではQoSの制御が必要
    - トラフィックの集中は予測不能
    - ネットワークのトラフィックは自己相似的

- 二者択一ではない
  - コストと運用を考慮したバランス

QoS技術要素

- アドミション制御 (admission control)
  - 動的な資源配分

- クラシファイア (classifier)
  - 到着パケットを対応するグループに分ける機構

- パケット・スケジューラ (packet scheduler)
  - 各グループに応じたパケットの送出

- シェーパ (shaper)
  - バーストを一定のレートに均す

アドミション制御

- セットアップ・プロトコルによってパス上の資源を確保する (signaling)
- 資源が不足すると事前に失敗する （エラーが返る）
- 資源の解放（特に故障時）
- 関連技術
  - ポリシー、ルーティング、課金

クラシファイア

- パケットをクラス分けする機構
  - IPでは５つ組が使われてきた
    - src_addr, dst_addr, src_port, dst_port, proto
    - （ファイアウォールのパケットフィルタと同様）
  - TOSフィールド
  - ワイルドカードの扱い

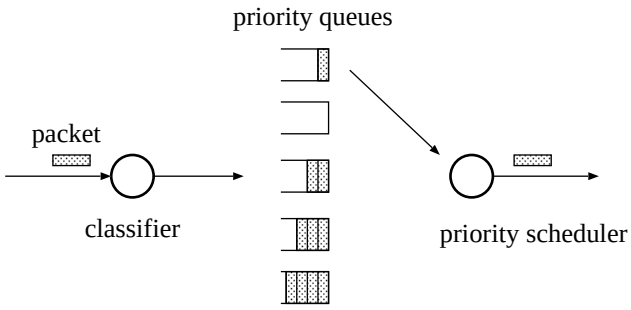
パケット・スケジューラ

□キューイング方式
- ○Priority Queueing
- ○WFQ (Weighted Fair Queueing)
- ○CBQ (Class-Based Queueing)

□バッファ管理
- ○Drop-Tail/Drop-Head/Drop-Random
- ○RED (Random Early Detection)

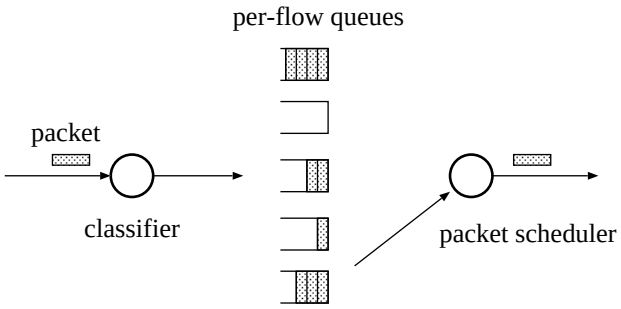
Priority Queueing

□優先スケジューリング
- ○単純な機構でリアルタイム性の保証
- ○低優先度クラスが枯渇する可能性



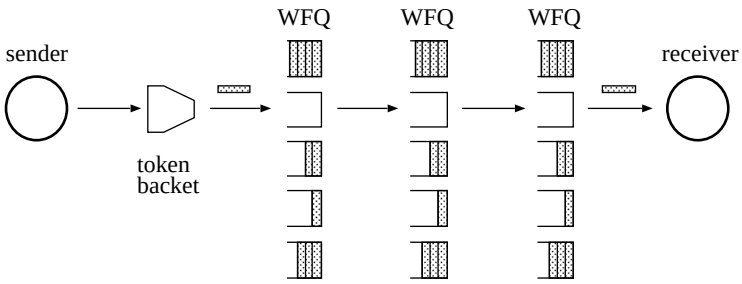
WFQ (Weighted Fair Queueing)

- フローごとに独立したキューを割り当てる
  - 他のフローの影響を一定以下に押えることが可能
  - フローの数だけキューが必要
    - 実装には何らかの近似法が使われる



Parekh's Model

- トークンバケットとWFQの組み合わせ
  - 最大遅延保証が可能なことを証明



Parekhの最大遅延計算

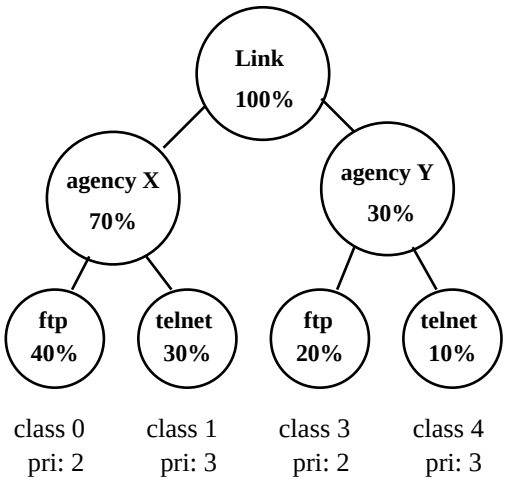
- バースト遅延 + 自フローの遅延 + 他のフローによる遅延

$$D_i = \frac{b_i}{g_i} + \frac{(h_i - 1)\, l_i}{g_i} + \sum_{m=1}^{h} \frac{l_{max}}{r_m}$$

D_i	delay bound for flow i
b_i	token bucket size for flow i
g_i	weighted rate for flow i
h_i	hop count for flow i
l_i	max packet length for flow i
r_m	bandwidth at hop m

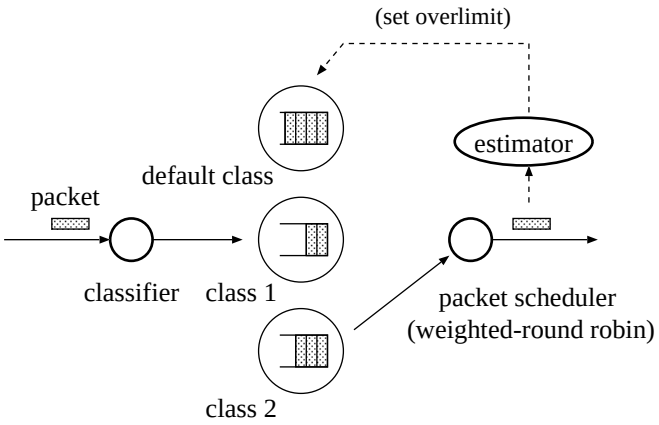
階層的リンクの共有

- 集約されたフローを階層的に制御



CBQ (Class-Based Queueing)

- 階層的リンクの共有を実現
- 非ワークコンサービング

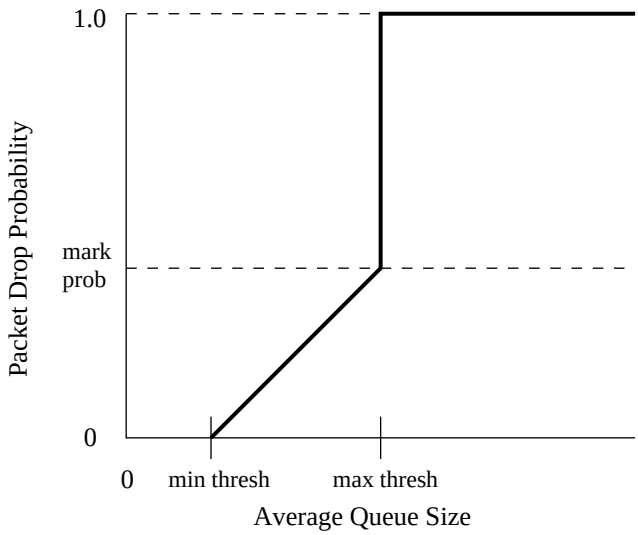


RED (Random Early Detection)

□ 平均キューサイズに応じて確率的にパケットを廃棄
- TCPがキューが溢れる前に流量をコントロールできる
  - キュー長を短く保つ
  - キューイング遅延を小さく保つ
- 平均キューサイズを使うことで短いバーストを許容
- バッファ占有率に応じたフェアな廃棄
- 廃棄のかわりにマークする
  - ECN (Explicit Congestion Notification)

□ 欠点
- パケット損失に応答しないトランスポートには無防備
  - RED Penalty-box

REDパケット廃棄確率



Int-servとRSVP

Integrated Services

- インターネットにおけるQoSへの取り組み
  - 研究 89～ IETF 94～
- 標準化 (Interoperabilityに関するもの)
  - Int-serv
    - サービスの分類、QoSパラメータの指定
  - RSVP
    - セットアップ・プロトコル（レファレンス・モデル）
    - プロトコル - IETF
    - 実装　　- ISI
  - ISSLL (Integrated Services over Specific Link Layers)
    - 特定のデータリンク層に依存する技術
- 標準化しないもの（実装依存）
  - トラフィック制御機構、ポリシ制御機構

Int-serv

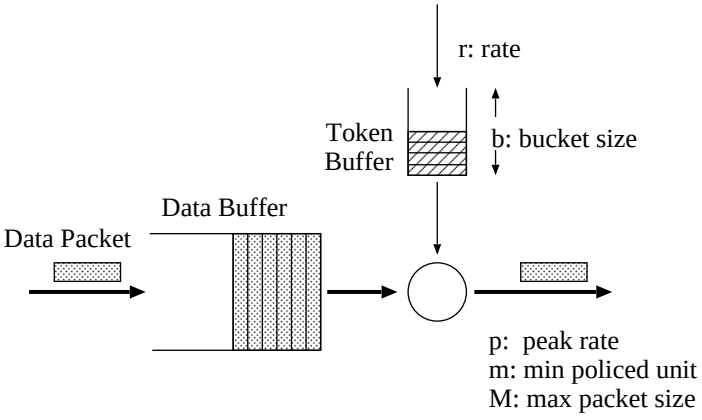
- Guaranteed Service
  - 帯域保証
  - 最大遅延の保証
- Controlled Load Service
  - 適応型アプリケーションを想定
    - 流量が制御される
    - （遅延値目標がある）
  - 利用率の低いBest-effortネットワークをエミュレート
    - たとえ、ネットワークが混んでいても実現
    - 統計多重を用いる（ソフトな保証）

最大遅延の保証

- リアルタイム・アプリケーション
  - プレイバック・ポイント
- トラフィックの分離 (isolation)
  - 他のトラフィックからの保護
- フロー制御とシェーピング
  - 自分自身のトラフィックからの保護

Token Bucket TSpec

- トークンバケットで表されるトラフィックパラメータ
  - [r, b, p, m, M]



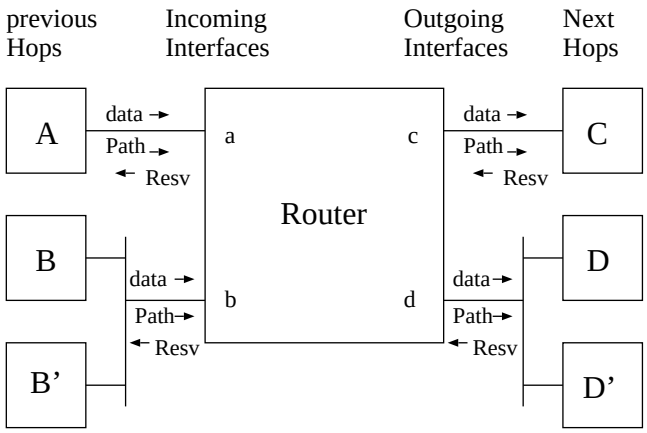
RSVPの特徴

- RSVPは資源予約プロトコル
  - マルチキャストとユニキャストの資源予約を行なう
  - IPv4とIPv6で動作する
- シンプレックス
  - 一方向のデータフローに対する予約を行なう
- レシーバ主導
  - レシーバが予約を起動する
- ソフトステート
  - 経路変更、マルチキャストのメンバ変更に動的に適応

RSVPの特徴 (2)

- ルーティング・プロトコルからの分離
  - ルーティング情報をもらって利用する
- トラフィック制御、ポリシ制御からの分離
  - 予約要求はトラフィック制御／ポリシ制御モジュールに渡され、許可の判断が行なわれる
  - オブジェクトを転送、管理するがその内容はRSVPの知るところではない
- 複数の予約スタイル
  - Fixed-Filter, Shared-Explicit, Wildcard-Filter
- 非RSVPネットワークの透過
  - RSVPをサポートしないルータが途中にあっても動作する

RSVPのモデル



RSVPメッセージタイプ

- ○Path
  - ▹フロー情報のアナウンス
- ○Resv
  - ▹予約
- ○PathTear、ResvTear
  - ▹Path、Resvの終了（ステートの破棄）
- ○PathErr、ResvErr
  - ▹Path、Resvの失敗（ステートの破棄）
- ○Confirm
  - ▹予約確認
- ○DiagReq
  - ▹診断要求
- ○DiagRep
  - ▹診断応答

Pathメッセージ

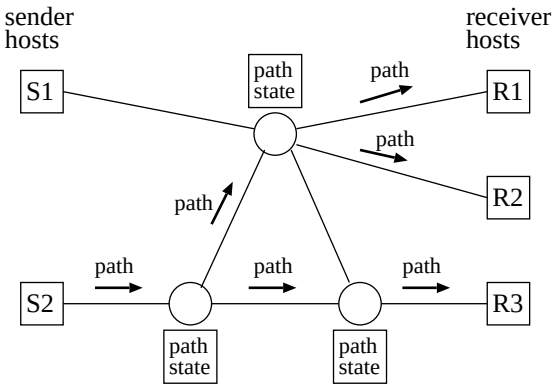
- Sender
  - 定期的にPathメッセージを送る
    - Sender Template:
    - -- Filter Spec: フローの指定 (e.g., dst addr/port)
    - Sender Tspec: オリジナルのフロー情報
    - （Adspec: 中間ルータが書き換えるフロー情報）
- Router
  - Adspecを変更しながら下流に転送

Resvメッセージ

- Receiver
  - 受信しているPathメッセージに対応して、定期的にResvメッセージを送って予約をする
    - Flowspec: 予約のQoS指定
    - -- Tspec: 予約要求Tspec
    - --（Rspec: Guaranteed QoSの指定）
    - FilterSpec: フローの指定
- Router
  - Resvメッセージにしたがって予約を確保し、上流に転送
  - 予約は可能であればマージする

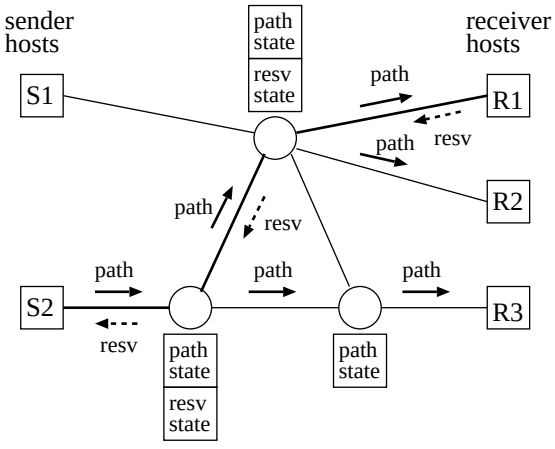
Pathメッセージの流れ

○S2からのマルチキャスト



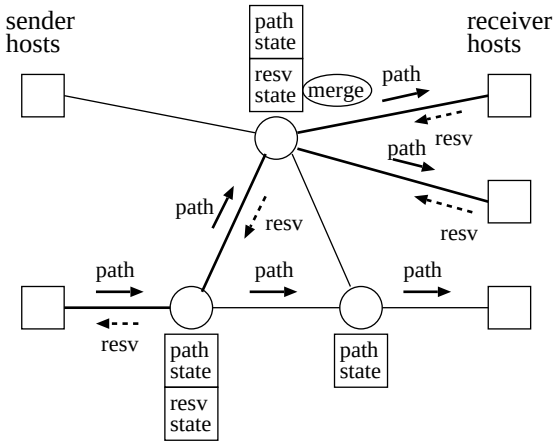
Resvメッセージの流れ

○R1からの予約



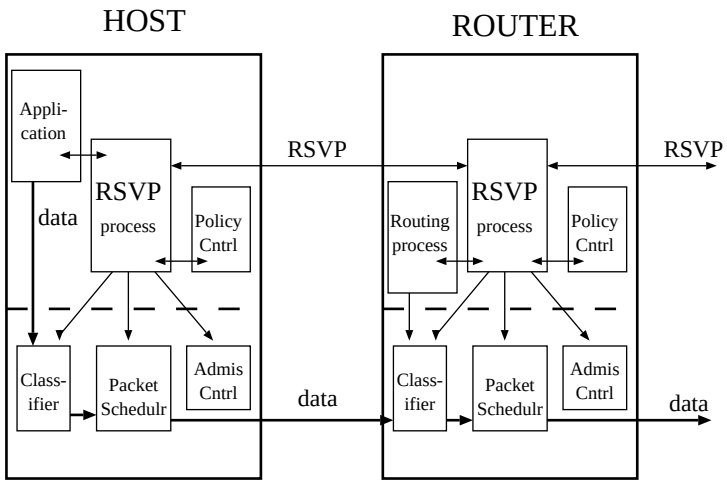
予約のマージ

○R2からの予約のマージ

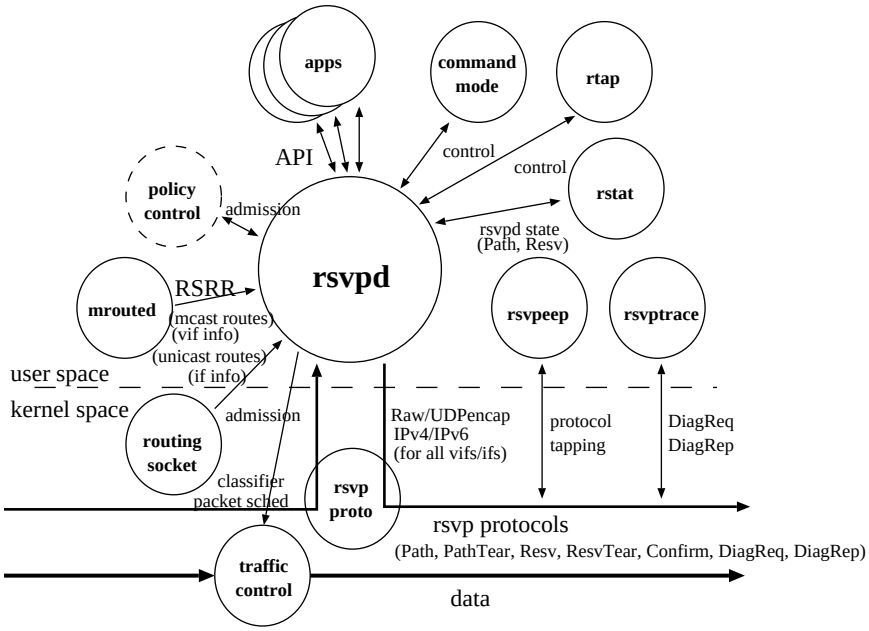


RSVPの実装モデル

□ホストとルータ



ISI RSVPD の実装



ISSLL (Integrated Services over Specific Link Layers)

- IS802
  - Subnet Bandwidth Manager (SBM)
- ISATM
  - Int-servパラメタとATMパラメタのマッピング
- ISSLOW
  - ダイアルアップからT1程度
    - パケット遅延 (1500B / 28.8Kbps ~= 400msec)
    - ヘッダ圧縮
    - フレーミング

RSVPの課題

- ポリシ制御
- トネリング
- ステートの集約
- ２階層資源管理
- RSVP over MPLS
- RSVP over diff-serv

RSVPのまとめ

- 標準化は一段落

- RSVPは死んだか？
  - 一部にRSVPは失敗したという批判がある
    - 十分に普及していない
    - 研究主導でビジネスがついて来なかった
  - メディアの過剰な期待の反動
  - イントラネットでは有効
  - 広域ではスケーラビリティが問題
    - RSVP over diff-servに期待

Diff-serv

Differentiated Services

- Int-serv/RSVPへの疑問
  - 複雑さ、スケーラビリティ、運用
- ISPの要求
  - 簡単に実現できるプレミアムサービス
  - （現状でも非標準で行なわれている）
- ルーターベンダの思惑
  - ISPに売れるルーター
- TOSフィールドの再定義

Diff-servの目的

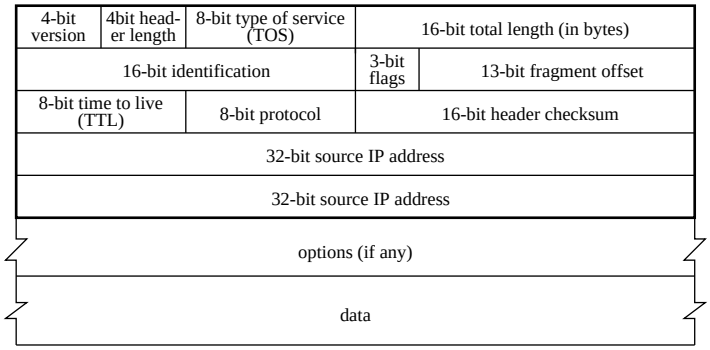
- アーキテクチャへの要求
  - スケールする
    - per flowステートを持たない
  - シンプル
    - TOSフィールドを使ってClassifierを簡単にする
- イントラドメイン／インタードメインのサポート
  - 組織の要求
  - ISPのビジネスモデル
- （現実重視のインターネットらしい発想）

Diff-servの始まり

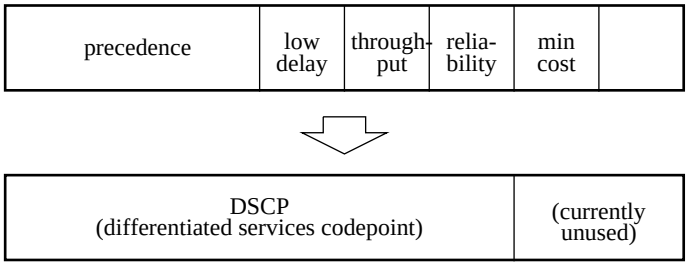
- 1997/08 Munich IETF
  - Int-serv WGが Diff-serv BOFを開いた
  - Premium Service Model
    - V. Jacobson (LBL)
  - Drop preference Model
    - D. Clark (MIT)
  - Cisco’s CoS
    - F. Baker (Cisco)
- 1998/03 IETF Diff-serv WG設立
  - 大学、政府、ベンダ、ISPの大物が協力して急速に標準化が進んでいる

IPv4ヘッダ

- 定義があいまいなTOSフィールドを再定義してDSフィールドとする



TOSフィールド

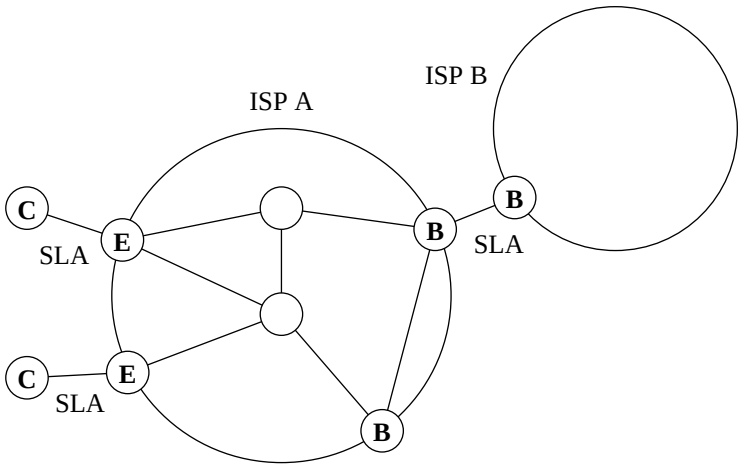


- 2ビットはECNのために残されている
- IPv6のTraffic Classフィールドにも適用

Diff-serv モデル

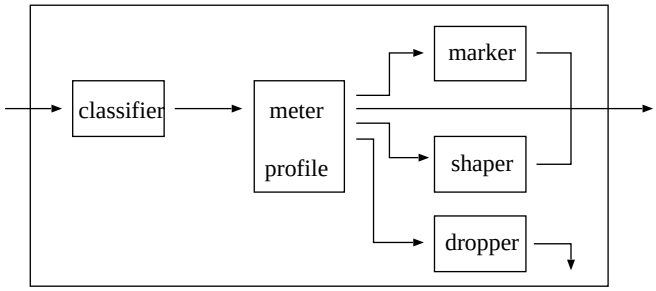
- ネットワークの入口でトラフィックをコントロールする
  - エッジノード
    - コードポイントを設定
    - SLA: Service Level Agreement
  - 中間ノード
    - コードポイントに応じてパケットスケジュール
  - バウンダリノード
    - ネットワーク間の取り決めにしたがってコードポイントを書き換える

Diff-serv ネットワークモデル



トラフィックコンディショニング

□エッジノードにおけるルールの適用



traffic conditioner at an edge node

Forwarding Behavior

□パケット内のビット列がローレベルのForwardig Behaviorを指定する

□エッジがSLA(Service Level Agreement)に応じたビット列を設定する

- ○ルータ：forwarding behaviorを実現するメカニズム
- ○ISP: パラメータを設定しサービスを実現

□Behaviorはネットワークにローカル

- ○ISP独自のサービスが提供できる

□ISP間の取り決め

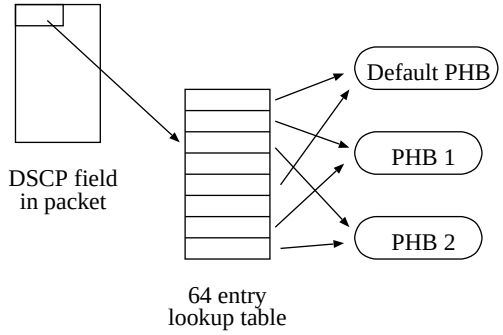
- ○ISP-ISP SLAに応じた再マーキング
- ○グローバルに使える標準を推奨する

PHB (Per Hop Behavior)

- 外部から観測可能なパケットフォワード動作の記述
  - メカニズムとポリシの分離
  - (forwarding vs routing analogy)
- PHBグループ
  - 一連のサービスを実現するために使われる複数のPHB
- さまざまなPHB
  - Best-effortからGuaranteedサービスまでの多様なサービスを実現するためのメカニズム

PHBの選択

- ６ビットコードポイント、６４個のPHB （空間が小さい）
  - コードポイントからテーブルを引きPHBを選ぶ
    - 複数のコードポイントをひとつのPHBにマップできる
  - ISPは独自にコードポイントを設定できる
    - 境界でコードポイントは書き換えられる



Standard PHBs

- Default PHB
  - codepoint "0": best-effort
- Class Selector PHBs
  - IP precedence互換
- Assured Forwarding PHBs
  - Drop preference モデル
- Expedited Forwarding PHBs
  - 仮想専用線モデル

コードポイントの割り当て

- コード空間
  - xxxxx0: Standard PHBs （３２個）
  - xxxx11: Experimental/Local Use （１６個）
  - xxxx01: Experimental/Local Use* （１６個）
- スタンダード空間
  - 000000: Default PHB （１個）
  - xxx000: Class Selector PHBs （７個）
  - +++++0: Assured Forwarding PHBs (１２個)
  - 101110: Expedited Forwarding PHB （１個）
    - AFの４個はClass Selectorと重複

Expedited Forwarding

- JacobsonのPremium Service
  - 仮想専用線を実現する
    - エンドエンドのパスを売る
  - 契約帯域を保証
    - エッジで契約分のトラフィックのみ通す
    - ISPは契約分の帯域を確保（統計多重はいけない）
    - Priority Queueで実現（WFQ、CBQも可）
    - （ATMのCBRサービスと同じ考え方）
  - 使っていない帯域はBest-effortが利用可能

- IETFでの一般化
  - エンドエンド以外のスコープ
  - 統計多重を使った容量計算

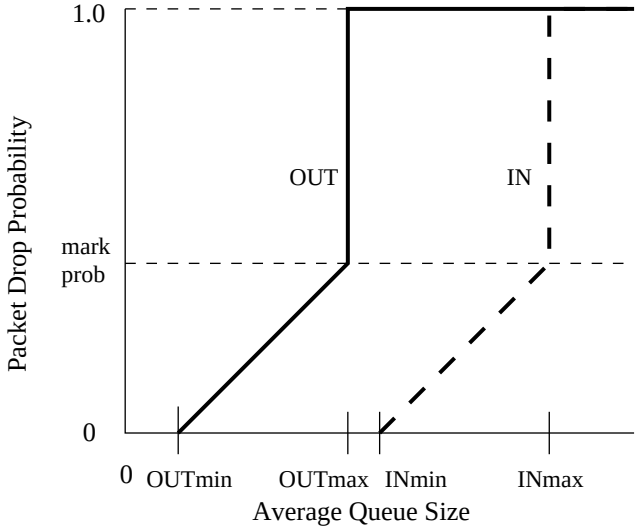
Assured Forwarding

- ClarkのDrop Preferenceモデル
  - 適応型アプリケーション向け
  - 相対的なQoSを持つPHBグループ (RIO, WRED)
  - 最低帯域保証のあるベストエフォット
    - エッジで契約分を越えたトラフィックをOUTとマーク
    - ルータはOUTパケットを先に廃棄する
    - キューは１つなのでパケットの順序入れ換えがない

- IETFでの一般化
  - ４つのクラス、３つのドロップレベル
  - Class Selector PHBとの関係 （重複する）

RIO (RED with In and Out)

- ○IN、OUTパケットに独立したREDパラメータを与える
- ○輻輳が起こるとOUTパケットから先に廃棄される



Diff-servの課題

- □受信者ベースのコンディショニング
  - ○エッジ機能？
- □RSVP over Diff-serv
  - ○RSVPのスケーラビリティの問題を解決？
- □Diff-serv over MPLS
- □２階層資源管理
  - ○ISP境界問題
- □メカニズムについてはまとまりつつある
  - ○大変なのはこれから
  - ○(forwarding vs routing analogy)

Diff-servのまとめ

- シンプルで柔軟な枠組
- 最低限の標準化
  - コードポイントがPHBを選択する
- どのようなサービスを実現するかは実装、運用依存
  - いくつかの標準モデルを推奨

まとめ

- 多様なユーザの要求が出現
  - QoS技術は多様なサービスを提供するためのメカニズム
  - サービスを提供するには運用技術が最大の課題
- Int-serv/RSVP
  - QoS技術の発展、理解には多大な貢献
  - RSVPは当初期待されたほどは普及していない
    - 現状はイントラネット向きのサービス
    - diff-servによって普及が広がる可能性
- Diff-serv
  - 主要なメカニズムの標準化がまとまりつつある
  - 急速な普及が期待される
    - 運用面の課題も多い
    - RSVPや他の技術へのインパクトが大きい
- QoS技術の大きな転換期！

<**関連リンク**>
IETF: http://www.ietf.org/
IETF diffserv WG: http://www.ietf.org/html.charters/diffserv-charter.html
IETF intserv WG: http://www.ietf.org/html.charters/intserv-charter.html
IETF issll WG: http://www.ietf.org/html.charters/issll-charter.html
IETF rsvp WG: http://www.ietf.org/html.charters/rsvp-charter.html
RSVP: http://www.isi.edu/rsvp/
Diffserv at MIT: http://diffserv.lcs.mit.edu/
CBQ: http://www-nrg.ee.lbl.gov/floyd/cbq.html
ALTQ: http://www.csl.sony.co.jp/person/kjc/software.html

<**関連書籍**>
Quality of Service. P. Ferguson and G. Huston. Wiley. ISBN 0-471-24358-2.
An Engineering Approach to Computer Networking. S. Keshav. Addison-Wesley.
ISBN 0-201-63442-2.
High-speed Networks: TCP/IP and ATM Design Principles. William Stallings.
Prentice Hall. ISBN 0-13-904954-1
Gigabit Networking. Craig Partridge. Addison-Wesley. ISBN 0-201-56333-9.
<**キューイング理論**>
Queueing Systems vol. 1 and vol. 2. Leonard Kleinrock. Wiley-Interscience.
ISBN 0-471-49110-1, 0-471-49111-X