Internet QoS

Andrea Bianco
Telecommunication Network Group
firstname.lastname@polito.it
http://www.telematica.polito.it/

Main IETF proposals

- Best-effort
  - «Improve» TCP congestion control features (other slide set)
  - Improve network efficiency through clever discarding policies
- QoS architectures
  - Integrated Services
    - RSVP
  - Differentiated Services
    - Bandwidth Brokers
- MPLS (Multi-Protocol Label Switching) (other slide set)
  - Label swapping in Internet
- Protocols for multi-media applications (other slide set)

QoS building blocks

- In the IETF, some fundamental principles needed to provide QoS were defined
- They are common sense heuristic criteria (although sometimes there is not a full agreement on them)
- Often named principles or postulates
- Derived from concepts already largely explored within B-ISDN (although it cannot be said)

First postulate

- Packet classification (at network edge)
  - Mandatory to permit to switching devices to distinguish between different clients, flows, or traffic classes
  - Fundamental to define different QoS levels to data classified as belonging to different flows
  - It also enables differentiated pricing policies
  - Executed at network edge
    - Either directly by hosts or more likely by dedicated devices in ingress routers
  - Based on IP addresses, applications, services, packet content …..

Second postulate

- Traffic contract verification
  - The service provider can prevent user frauds
  - The user can check whether the received service is conformant to the negotiated one
  - Requires the definition of a known traffic profile
  - Algorithm to measure traffic characteristics: Token Bucket Algorithm (TBA)
  - Special devices shape or police traffic according to a TBA algorithm
  - Executed mostly at network ingress or at networks border

Third postulate

- Flow isolation
  - Traffic separation for data generated by different applications, services, users, flows…
    - Buffer separation
  - Enables QoS aware traffic management according to traffic classes
  - Enables different priorities in the network
  - Must be implemented in all routers
  - Possible algorithms
    - Trunking (resource partitioning)
    - WRR
    - WFQ

Fourth postulate

- Access control (CAC)
  - A new call (?) can be accepted only if:
    - 1) can receive the requested QoS with high probability
    - 2) does not damage the QoS perceived by already accepted calls
    - 3) does not create instability (congestion) in the network

Access control (CAC)

- Available resources signaling
- Requested service signaling
- Algorithms to evaluate the expected QoS

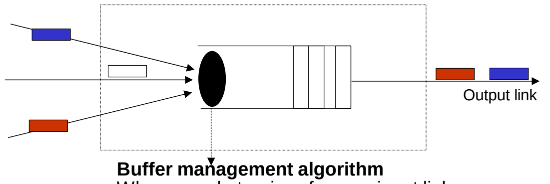
- Devices must support the three above described functions

Fifth postulate

- High resource utilization. Reasons:
  - Service cost kept low
  - High revenues
  - Enabling factor to introduce services with high added value
- To obtain it
  - Statistical multiplexing
  - Statistically described QoS requests
  - Work-conserving scheduling algorithms (WFQ, RR, WRR, PQ, CPQ)
- Not everybody agrees on this idea

Router buffer management

- Packet discarding policies in router buffers have deep impact on both efficiency and fairness



Buffer management algorithm
When a packet arrives from an input link, the algorithm determines whether to accept it or not

Router buffer management

- Two fundamental issues
  - When to drop a packet?
    - When the buffer is full? (Drop-tail)
    - When the buffer occupancy is growing too large? (AQM: Active Queue Management)
  - Which packet to discard?
    - The arriving packet (is congestion caused by this packet?)
    - A packet belonging to the most active flow, i.e., the flow that has the largest number of packets in the buffer (complex)
    - The packet at the head of the queue (it could be too old to be useful)

Router buffer management

- Goals:
  - Control the number of packets in the buffer to:
    - Offer fairness to best-effort flows
    - Protect from non responsive flows (flows not reacting to congestion signals)
    - Obtain a high output link utilization

DropTail buffer management

- The most obvious and simplest algorithm
- Idea: when the buffer is full, drop the arriving packet
- Pros:
  - Easy to implement
  - Large buffer size permit to reduce packet losses
- Cons:
  - All flows punished regardless of their behaviour or service requirements
  - Non the best solution for TCP
    - TCP connection synchronization (many connections experience drops at the same time)
    - Too many losses in the same TX window cause timeout expiration

AQM buffer management

- Active Queue Management (AQM) refers to all buffer management techniques that do not drop all incoming packets
- The most well known AQM algorithm (and one of the first to be proposed) is named RED (Random Early Detection),
  - Several modifications/improvements have been proposed

Random Early Detection

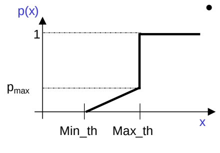
- Simple to implement
  - Works with a single queue
- Not flow aware
- Goal is to obtain a low (not null) average buffer occupancy
  - Low delays useful for multimedia applications and TCP
  - High output link utilization
- Try to approximate a fair dropping policy
- “TCP friendly” packet dropping
  - TCP suffers if packets are lost in bursts
  - If possible, at most one packet loss per window for each TCP connection

Random Early Detection

- Proposed by Sally Floyd and van Jacobson
- Adoption is recommended in RFC 2309
- Most routers adopt it (in some flavor)
- Principles:
  - Detect congestion through measurement of the average buffer occupancy
  - Drop more packets if congestion more severe
  - Drop more packets from more active flows
  - Drop packets in advance, even if the buffer is not full

RED: fundamental principles

- How to detect congestion?
  - Estimate the average buffer occupancy *x* through a low-pass numeric filter
  - Drop packets with probability *p(x)*, adopting a no drop and full drops thresholds



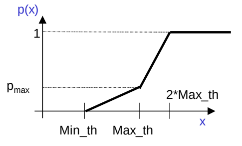
- Why probabilistic dropping?
  - ✓ Avoid dropping several adjacent packets in the same flow
  - ✓ More active flows are statistically more penalized
  - ✓ Avoids TCP connections synchronization

RED: Algorithm

```
Packet arrival :
compute average queue occupancy: avg
if (avg < min_th)
  // no congestion
  accept packet
else if (min_th <= avg < max_th )
  // near congestion, probabilistic drop
  calculate probability Pa
  with probability Pa
     discard packet
  else with probability (1-Pa)
     accept packet
else if avg => max_th
  discard packet
```

RED: problems

- Difficult to correctly set-up algorithm parameters
  - Performance may become worse than droptail (Christiansen et al, SigComm'00)
- When the number of TCP flow is high, p(x) oscillates around p_{max}, making RED unstable (Firoiu-Borden, Infocom'00)
  - To avoid this, gentle RED was proposed



RED: problems

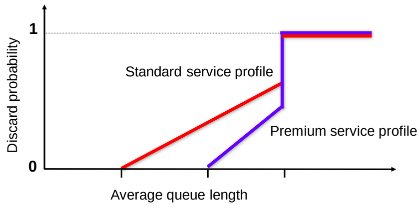
- RED is essentially a closed-loop control algorithm
  - If flows are "non-responsive" (e.g., non adaptative multimedia), RED does not work properly
  - Even when only responsive flows exist, (e.g., TCP or adaptative multimedia), the low-pass filter introduces a delay (Hollot et al., Infocom'01) –
    - Using the instantaneous buffer occupancy?

AQM algorithms

- RED modifications
  - FRED (Ling-Morris, SIGCOMM'97): estimate the number of active flows to punish flows using more bandwidth
  - BRED (Anjum-Tassiulas, INFOCOM'99): Balanced RED to punish flows with more packets stored in the buffer
  - SRED (Lakshman-Wong, INFOCOM'99): Stabilized RED to change p(x) as a function of the number of active flows
  - DRED (Aweya et al., Computer Networks, 2001) changes p(x) as a function of the distance of the queue occupancy from a threshold
- BLUE (http://thefengs.com/wuchang/blue/)
- ...

RED extension: Weighted RED (WRED)

- Differentiate the discard probability for different type of packets



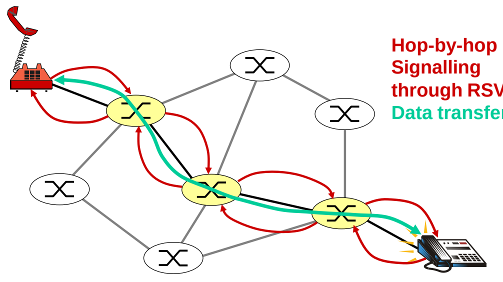
Internet QoS architectures

- IntServ
  - Integrated Services
- DiffServ
  - Differentiated Services

Integrated Services (IntServ)

- Idea is similar to the B-ISDN architecture (but you shouldn't say it)
  - QoS provided to and negotiated for each application flow (first postulate)
  - Police traffic for each flow (second postulate)
  - Nodes are assumed to reserve needed resources for each flow (third postulate)
- Signalling procedure to determine whether or not to accept a flow: each application tries to open a separate flow that may be accepted or rejected (fourth postulate)

IntServ: opening a user call



Integrated Services

- Traffic flow characterized by a vectorial representation
  - The “T-spec” of each flow is the set of parameters that describe the traffic the application will inject in the network
- QoS requirements characterized through a vectorial representation
  - The “R-spec” of each flow is the set of parameters that describe the QoS requests (always associated to a T-spec)
- T-spec and R-spec are used by nodes to establish whether enough resources are available to satisfy a given T-spec R-spec pair

RSVP
Resource ReSerVation Protocol

- Signaling protocol for IntServ
- Hop-by-hop transport service over IP for signaling messages
- Does not specify
  - multicast routing protocols
  - CAC
  - Node resource reservation algorithm
  - How to provide the requested QoS

RSVP: design specifications

- Support for both unicast and multicast
- Support heterogeneous receivers
  - Receiver driven protocol:
    - Receivers ask for the requested QoS
- Automatic adaptation to flow modifications
  - Soft-state
    - Nodes keep state information only for a limited amount of time
    - Resource are not explicitly freed
    - Each reservation must be periodically refreshed, otherwise it is automatically cancelled by a timer expiration

RSVP: notes

- Each data flow issues its own signaling request
- RSVP control messages (e.g., reservation request) are encapsulated in IP datagrams
- No end-to-end ack is required to confirm that a reservation has been made (but failures must be explicit)

RSVP: operations

- Example: multicast audio-conference with a source and a set of registered receivers
- The source sends (almost periodically) PATH messages to the (possibly multicast) address of receivers, containing
  - T-spec (token rate, token bucket depth, minimum policed size, maximum packet size, peak rate, …)
- Each receiver send RESV messages over the inverse path
  - T-spec
  - R-spec (rate to be reserved, end-to-end delay tolerance)
  - Type of Intserv service requested (controlled load or guaranteed quality)
  - F-spec (filter that specifies the subset of packets for which the reservation is being made)

RSVP: operations

- RESV messages, hop per hop, permit to reserve resources requested by receivers
  - – If a router does not have enough resources, it explicitly notifies the receiver(s) that has sent the RESV message
  - – If two or more RESV messages try to reserve resources for the same flow over the same link, a merging procedure activated prior of forwarding the RESV message
    - Merging procedures may become non trivial
- At the end of the session, the source or the receiver(s) send a TEARDOWN message

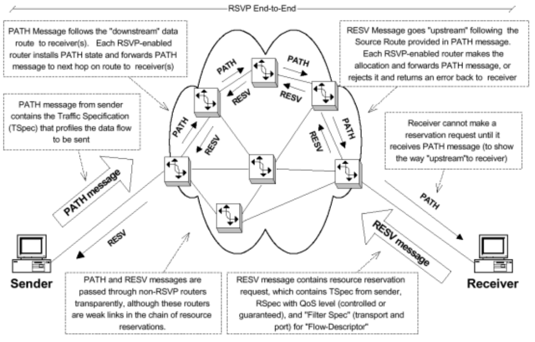
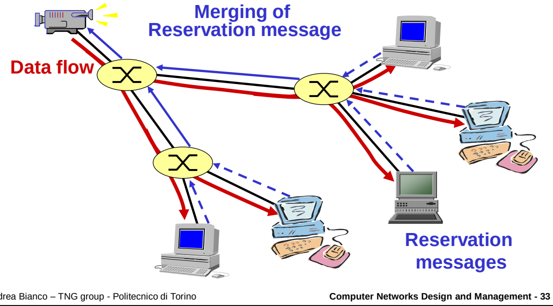


Figure 1: RSVP "PATH" and "RESV" messages are used to establish a resource reservation between a sender and receiver. There is an explicit tear-down of reservations also (not shown).

RSVP: reservation merging



RSVP: the soft state

- RSVP manages route changes natively:
  - If routes are stable, periodic PATH and RESV messages "refresh" the reservation status at intermediate nodes
  - If routes change, new PATH messages automatically identify the new path and new RESV messages will follow the new path trying to make a reservation
  - Not refreshed reservations expire
- The session has a quality guarantee for the whole duration only if routes do not change
  - Over the new path, resources may be not available

Services in IntServ

- Two kinds of services
  - Guaranteed Quality
  - Controlled Load

IntServ: Guaranteed Quality service

- Shenker, S., Partridge, C., and R. Guerin, "Specification of Guaranteed Quality of Service", RFC 2212, September 1997
  - Also named hard real time guarantees
- Both T-spec and R-spec needed
- Provide an absolute a-priori delay bound a packet can observe when traversing a node
  - no guarantees on average delays or on jitter
  - zero losses (reserved buffer)
- Admission control based on worst-case analysis
- Guarantees provided to conformant packets
  - Non conformant packets become best-effort traffic (out-of-order delivery possible)
- Fairly complex, the idea is to emulate a token bucket device in each node for each flow

GQ service: T-Spec

- Traffic defined in the T-spec as
  - Token bucket (r = rate, b = bucket size)
  - peak rate (p)
  - max segment size (M)
  - min segment size (m)
- Traffic is controlled by M + min(pT, rT+b-M) for all T
  - M bits for the current packet
  - M + pT: not more than a packet over the peak rate
  - Not over the token bucket capacity rT+b

GQ service: R-Spec

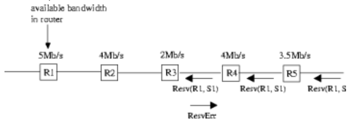
- Minimum flow requirements
  - R: packet sending rate
  - S: maximum admissible slack (end-to-end)
    - amount by which the actual end-to-end delay bound (due to the current reservation of R bandwidth) will be below the end-to-end delay required by the application
      - i.e. anticipation with respect to the required end-to-end delay
    - S must be ≥ 0 otherwise the required end-to-end delay is not satisfied

GQ service: R-Spec

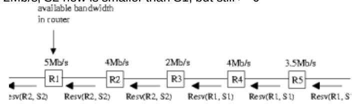
- R-spec are modified by each router
    - (Rin, Sin) input values
    - (Rout, Sout) output values
    - Sin – Sout = max delay in the router
- If the router allocates to the flow
  - a buffer size β
  - a rate ρ (which must be ≥ r)
- Rout = min(Rin, ρ)
- Sout = Sin – β/ρ (when S ≥ 0)
- Flow accepted only if
  - ρ ≥ r (rate bound)
  - β ≥ b (bucket bound)
  - Sout ≥ 0 (delay bound)

The role of slack

- Hp: token bucket rate r=1.5Mb/s
- R1=2.5Mb/s, S1=0



- R1=3Mb/s, S1>0 (anticipation thanks to the higher reserved rate)
- R2=2Mb/s, S2 now is smaller than S1, but still >=0



thanks to the slack, now the request is successful

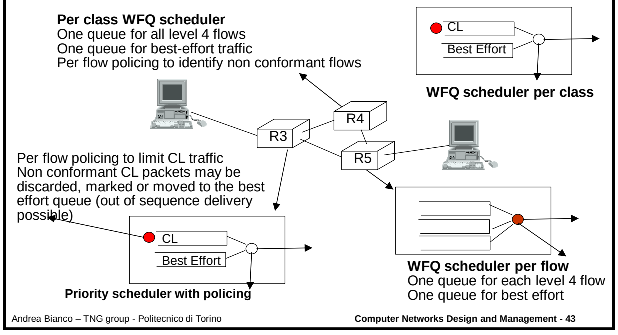
Services in IntServ

- Two kinds of services
  - Guaranteed Quality
  - Controlled Load

IntServ: Controlled Load service

- Wroclawski, J., "Specification of the Controlled-Load Network Element Service", RFC 2211, September 1997
  - Also named soft real time guarantees
- Only T-spec needed
- Provide a quality almost indistinguishable from the QoS obtained if the network element was not overloaded
- No absolute guarantees
  - Only statistical guarantees on delay and losses
- Admission control may be based on measurements
- The main goal is to improve the best effort service for real-time applications

IntServ: Controlled load implementation



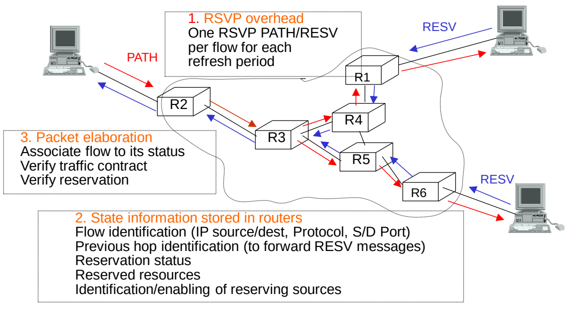
IntServ references

- Request For Comments
  - RFC 1633: General QoS architecture
  - RFC 2210: RSVP – IntServ signaling
  - RFC 2211: “Controlled-Load” service
  - RFC 2212: “Guaranteed Quality” service
  - RFC 2215: Parameters and configuration

IntServ: observations

- Need somehow to rely on fixed path
- Main issue is scalability
  - Lot of signaling messages
  - Each router must keep state information for each session (level 4 flows)
    - RSVP messages must be processed in each router
    - Packet classification is needed
    - Per flow/session policing/queueing/scheduling required
- A precise traffic definition is needed
  - May be difficult/impossible for some applications

IntServ: scalability at a glance



Internet QoS architectures

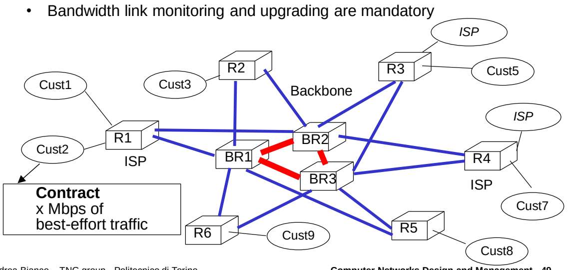
- IntServ
  - Integrated Services
    - main idea: reserve network resources for each flow
- DiffServ
  - Differentiated Services
    - main idea:
      - classify packets into traffic classes
      - preconfigure network behaviour for each class

DiffServ: Differentiated Services

- Simpler network architecture
- Only aggregated flows (classes) are considered (to achieve scalability)
  - QoS definition is per class
- Service models should be flexible
- QoS support without requiring complex signaling

Why DiffServ? ISP problem

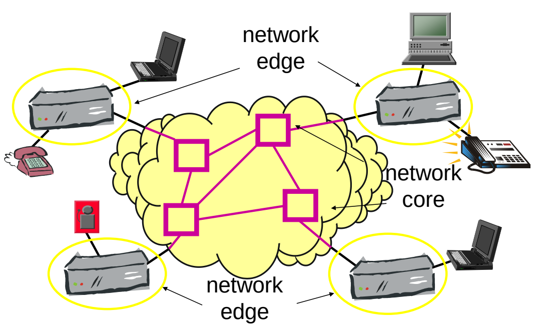
- When designing the network, ISPs make “economical” choices
- For each client the ingress traffic level (worst case) is known, traffic destination is unknown
- Bandwidth link monitoring and upgrading are mandatory



DIffServ: core-and-edge architecture

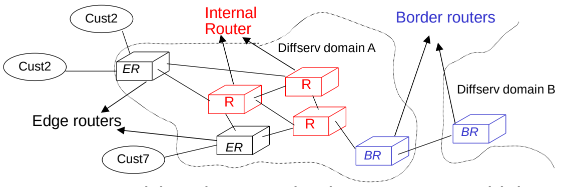
- Differentiation between network edge and network core
- More complex functions (operate at the flow level) executed at network edge only
- Core network concentrate only on few basic functions (operate on large aggregates)

Core-and-edge DiffServ architecture



DiffServ: core and edge architecture

- Network logically separated in two parts



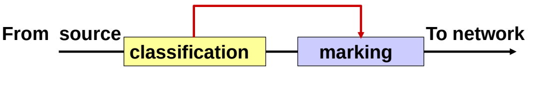
  - Internal (core) router simple, to operate at high speed
  - Complex functions only executed at edge and **border** routers

DiffServ functions

- Edge functions
  - Packet classification (first postulate)
    - Class of service is explicitly written in each IP packet through a marking procedure executed by:
      - client
      - edge router
      - border router
  - Traffic conditioning (second postulate)
- Core functions
  - Packet switching and transmission only according to the class of service to which packets belong to
    - (per-hop-behavior)
  - Complexity scales with the number of services, not with the number of flows
  - Per class of service isolation through proper scheduling (third postulate)

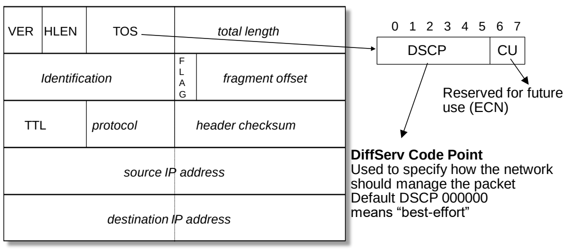
DiffServ: packet classification

- Two aspects:
  - Choosing the traffic class or behavior aggregate (packet classification)
  - Assigning the DSCP (Differentiated Service Code Point) code (packet marking)



DiffServ: packet marking

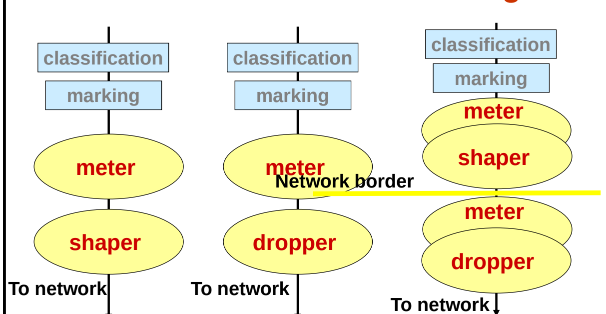
- Redefinition of the ToS byte in the IP header



DiffServ: traffic conditioning

- The conditioning process permits to make a flow of packets with a given DSCP conformant to a given traffic contract
- It adopts a metering device cascaded with a shaper or dropping device

DiffServ: traffic conditioning



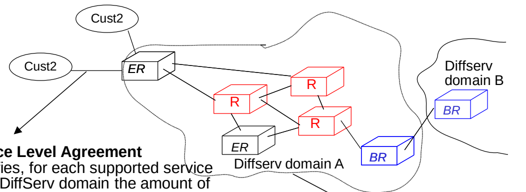
DiffServ: traffic conditioning

- The metering device compares the characteristics of the packet flow with respect to a given traffic contract (traffic profile) (Fourth Postulate)

Meter + Shaper
or
Meter + Dropper

} token bucket device

DiffServ: service providing

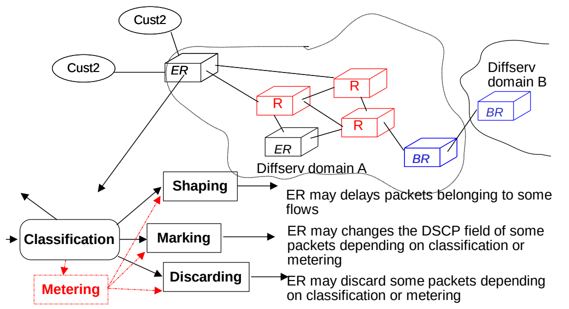


Service Level Agreement
Specifies, for each supported service in the DiffServ domain the amount of traffic that Cust2 can send in the network

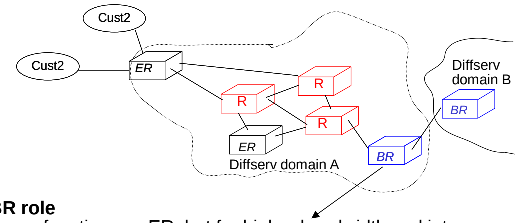
The ISP provides some guarantees:
- Packet losses in a given timeframe
- End-to-end delays
- Network reliability
- ...

Diffserv Codepoints
The network administrator defines the provided services and associates them to a specific DSCP

DiffServ: service providing



DiffServ: service providing



BR role
Same functions as ER, but for higher bandwidth and inter-domain traffic. If needed, may re-execute the marking process if domains A and B use different DSCPs (mapping among different domains)

DiffServ Per-Hop-Behavior (PHB)

- Set of coherent rules that permit to transfer packets according only to their DSCP field
  - Behaviour must be measurable externally, no specification on internal mechanisms
- PHB defined
  - default (best effort)
  - class selector
  - expedited forwarding
  - assured forwarding

Default PHB

- Standardized in RFC 2474
- Base service
- Preserve compatibility with the Internet best-effort service
- DSCP = 000000 (recommended)

DiffServ Per-Hop-Behavior (PHB)

- PHB defined
  - default (best effort)
  - class selector
  - expedited forwarding
  - assured forwarding

Class-Selector PHB

- Defined to preserve compatibility with IP-precedence schemes supported in the network
- The DSCP assumes values as xxx000, where x can be either 0 or 1
- These codes (xxx000) are also named Class-Selector Code Points
- For example, a packet with DSCP=110000 (equivalent to a 110 value in the IP-precedence scheme) is treated preferentially with respect to a packet with DSCP=100000.

DiffServ Per-Hop-Behavior (PHB)

- PHB defined
  - default (best effort)
  - class selector
  - expedited forwarding
  - assured forwarding

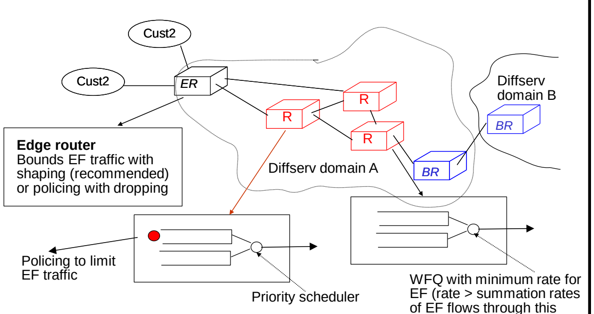
Expedited Forwarding PHB

- Originally standardized in RFC 2598, now RFC 3246
- The service rate of each class is >= than a specified rate, independently of other classes (class isolation)
- Relatively simple definition
- Hopefully, can be obtained with low-complexity algorithms

Expedited Forwarding PHB

- EF can be supported via a priority-queueing (PQ) scheduling jointly with a class-dependent rate-limiting scheme
  - priority-queueing allows unlimited preemption of other traffic, thus a token-bucket rate limiter is needed to limit the damage EF traffic could inflict on other traffic
- EF permits to define a virtual-leased circuit service or a premium service
- The suggested DSCP is 101110.

EF: possible implementation



DiffServ Per-Hop-Behavior (PHB)

- PHB defined
  - default (best effort)
  - class selector
  - expedited forwarding
  - assured forwarding

Assured Forwarding PHB

- Standardized in RFC 2597
- Defines 4 classes with 3 discard priority for each class
  - 12 DSCP
- More complex than EF-PHB
- QoS guarantees may be associated with bit rate, delay, losses and buffering requirements
- Should be used to provide services with a well defined QoS
- The AF behavior is explicitly modeled on Frame Relay's Discard Eligible (DE) flag or ATM's Cell Loss Priority (CLP) capability. It is intended for networks that offer average-rate Service Level Agreements (SLAs) as FR and ATM

Assured Forwarding PHB

- QoS similar to the IntServ Controlled Load Service
- Traffic may be subdivided into several classes
  - An example: Olympic service
    - Gold: 50% of the available bit rate
    - Silver: 30% of the available bit rate
    - Bronze: 20% of the available bit rate

Assured Forwarding PHB

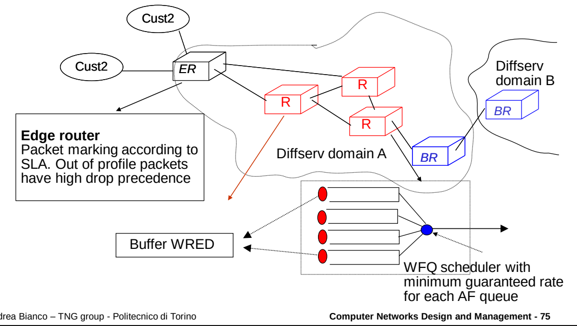
- Up to 4 AF classes may be defined: AF1 (worst), AF2, AF3, AF4 (best).
- To each class a pre-defined amount of available buffer and bit rate at each interface is assigned, according to SLA specifications
- To each class, three different drop-precedence levels can be assigned
  - Implies the use of AQM scheme

Assured Forwarding PHB

- An AF class is specified via a DSCP value in the form xyzab0, where
  - xyz may assume the values {001,010,011,100}
  - ab describes the drop precedence level

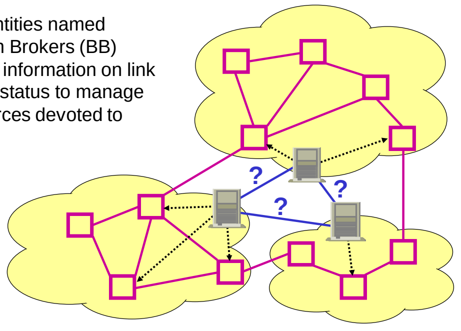
Drop Precedence	Class 1	Class 2	Class 3	Class 4
Low drop precedence	001010 AF11	010010 AF21	011010 AF31	(best) 100010 AF41
Medium drop precedence	001100 AF12	010100 AF22	011100 AF32	100100 AF42
High drop precedence	(worst) 001110 AF13	0101110 AF23	011110 AF33	100110 AF43

AF: possible implementation



DiffServ network management

Special entities named Bandwidth Brokers (BB) exchange information on link and node status to manage the resources devoted to PHB



DiffServ: Request For Comments

- RFC 3260: New Terminology and Clarifications for Diffserv
  - RFC 2474: Definition of the Differentiated Services Field (DS Field) (formats)
  - RFC 2475: An Architecture for Differentiated Services (the base architecture)
  - RFC 2597: Assured Forwarding PHB Group **(**service models)
- RFC 2638: A simplified architecture
- RFC 2697: Single rate Three Color Markers (srTCM)
- RFC 2698: Two rate Three Color Marker (trTCM)
- RFC 3246: An Expedited Forwarding PHB (Per-Hop Behavior) (service models)
- RFC 3290: An Informal Management Model for Diffserv Routers
- RFC 4594: Configuration Guidelines for DiffServ Service Classes

DiffServ: marker and shapers

- Two main markers/shapers defined:
  - srTCM (Single Rate Three Color Marker)
  - trTCM (Two Rates Three Color Marker)
- Label packets as green, yellow or red
  - Color may be associated with a DSCP (or to a AF drop precedence)
- Possible packet management
  - Drop red packets
  - Forward as best effort yellow packets
- Two behaviours
  - Color blind
    - Packets to be marked/shaped are received colorless
  - Color aware
    - Packets to be marked/shaper are received already colored

DiffServ: srTCM

- Based on three parameters:
  - CIR (Committed Information Rate)
  - CBS (Committed Burst Size)
  - EBS (Excess Burst Size)

1 rate + 2 levels of burstiness

- Green packet if within CBS, yellow packet if within CBS+EBS, red if it exceeds CBS+EBS
- Meter exploits two token buckets, named C and E, both generating tokens at rate CIR
- At algorithm startup
  - TB_C = CBS
  - TB_E = EBS
- Token bucket sizes TB_C and TB_E incremented at rate CIR (but create a token in E only when C is full)

DiffServ: srTCM

- When a packet of size B is received at time t
- Color-blind marker
  - if TB_C(t) - B >= 0
    - Green packet and TB_C=TB_C-B
  - else, if TB_E(t) - B >= 0
    - Yellow packet and TE_C=TE_C-B
  - else red packet
- Color-aware marker
  - if TB_C(t) - B >= 0 AND color=green
    - Green packet and TB_C=TB_C-B
  - else, if TB_E(t) - B >= 0 AND (color=green OR color=yellow)
    - Yellow packet and TE_C=TE_C-B
  - else red packet

DiffServ: trTCM

- Based on four parameters:
  - PIR (Peak Information Rate)
  - PBS (Peak Burst Size)
  - CIR (Committed Information Rate)
  - CBS (Committed Burst Size)

2 rates + 2 levels of burstiness

- Yellow packet if it exceeds CIR, red if it exceeds PIR, else green
- Meter exploits two token buckets, named P and C, generating tokens at rate PIR and CIR respectively
- At algorithm startup
  - TB_P = PBS
  - TB_C = CBS
- Token bucket sizes TB_P and TB_C incremented at rate PIR and CIR up to the values PBS and CBS

DiffServ: srTCM

- When a packet of size B is received at time t
- Color-blind marker
  - if TB_P(t) - B < 0
    - Red packet
  - else, if TB_C(t) - B < 0
    - Yellow packet and TB_P = TB_P - B
  - else
    - Green packet and TB_P = TB_P–B and TB_C=TB_C-B
- Color-aware marker
  - if TB_P(t) - B <0 OR color=red
    - Red packet
  - else if TB_C(t) - B < 0 OR color=yellow
    - Yellow packet and TB_P = TB_P-B
  - else
    - Green packet and TB_P = TB_P–B and TB_C=TB_C-B

DiffServ: Service Classes as in RFC 4594

- A service class is a set of packets requiring a specific set of delay, loss and delay jitter
- Packets generated by similar applications are aggregated in the same service class
- RFC 4594 objectives:
  - Present a diffserv "project plans" to provide a useful guide to Network Administrators in the use of diffserv techniques to implement quality-of-service measures appropriate for their network's traffic
  - describes service classes configured with Diffserv and recommends how they can be used and how to construct them using (DSCPs), traffic conditioners, PHBs, and AQM) mechanisms. There is no intrinsic requirement that particular DSCPs, traffic conditioners, PHBs, and AQM be used for a certain service class, but as a policy and for interoperability it is useful to apply them consistently.

DiffServ: Service Classes as in RFC 4594

- Service class definitions based on the different traffic characteristics and required performance
- A limited set of service classes is required. For completeness, twelve different service classes are defined
  - two for network operation/administration (signalling, management traffic)
  - ten for user/subscriber applications/services
- Network administrators are expected to implement a subset of these classes
- Service classes defined through
  - traffic characteristics
  - tolerance to delay, loss and jitter

DiffServ: Service Classes

Service Class	Traffic characteristics	Tolerance to		
		Loss	Delay	Jitter
1. Network control	Variable size packets Mostly inelastic short messages, bursty (BGP)	Low	Low	Yes
2. OAM	Variable size packets, Elastic & inelastic flows	Low	Medium	Yes
3. Telephony	Variable size packets Constant emission rate Inelastic and low-rate flows	Very low	Very low	Very Low
4. Signalling	Variable size packets Short-lived flows	Low	Low	Yes
5. Multimedia Conferencing	Variable size packets Constant transmit interval Rate adaptive. reacts to loss	Low Medium	Very Low	Low
6. Real-time interactive	RTP/UDP streams, inelastic Mostly variable rate	Low	Very Low	Low

DiffServ: Service Classes

Service Class	Traffic characteristics	Tolerance to		
		Loss	Delay	Jitter
7. Multimedia streaming	Variable size packets Elsatic with variable rate	Low Medium	Medium	Yes
8. Broadcast Video	Constant and variable rate Inelastic, non bursty traffic	Very Low	Medium	Low
9. Low latency data	Variable rate, bursty Short lived elastic flows	Low	Low Medium	Yes
10. High-throughput data	Variable rate, bursty, Long –lived flows	Low	Medium High	Yes
11. Standard	A bit of everything	Not specified		
12. Low priority data	Non real time and elastic	High	High	Yes

DiffServ: DSCP Values

Service Class	DSCP Name (reccomm)	DSCP Value (reccomm)	Application Examples
1. Network control	CS6	100000	Network Routing
2. OAM	CS2	010000	OAM
3. Telephony	EF	101110	IP Telephony Bearer
4. Signalling	CS5	101000	IP Telephony Signalling
5. Multimedia Conferencing	AF41 AF42 AF43	100010 100100 100110	H.323/V2 video conferencing (adaptive)
6. Real-time interactive	CS4	100000	Video Conferencing and Interactive gaming

DiffServ: DSCP Values

Service Class	DSCP Name (reccomm)	DSCP Value (reccomm)	Application Examples
7. Multimedia streaming	AF31 AF32 AF33	011010 011100 011110	Streaming video and audio on-demand
8. Broadcast Video	CS3	010000	Broadcast TV and live events
9. Low-Latency Data	AF21 AF22 AF23	010010 010100 010110	Client-server transcations Web-based ordering
10. High-Throughput Data	AF11 AF12 AF13	001010 001100 001110	Store and forward applications
11. Standard	DF (CS)	000000	Undifferentiated applications
12. Low-Priority Data	CS1	001000	Any flow that has no BW assurance

DiffServ: Service Classes as in RFC 4594

- Further information defined for each service class
  - Type of conditioning executed in the border node or in the first DF node
  - Type of queueing to be adopted
    - Rate queueing (WRR; WFQ)
    - Priority queuing
  - If AQM techniques should be used

A possible integrated solution