

IP QoS with Linux

Linuxtag Stuttgart, 29th June - 2nd July 2000

MATTHIAS KRANZ

mskranz@acm.org

GMD FOKUS Competence Center GloNe

Agenda

✍ Introduction

- ⇒ What is *Quality of Service*?
- ⇒ Why do we need it?

✍ The Big Picture

- ⇒ Policy Management
- ⇒ Authentication Infrastructure
- ⇒ Accounting and Billing

✍ The way Linux is doing it

- ⇒ Traffic Control
- ⇒ RSVP
- ⇒ DiffServ

✍ Conclusion

✍ Resources and References

Introduction

Bandwidth is the answer! What's the question?

- ✍ Up to now, "Best effort" service was fine, so why do we need to change it?
- ✍ Traffic has not only increased, but changed its character
- ✍ No problems with *Mail, File Transfer and Web*, but ...
- ✍ ... everybody is talking about *IP-Telephony and Video on Demand* :)

So we need a new technology approach and of course a buzzword!

- Here it is: Quality of Service!

Quality of Service I

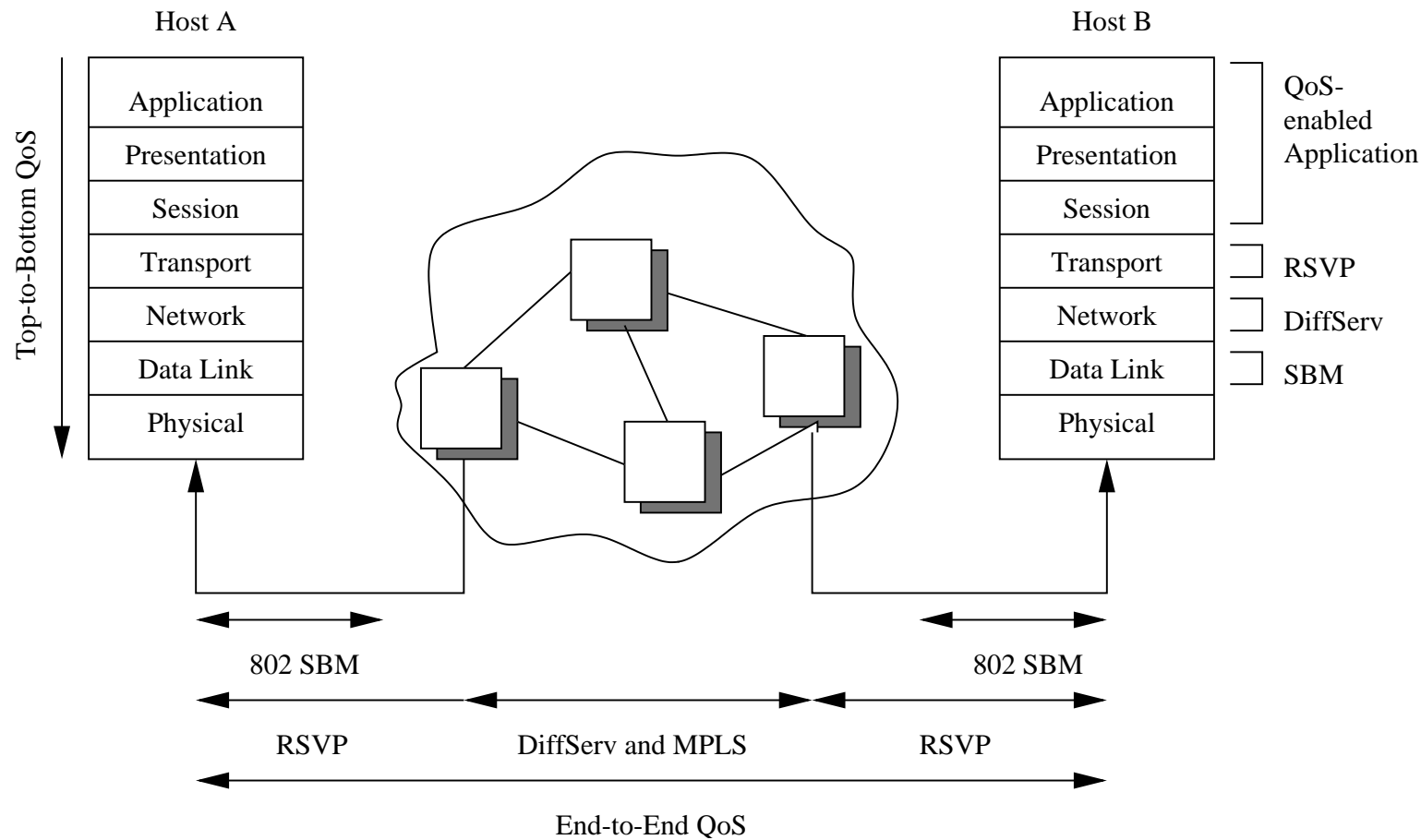
- ✍ The problem is: Quality of Service means different things to different people.
 - ⇒ Flying by airplane - *Economy, Business or First Class*
- ✍ Different service often means different prices
- ✍ The big questions is: How to deploy it to the Internet?
 - ⇒ So far, the Internet was successful 'cause of its simplicity
 - ⇒ Moving complexity from the end points to the core of the net might cause problems.

Quality of Service II

QoS is the ability of a network element to provide some level of assurance for consistent network data delivery.

- ✍ QoS does not create bandwidth, but manages it more effectively
- ✍ Two different types
 - ⇒ Resource Reservation (Integrated Services)
 - ⇒ Prioritization (Differentiated Services)

The Big Picture I



The Big Picture II

✍ Policy Management

- ⇒ Distributed Policy Management via COPS
- ⇒ Policy Enforcement Points retrieve policies from the Policy Distribution Points
- ⇒ Bandwidth Broker

✍ Authentication Infrastructure

- ⇒ Public Key and Certification Infrastructure

✍ Accounting and Billing

- ⇒ Authentication, Authorization and Accounting (AAA)
- ⇒ Bandwidth Broker and Best Carrier Finder

Kernel Configuration

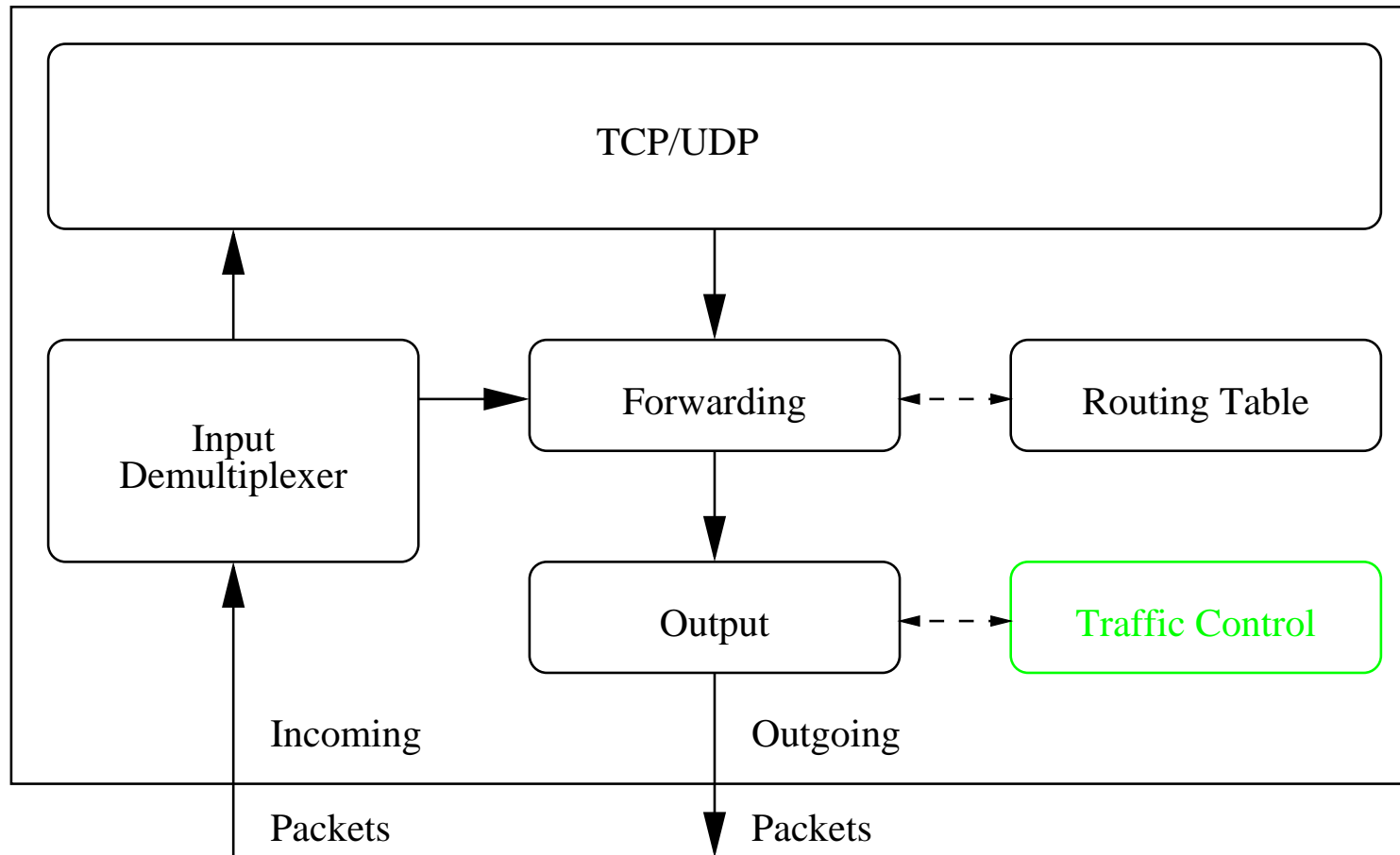
✍ Networking Options

- ⇒ Kernel/User netlink socket, Network firewalls, TCP/IP Networking,
IP: Firewalling, IP: Optimize as router not host

✍ QoS and/or Fair Queuing

- ⇒ Qos and/or fair queuing
- ⇒ Class Based Queuing (CBQ)
- ⇒ Clark-Shenker-Zhang (CSZ)
- ⇒ Priority (PRIO)
- ⇒ Random Early Detection (RED)
- ⇒ Stochastic Fair Queuing (SFQ)
- ⇒ True Link Equalizer (TEQL)
- ⇒ Token Bucket Filter (TBF)

The way Linux is doing it

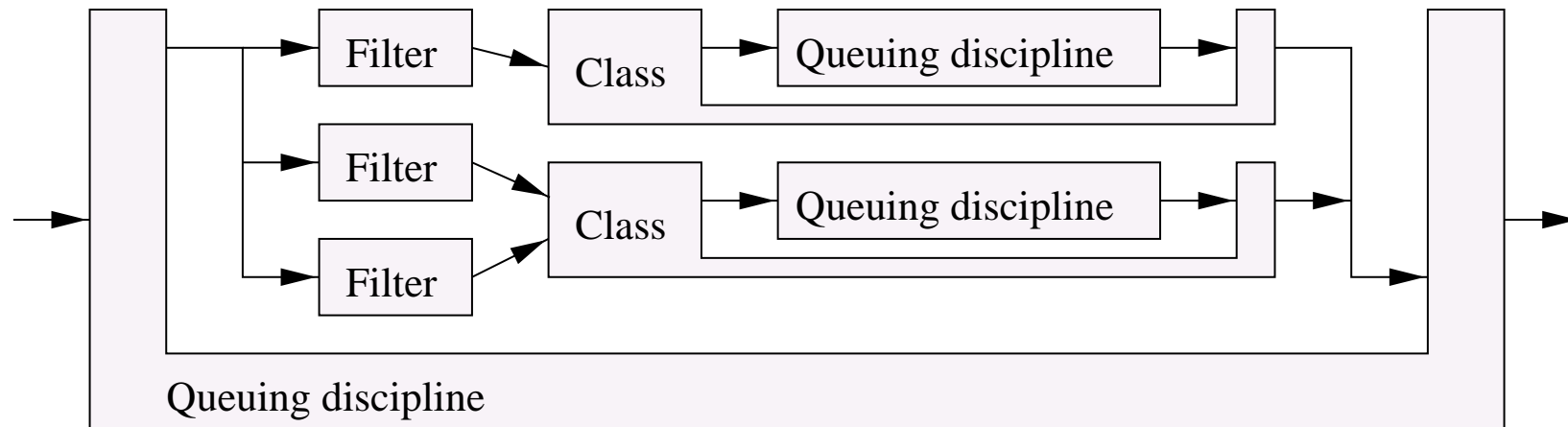


Traffic Control I

- ✎ The TC code consists of the following components
 - ⇒ Queuing Disciplines (QDiscs)
 - ⇒ Classes
 - ⇒ Filters
 - ⇒ Policing
- ✎ Each network device has a Queuing Discipline



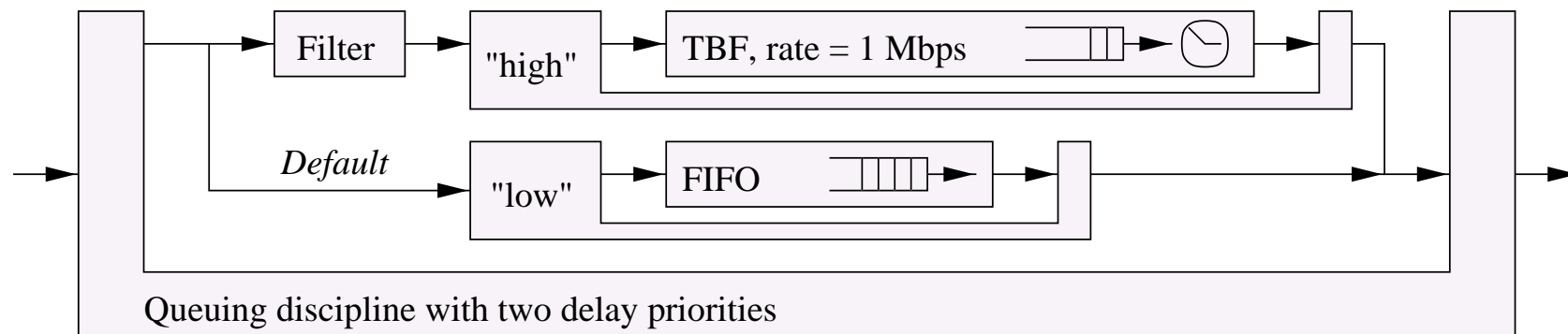
Traffic Control II



- ✍ Filters are used to distinguish between different classes of packets
- ✍ Each class consists of a queuing discipline in turn

Traffic Control III

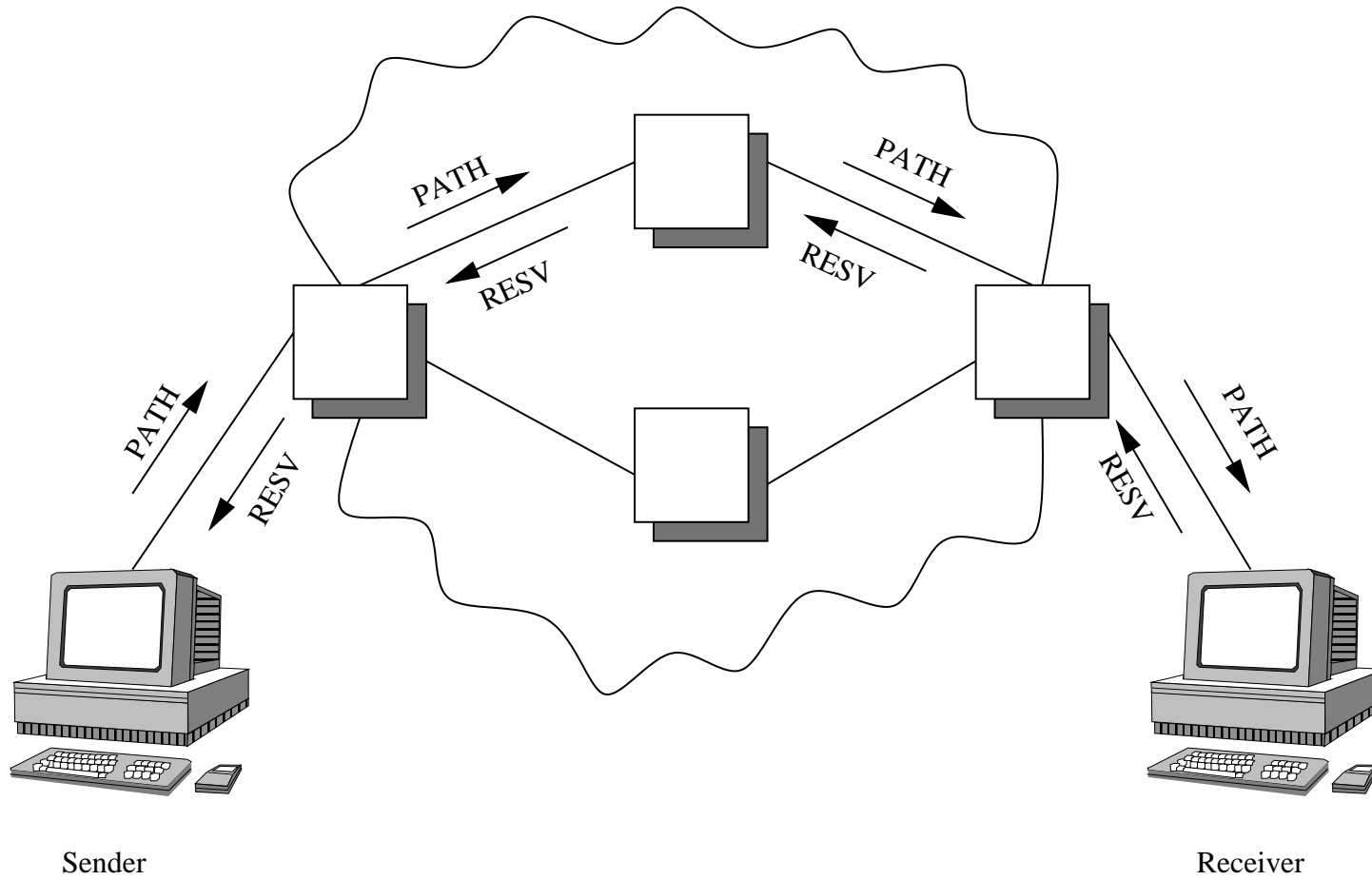
Example



RSVP I

- ✍ Resource **ReSerVation Protocol**
- ✍ Signaling Protocol
- ✍ Kind of Circuit Emulation on IP networks
- ✍ Complex and far away from *traditional IP*
- ✍ Enables **Integrated Services**
 - ⇒ Guaranteed
 - ⇒ Controlled Load

RSVP II



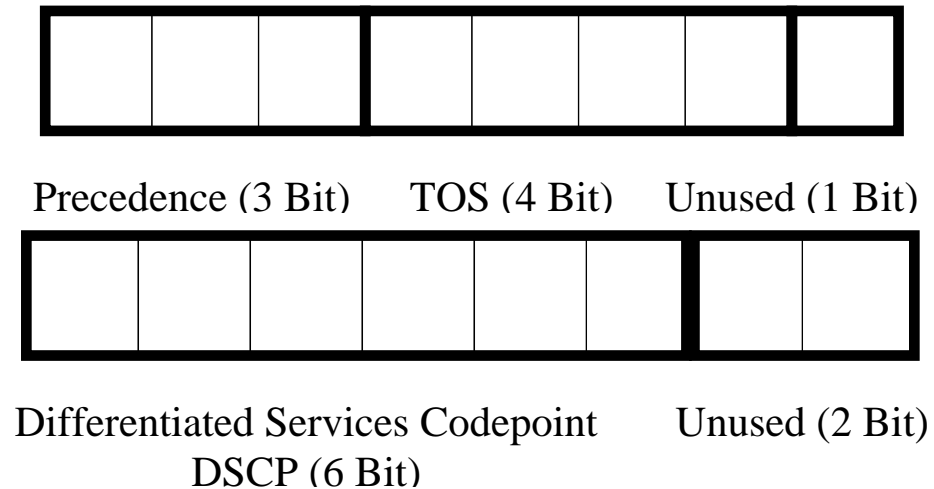
RSVP III

- ✍ Sender characterizes outgoing traffic in terms of the upper and lower bounds of bandwidth, delay, and jitter (PATH)
- ✍ Each router along the downstream route establishes a path-state"
- ✍ Receiver sends a reservation request message (RESV)
- ✍ Each router along the upstream uses the admission control and allocates the necessary resources
- ✍ Last router sends a confirmation message back to the receiver

DiffServ

- ✍ Packets are classified and marked
- ✍ Classification, Policing and Conditioning is only required at network boundaries
- ✍ Inner network nodes only apply certain forwarding rules
- ✍ DS field in the IP header
 - ⇒ TOS Byte (IPv4)
 - ⇒ Traffic Class Byte (IPv6)
- ✍ Two service levels
 - ⇒ Expedited Forwarding (EF)
 - ⇒ Assured Forwarding (AF)

DiffServ II



- ✍ DSCP is used to select the PHB a packet experiences at each node
- ✍ Structure is incompatible with IPv4

Conclusion

- ✍ The Linux Traffic Control Design
 - ⇒ High modularity holds complexity but offers very high flexibility
 - ⇒ Offers solid basis for implementations like Linux DiffServ
- ✍ Since its all Open Source its easy
 - ⇒ to add features
 - ⇒ to fix bugs
 - ⇒ to implement new ideas
 - ⇒ to share the knowledge

Thanks to Alexey Kuznetsov and many more ...

Resources and References I

✍ **DiffServ on Linux Web Site**

<http://icawww1.epfl.ch/linux-diffserv/>

✍ **K.I.D.S. (University of Karlsruhe)**

<http://www.telematik.informatik.uni-karlsruhe.de/forschung/diffserv/KIDS/>

✍ **ITT Center University of Kansas**

<http://qos.ittc.ukans.edu/>

✍ **RSVP** <ftp://ftp.sunet.se/pub/os/Linux/ip-routing/rsvp/>

✍ **Download site iproute2+tc**

<ftp://ftp.inr.ac.ru/ip-routing/>

Resources and References II

- ✧ RFC 2205 - Resource Reservation Protocol
Braden, Zhang, Herzog, Berson, Jamin
- ✧ RFC 2210 - The Use of RSVP with IETF Integrated Services
Wroclawski
- ✧ RFC 2475 - An Architecture for Differentiated Service
Blake, Black, Carlson, Davies, Wang, Weiss
- ✧ RFC 2474 - Definition of the Differentiated Services Field (DS Field)
Nichols, Blake, Baker, Black
- ✧ Linux Network Traffic Control - An Implementation Overview
Almesberger
- ✧ Differentiated Services on Linux
Almesberger, Salim, Kuznetsov
- ✧ Linux - Advanced Networking Overview
Radhakrishnan