

# Rule-based Modular Representation of QoS Policies

Yasusi Kanada  
Hitachi Ltd., Central Reserach Laboratory

## Internet QoS Guarantee and Its Approaches

---

### ■ Needs of QoS guarantee in the Internet

- Mission-critical communications are increasing.
- Multi-media traffics are increasing.

### ■ IntServ and DiffServ from IETF

- Integrated services (IntServ)
  - Flow-based QoS control architecture
  - High overhead and not scalable
- Differentiated services (DiffServ)
  - Class-based QoS control architecture
  - Low overhead and scalable — practical in large-scale networks

## A Model of A DiffServ-ready Network

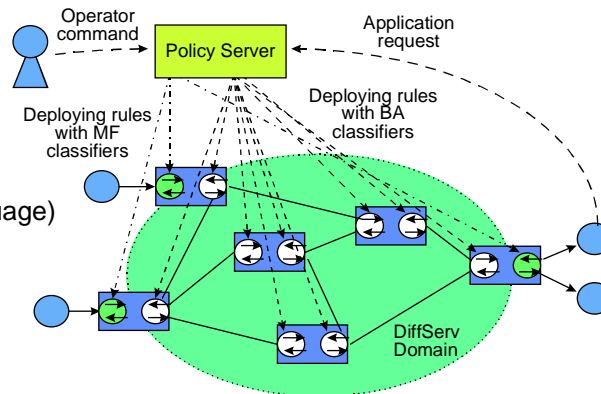
### ■ A QoS policy server

- Required for controlling QoS conditions or routers.

### ■ QoS-ready routers (and QoS-ready switches)

### ■ Interface between a policy server and network nodes

- SNMP
- COPS
- API (CORBA IIOP)
- CLI (command language)



Networking Architecture Workshop 2000-2-3

3

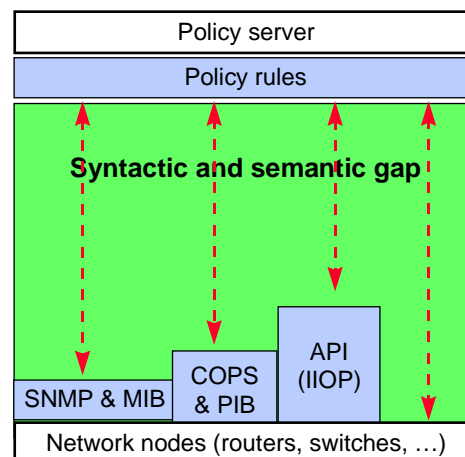
## Problems of conventional PS-Router Interfaces

### ■ Poor syntax

- SNMP: get/set a single value.
- API: function calls only.
- No structuring methods (control structures).

### ■ Poor semantics

- No relation nor constraints can be described.
- Protocols specify only very limited part of the semantics.
- Semantics must be specified formally for interoperability.
  - Standard protocols do not guarantee interoperability any longer.



Networking Architecture Workshop 2000-2-3

4

## An Alternative Interface: A Rule-based Programming Language

---

### ■ Why a language?

- Because a language is a combination of syntax and semantics.

### ■ Why programming?

- Policy-based control is programming.
  - Network nodes have been configured only using parameters (data).
  - We need programs for configuration, because the function to be configured is so complex.

### ■ Why rule-based?

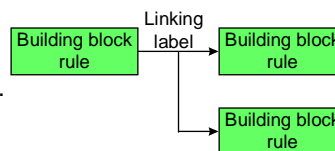
- Because a policy is a rule-based program.
- This language may be similar to languages for expert systems, such as OPS5 or Nexpert Object.

## Elements of the Rule-based Language

---

### ■ The language consists of

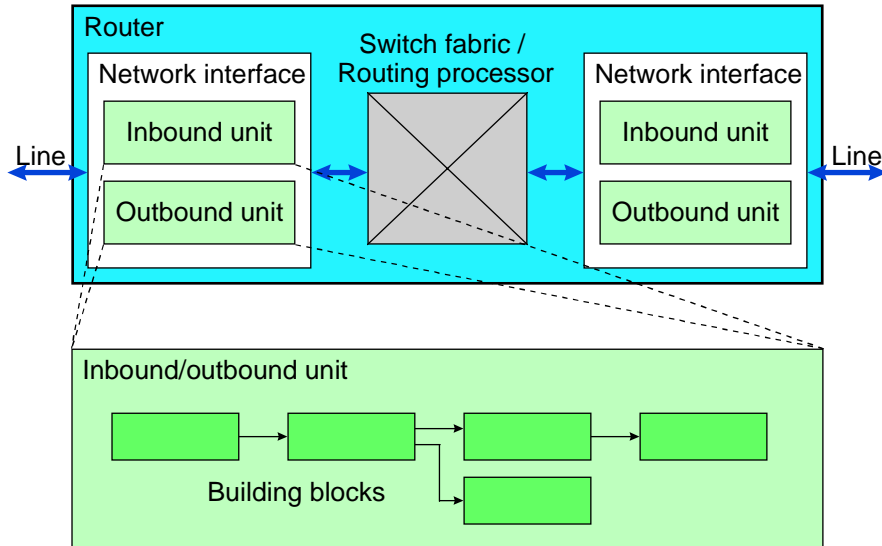
- Building block rules
  - Primitive rules to construct policy rules.
- Linking labels
  - Connections between building blocks.



### ■ What is linking labels?

- A linking label is something like a DSCP.
- The number of linking labels is almost not limited.
  - The number of DSCPs is only 64 — not sufficient!
- The linking label is not put on a packet.
  - The linking label never goes out from a router — it is internal to the router.
  - The linking label may exist out of a packet, or it may be virtual.

## A Model of DiffServ-ready Routers



Networking Architecture Workshop 2000-2-3

7

## Required Primitives: Building Block Rule Types

### ■ Matching rules

- Rules for flow classification.
- Example: `if (Source_ip == 192.168.1.*) ...;`

### ■ Policing rules (Metering rules)

- Rules for policing (bandwidth control, etc.).
- Example: `if (Average_rate <= 1Mbps) ...;`

### ■ Marking rules

- Rules for writing a DSCP.
- Example: `if (...) DSCP = 46;`

### ■ Discarding rules

- Rules for discarding packets.
- Example: `if (...) discard_all;`

### ■ Scheduling rules

- Rules for shaping and/or scheduling packets.
- Example: `if (...) queue_priority = 6;`

Networking Architecture Workshop 2000-2-3

8

## Outline of the Rule-based Language

---

### ■ A policy rule example — a procedural description

- ```
if (Source_ip == 192.168.1.*) {  
    if (Average_rate <= 1Mbps) {  
        DSCP = 46; // EF  
        queue_priority = 6;  
    } else {  
        discard_all;  
    }  
};
```

## Outline of the Rule-based Language (cont'd)

---

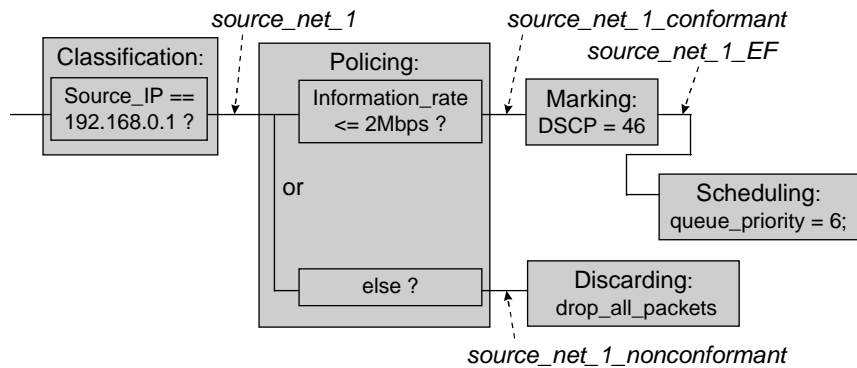
### ■ Representation of the policy in the language

- Matching rule
  - ```
if (Source_ip == 192.168.1.*) Label = s1;
```
- Policing rules
  - ```
if (Label == s1 && Average_rate <= 1Mbps)  
    Label = s1_conformant;  
if (Label == s1 && Average_rate > 1Mbps)  
    Label = s1_non_conformant;
```
- Marking rule
  - ```
if (Label == s1_conformant) {  
    DSCP = 46; Label = s1_EF; };
```
- Discarding rule
  - ```
if (Label == s1_non_conformant) discard_all;
```
- Scheduling rule
  - ```
if (Label == s1_EF) queue_priority = 6;
```

## Outline of the Rule-based Language (cont'd)

### ■ Are decomposed rules too complicated?

- The program not much complicated.



## Building Blocks on Top of Conventional Protocols and/or APIs

### ■ The language can be implemented on top of

- SNMP (using a MIB)
- PIB (using a PIB)
- API (using function calls)

### ■ A MIB/PIB for the building block approach

- A preliminary version was proposed to 46th IETF (November 1999)
  - Draft name: draft-kanada-diffserv-qospifmib-00.txt
  - Presented at:
    - RAP WG (Resource Allocation Protocol WG)
    - CFGMGMT BOF (Configuration Management BOF)
    - Diffserv WG — Q&A only

## Concluding Remarks

---

### ■ Most policy rules for DiffServ can be described using

- Five types of building block rules: matching, policing, marking, discarding, and scheduling.
- Linking labels.

### ■ Building block rule architecture is not restricted to DiffServ.

- Applicable to other QoS services.
- Applicable to Active Networks (programming networks).

### ■ Future work

- Definition and implementation of the rule-based language
  - Including an implementation for Hitachi GR2000.